Some examples and graphics depicted herein are provided for illustration only. No real association or connection to ServiceNow products or services is intended or should be inferred.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Please read the ServiceNow® Website Terms of Use at www.servicenow.com/terms-of-use.html
# Table of Contents

**Now Platform App Engine**................................................................................................. 1
  Creating applications................................................................................................................ 5
    Parts of an application............................................................................................................ 7
  Contextual development environment..................................................................................... 25
  Creation restrictions across application scopes........................................................................ 73
  Delegated development and deployment.................................................................................. 75
  Application collaboration.......................................................................................................... 90

**Application Management**.................................................................................................... 95
  Opt in to the ServiceNow Store products.................................................................................. 96
  Manage entitlements from ServiceNow instance........................................................................ 97
  Skipped records that occur during application installation...................................................... 99
  Accessing unpurchased recommended apps on ServiceNow Store........................................ 103
  Request and install free application trial.................................................................................. 104
  Request a free or paid application........................................................................................... 105
  Complete the offline application purchase transaction........................................................... 107
  Schedule installation................................................................................................................ 108
  Install or update all applications in a ServiceNow product...................................................... 111
  Install multiple applications in a batch..................................................................................... 112
  Install a ServiceNow Store application.................................................................................... 113
  Install an update to a ServiceNow Store application.............................................................. 114
  View the upgrade history of a ServiceNow Store application................................................. 115
  Select an application from the application picker.................................................................... 116
  Repair a ServiceNow application............................................................................................. 117
  Preserve unpublished applications during a system clone....................................................... 117
  Opt out of the ServiceNow Store products............................................................................. 120
  Application sharing.................................................................................................................. 120
  Rolling back and removing applications................................................................................. 132
  Domain separation and Application Management.................................................................. 136

**Application tools**.................................................................................................................. 136
Now Platform App Engine

The Now Platform® App Engine provides a single mobile and web application development platform to build business applications and extend existing apps that power your digital transformation.

Build with no code or pro code

- ServiceNow Studio
- Automated Test Framework
- Web Services
- Scripts

Build or extend apps to accelerate your business

With the Now Platform capabilities as your foundation, developers of all levels can reduce time to market and speed up delivery. You get full-stack development power with an application structure that is set up in the base system.

View and download the full infocard for a highlight of Now Platform App Engine features.

Build new apps fast

Turn business processes into digital workflows to deliver faster, smarter experiences. Build new apps with no-code development tools.
### Extend your ServiceNow apps

Extend apps using a shared code set. You can build business apps from scratch or extend base system ServiceNow applications—all on one platform.

### Turn anyone into a developer

From no-code to pro-code, enable anyone to innovate with a full stack of developer tools. Develop faster with application templates and reusable components.

### Integrate everything

Quickly connect your systems and apps. The Now Platform provides a rich set of APIs and tools for integrating with virtually any service or third-party app.

## Build new apps fast

Use base system solutions like Guided Application Creator to quickly build an app to automate a manual process. Get straight to work using standard tools and technologies, such as JavaScript and Angular, or use a no-code or low-code solution to get started quickly. The Now Platform comes with intuitive graphical tools that make app design easy even if you’ve never built an app before. And when you’re ready to grow the power of your custom apps even further, it’s easy to add new business logic with the ServiceNow development framework.

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Extend your ServiceNow apps

Custom apps enable you to extend the services you offer. For example, when using ServiceNow HR, instead of using email to handle time-off requests, build an easy-to-access and easy-to-use custom application on the Now Platform UI to capture this information as an extension of your ServiceNow HR solution.

Turn anyone into a developer

If your organization has pent-up demand for custom applications across business teams, you can delegate certain capabilities to business stakeholders and analysts and work together to build apps that meet specific business needs.
Building your custom applications on the Now Platform centralizes activities and data on a single system of record, which helps to break down data silos and unlock insights across the enterprise.

**Integrate everything**

![REST API Explorer](image)

When your project does require the full power of coding for more complex workflow designs, integrations, or user experiences, use Now Platform to get straight to work. Write custom code using standard tools such as JavaScript and Angular, all within an integrated development environment. The Now Platform APIs, integration tools, and analytics capabilities help you address business problems faster by integrating valuable resources inside and outside the enterprise.

**Get started**

- Access developer trainings, connect with the developer community, and explore a personal developer instance through the ServiceNow® Developer Site.
- Sign up for the ServiceNow Application Development Fundamentals course to learn to design, build, test, and deploy an application.
- Connect, collaborate, and get answers on the Now Community by visiting the Developer forum.
Applications and features

- App Engine Studio
- Automated Test Framework (ATF)
- Custom components
- Delegated development
- Guided Application Creator
- JavaScript API reference
- REST API reference
- Scripts
- ServiceNow application repository
- ServiceNow Extensions for Visual Studio Code
- ServiceNow Studio
- Service Creator
- System update sets
- Team Development
- UI Builder
- Web services
- Developer guides

Creating applications

Develop an application in the Now Platform that can help your organization store information and manage business processes.

Basic application development process

<table>
<thead>
<tr>
<th>Process</th>
<th>Description</th>
<th>Recommended tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Define business requirements.</td>
<td>Create business requirements and track their progress. Your requirements should answer questions such as:</td>
<td>- Project Portfolio Suite</td>
</tr>
<tr>
<td></td>
<td>• What does the application need to do?</td>
<td>- Visual Task Boards</td>
</tr>
<tr>
<td></td>
<td>• Who will use it?</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• How will you know if it works?</td>
<td></td>
</tr>
<tr>
<td>Process</td>
<td>Description</td>
<td>Recommended tools</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>-------------------</td>
</tr>
</tbody>
</table>
| 2. Define the data model. | Identify what information the application needs to track.  
- What information needs to be captured as a record?  
- What references are there between records?  
- Are there any existing tables you can extend? | ServiceNow Studio |
| 3. Build the application. | Create application and configuration records.  
1. Create a custom application record and set the application scope.  
2. Create an application data table to store application-specific data.  
3. Design the user interface, such as how it appears in the Now Platform, an application workspace, or the ServiceNow mobile app.  
4. Set application access settings to permit or restrict other applications from accessing application data.  
5. Add business logic and automation to meet your business requirements.  
Support multiple developers working on the application.  
- Push the application to other instances in the team development environment.  
- Link the application to a Git repository to save and manage multiple versions. | Guided Application Creator  
ServiceNow Studio  
Flow Designer  
Team Development |
Basic application development process (continued)

<table>
<thead>
<tr>
<th>Process</th>
<th>Description</th>
<th>Recommended tools</th>
</tr>
</thead>
</table>
| 4. Test the application.     | Verify the application meets your business requirements. Your testing should cover:  
• Record operations such as create, read, update, and delete.  
• User interface elements such as views and UI policies.  
• Runtime operations such as business rules and event script actions. | • Project Portfolio Suite  
• Visual Task Boards                                                                 |  
| Share the application.       | Share the completed application with others.  
• Publish the application to the ServiceNow application repository to share it with other instances assigned to the same company.  
• Publish the application to the ServiceNow Store to share it with everyone.  
• Publish the application as an Update Set to comply with a change management or backup policy. | • ServiceNow application repository  
• System update sets  
• ServiceNow Studio                                                                 |  

For more information, see Application sharing.

Watch this 11-minute video to learn more about defining business requirements for an app on the Now Platform. Sets the use case scenario for the Builder video series and answers questions to define business requirements for an app on the Now Platform.

Watch this seven-minute video to learn about steps that are irreversible in the app-building process. Explains steps that are irreversible in the app-building process, so that you can make informed decisions that will facilitate maintenance of your app going forward.

Parts of an application

Applications consist of several types of files and records that collectively deliver a service.
Custom application record

The custom application record defines and identifies an application and all its associated artifacts.

It is similar to a system dictionary record for a table or column in that it stores the most current configuration of an application. The system automatically creates a custom application record during the application creation process. Application developers can use this record to perform the following tasks.

- Change the application name
- Change the application version
- View the scope the system uses to identify application files and configuration records
- Enable scoped administration
- Manage design and runtime access to the application
  - Select what JavaScript standards the application supports
  - Select how the system tracks runtime API operations
  - Permit or restrict access to tables from other applications
• Monitor or enforce subscriptions
• Select the default menu in which to display application modules
• Set the user role required to access the application
• Add or update a logo
• View all application files
• View resources from other applications on which the application depends
• View the run-time resource to which the application has been granted access
• View the design-time resources to which the application has been granted access

**Application versions**

Each installed application has a version as defined by its application developer in the custom application record.

The system uses this version information to determine whether there are updates available from the ServiceNow application repository or ServiceNow Store.

**Application scope**

Application scoping protects applications by identifying and restricting access to application files and data.

Administrators can specify what parts of an application are accessible to other applications from:

• The Custom application record
• Each application Table record

For example, suppose that you create a conference room booking application in its own application scope. By default, the application can access and change its own tables and business logic but other applications can't unless you give them explicit permission. The application scope ensures:

• The conference room booking application does not interrupt core business services.
• Other applications do not interfere with its normal functioning.

By default, all custom applications have a private scope that uniquely identifies them and their associated artifacts with a namespace identifier. The application scope prevents naming conflicts and allows the contextual development
environment to determine what changes, if any, are permitted. Application developers specify an application scope when they create an application.

**Tip:** Global apps can alter data that you don’t intend to alter. You should leverage scoped apps to create new tables, and consider transitioning old ones to scoped apps. This allows you to split responsibilities with Delegated Development. To learn more about delegated development, see Delegated development and deployment.

Selecting the **Can Edit Application in Studio** option does not affect any custom or global applications in development on an instance. If you are the owner and choose to publish the application, you can restrict the development of customizations in Studio of. If you set **Can Edit Application in Studio** to false and then publish it, those who download the application to their own instances won’t be able to edit the application in Studio. But they will have access to **Source Control** features inside of Studio.

**Related information**

Runtime access to applications tables

**Global scope**

The global scope is a special application scope that identifies applications developed prior to application scoping, or applications intended to be accessible to all other global applications.

Applications in the global scope do not append a unique namespace identifier to the application name. Global applications can have naming conflicts and data collisions when developers create multiple global applications with the same name.

Since all global applications are in the same scope, they bypass scope protections. Global applications allow other global applications access to their tables to
• Read records
• Run API requests
• Create configuration records

Typically, only applications provided by ServiceNow are in the global scope, however any custom applications created before application scope was implemented are also in the global scope.

Applications in the global scope are eligible for upload to the application repository, but not to the ServiceNow Store.

Related information
- Publish an application to the application repository
- Publish an application to the ServiceNow Store

Namespace identifier
The system adds a namespace identifier to the front of application artifacts such as tables, scripts, and configuration records.

The identifier cannot be changed or removed from application artifacts to ensure that they are always associated to the proper application and that they have a unique name.

The system generates a namespace identifier from the following information:

<table>
<thead>
<tr>
<th>Elements used to generate a namespace identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Element</strong></td>
</tr>
<tr>
<td>The prefix characters for a scoped application.</td>
</tr>
<tr>
<td>The instance customer prefix</td>
</tr>
<tr>
<td>(glide.appcreator.company.code)</td>
</tr>
<tr>
<td>The application ID</td>
</tr>
</tbody>
</table>
Elements used to generate a namespace identifier (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Requirements</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_&lt;glide.appctreator.company.code&gt;_.</td>
<td>Application developers set this ID when they create the application. The system uses the application name by default.</td>
<td></td>
</tr>
</tbody>
</table>

The example values generate a namespace identifier of x_acme_book_rooms.

Namespace identifier examples

The following examples illustrate generating namespace identifiers for applications, tables, and fields.

Example namespace identifiers

<table>
<thead>
<tr>
<th>Action</th>
<th>Element generated</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Generate a namespace identifier for a private scope application called Book Rooms.</td>
<td>x_acme_book_rooms</td>
<td>This is a combination of the vendor prefix and application ID.</td>
</tr>
<tr>
<td>2. Generate a namespace identifier for a global scope application called Marketing Events.</td>
<td>None</td>
<td>The system does not generate namespace prefixes for global applications.</td>
</tr>
<tr>
<td>3. Add the conference rooms table</td>
<td>x_acme_book_rooms_conference_rooms</td>
<td>This table is in the Book Rooms scope and begins with the namespace identifier.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Example namespace identifiers (continued)

<table>
<thead>
<tr>
<th>Action</th>
<th>Element generated</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>to the Book Rooms application.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Add a Marketing Event table to a global</td>
<td>u_marketing_event</td>
<td>Custom tables in the global scope always use the u_ namespace identifier.</td>
</tr>
<tr>
<td>application.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. Select Book Rooms in the application picker</td>
<td>capacity</td>
<td>The field is in the same scope as the table so it does not have its own namespace identifier.</td>
</tr>
<tr>
<td>and add the <strong>Capacity</strong> field on the Conference</td>
<td></td>
<td>However, to dot-walk to the field in a script, you would use the full path including the</td>
</tr>
<tr>
<td>Rooms table.</td>
<td></td>
<td>table namespace identifier: x_acme_book_rooms_conference_rooms.capacity.</td>
</tr>
<tr>
<td>6. Select Book Rooms in the application picker</td>
<td>x_acme_book_rooms_theme</td>
<td>The field is in a different scope from the table so it is prefixed with the namespace id:</td>
</tr>
<tr>
<td>and add the <strong>Theme</strong> field to the Marketing Event</td>
<td></td>
<td>x_acme_book_rooms. To dot-walk to the field in a script, you would use full path including the</td>
</tr>
<tr>
<td>table.</td>
<td></td>
<td>field namespace identifier: u_marketing_event.x_acme_book_rooms_theme.</td>
</tr>
<tr>
<td>7. Select Marketing Events in the application</td>
<td>u_theme</td>
<td>Custom fields in the global scope use the u_ prefix. To dot-walk to the field in a script,</td>
</tr>
<tr>
<td>picker and</td>
<td></td>
<td>you would use u_marketing_event.u_theme.</td>
</tr>
</tbody>
</table>

**Note:** This example assumes that the Marketing Event table allows other application scopes to add fields. For more information, see Application Access Settings.
Example namespace identifiers (continued)

<table>
<thead>
<tr>
<th>Action</th>
<th>Element generated</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>add the <strong>Theme</strong> field to the Marketing Event table.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Application tables

Application developers create tables and their associated lists and forms for users to add and update records.

An application owns its tables and determines whether other applications can access resources from them. For example, the Book Rooms application can store conference room data in the Conference Rooms [x_acme_book_rooms_conference_rooms] table and permit other applications to read this data.

The system uses standard access controls to manage user access to application data. During application creation, developers can specify an application-specific user role for these access controls. They can also use application access settings to manage runtime and design time access to application tables.

⚠️ **Note:** Certain Now Platform subscriptions include custom table allotments. You can create custom tables for any purpose, up to the allotment limit in the subscription. To learn more about how your usage administrator allocates the custom tables you create to subscriptions, see Allocating your custom tables to a subscription allotment.

### Related information

- Application access settings

### User interface elements

By default, the system creates a list and form view for each table.

Application developers can configure the layout of these views to provide a basic user interface. In addition, they can also create supporting menus, modules, or UI pages to access these list and form views.

Alternatively, developers can create their own custom user interface with Service Portal or the Content Management System.
Related information

Content Management System

**Application user roles**

Application developers can create application-specific user roles to control access to application data.

Application developers can create an end-user role for the primary users of a custom application. The system automatically restricts access to application data by:

- Standard record operations on application tables are restricted to the application user role or the admin role.
  - Create
  - Read
  - Write
  - Delete
- The application menu is restricted to the application user role or the admin role.
- All application modules are restricted to the application user role or the admin role.

Application developers can create additional user roles to support the application but must manually create any associated access controls or role requirements.

**Web services integrations**

Application developers can create web services integrations to application tables.

By default, the system allows other applications to access application tables using web services integrations. Application developers can allow or restrict web services access from the table application access settings.

**Dependencies for custom applications**

Every custom application record includes a related list identifying its dependencies on other applications.
Administrators can review this list to determine whether an application poses any risk to existing processes or data. Application developers can use this list to ensure that their applications have the proper access to other applications.

Sample application dependencies

<table>
<thead>
<tr>
<th>Application Files (21)</th>
<th>Dependencies (1)</th>
<th>Cross scope privileges</th>
<th>Design Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dependencies</td>
<td>New</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Sample application dependencies

Application files

Application files are configuration records that allow developers to extend application functionality.

Application developers create application files when they add configuration records for application logic such as business rules, workflows, and script includes. Scoped applications do not own these tables, but they do own the records (files) within these tables. For example, adding a business rule to check for available rooms from the Conference Room table adds an application file to the Business Rule [sys_script] table. Application developers can view the complete list of application files from the custom application record.

The Application File [sys_metadata] table is the parent table for all tables that contain configuration records. It provides a series of standard fields that define the attributes for a configuration record. Tables that contain configuration records extend the Application File table. For example, the Business Rule [sys_script] table extends the Application File table.

Developers do not create application file records directly from the Application File table. Instead, they create or modify configuration records, and the system creates or modifies the associated application file record. Most configuration record tables do not display a reference field or related list for the Application File table. By default, only Applications [sys_scope] and Tables [sys_db_object] have a reference to the Application File table.

The system also tracks application file records in Update Sets. Whenever you change an application file record or its related configuration record, the system adds a corresponding record in the Customer Updates [sys_update_xml] table. The system uses a single update record, ensuring that a configuration record and its application file are kept in sync when transferring applications between
instances. To avoid collision, users are warned when they edit an application file that has been previously edited in another Update Set.

Administrators can:

- View file properties for configuration records.
- Protect application files from changes during upgrades.
- View parent-child relationships between configuration records.

Related information

Add an application file to an application

View file properties

Administrators can view the application file properties of a single record.

Procedure

1. Navigate to the form view of the configuration record. For example, navigate to System Definition > Business Rules and select a business rule for the Incident table.
2. Right-click the form header and select Show File Properties.

The Application File table provides the standard fields that define the attributes for the configuration record.

3. To return to the configuration record view, click the Show Related Record related link.
4. Navigate between a customer update record, the file properties view, and the configuration record view.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show Related Record related link</td>
<td>Navigate to the configuration record</td>
</tr>
<tr>
<td>Show Parent Record related link</td>
<td>Navigate to the parent record of the current configuration record.</td>
</tr>
<tr>
<td>Descendants related link</td>
<td>View child configuration records, such as a field label translation.</td>
</tr>
</tbody>
</table>

**Application File form**

Use the Application File form to view relationships between applications and configuration records.
# Application File form fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display name</td>
<td>Display name for the configuration record.</td>
</tr>
<tr>
<td>Update name</td>
<td>Unique identifier for the configuration record. This value is used to identify versions and updates of the record.</td>
</tr>
<tr>
<td>Class</td>
<td>Table that contains the configuration record.</td>
</tr>
<tr>
<td>Application</td>
<td>Application that contains the configuration record.</td>
</tr>
<tr>
<td>Protection policy</td>
<td>Policy that determines if the configuration record is protected from changes. See Protected Application Files.</td>
</tr>
<tr>
<td>Created</td>
<td>Creation date of the configuration record.</td>
</tr>
<tr>
<td>Created by</td>
<td>User who created the configuration record.</td>
</tr>
<tr>
<td>Updated</td>
<td>Last update date for the configuration record.</td>
</tr>
<tr>
<td>Updated by</td>
<td>User who last updated the configuration record.</td>
</tr>
<tr>
<td>Related Record</td>
<td>Version records for the related configuration record. A version record is created every time a user changes the related record. Use this list to compare versions of the configuration record or to revert to a previous version. See Versions.</td>
</tr>
<tr>
<td>Versions</td>
<td></td>
</tr>
<tr>
<td>Related Record</td>
<td>Local update records for the related configuration record. An update record is created for the most recent change to the related record in a given Update Set. See Update Sets.</td>
</tr>
<tr>
<td>Updates</td>
<td></td>
</tr>
</tbody>
</table>

⚠️ Note: You may find it useful to manually add an **Owned By** (owned_by) field to this form. It indicates, for each application, who in IT owns the application and is responsible for maintaining information about it. The Data Certification function uses this field to send certification tasks every quarter, or at any time interval that is configured. You can assign certification tasks to other users, but using this field limits configuration effort. To learn more about adding a field to a form, and about Data Certification, see [Using the form designer](#) and [Data Certification](#).

### Application file protection policy

A read-only protection policy prevents anyone from modifying an application file or its related record.
Some application code shipped with the ServiceNow system requires special protection. Only a ServiceNow employee can alter the protection policy and then modify the application file or its related record. A ServiceNow employee cannot edit protected files without changing the policy designation first.

To prevent customizations from being overwritten by system upgrades, the upgrade process automatically skips changes to customer-updated records. If you modify an application file or related record that is later designated as Read-only in an upgrade, the application file maintains the default protection policy of Write. You keep the existing modifications and can continue modifying the records.

⚠️ **Note:** Reverting a customized file to its baseline state causes the record to inherit the new protection policy as well. For example, a record going from a Write protection policy to a Read-only protection policy.

### Relationships between configuration records

The Application File Types table defines parent-child relationships between configuration records.

The system uses this structure to keep configuration records that normally belong together in the same application.

⚠️ **Note:** Do not modify the Application File Types table as it provides system functionality.

For example, consider the parent-child relationships for a UI policy.

- The UI policy is a child of the application table.
- UI policy actions are children of the UI policy.
- UI policy actions have a parent UI policy and a grandparent application table.
- The UI policy actions and the UI policy are all descendants of the application table.
Fix scripts

A fix script is server-side JavaScript code that you run after an application is installed or upgraded.

Include fix scripts to make changes that are necessary for the data integrity or product stability of an application.

Administrators can create, manage, and run fix scripts. Users with the script_fix_admin role can create and manage fix scripts but cannot run fix scripts.

Note: An annotation with the following message shows up when you open an existing fix script or create a new fix script.

Any customizations you make to the fix script will apply only when you manually run the script. Instance upgrades use the out of box fix script.

Create a fix script

Create fix scripts to ensure the system installs or updates an application properly.

Before you begin
Role required: script_fix_admin or admin
About this task
Use fix scripts to add, update, and delete data, including rules, scripts, and property settings.

Procedure
1. Navigate to **System Definition > Fix Scripts**.
2. Click **New**.
3. Define the fix script by completing the fields on the form.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique, descriptive name for the fix script.</td>
</tr>
<tr>
<td>Unloadable</td>
<td>Select the check box to create Customer Update [sys_update_xml] records when the fix script runs. Clear the check box to run upgrades without creating these records (default). Fix script tests enforce the Unloadable option.</td>
</tr>
<tr>
<td>Before</td>
<td>Select the check box to run the fix script before installing or upgrading the application. Clear the check box to run the fix script after (default).</td>
</tr>
<tr>
<td>Description</td>
<td>Enter a description of the fix script.</td>
</tr>
<tr>
<td>Script</td>
<td>Enter the code for the fix script.</td>
</tr>
</tbody>
</table>

4. Click **Submit**.
5. Test the fix script and make any necessary changes.

Test a fix script
Test your fix scripts to ensure they install or update applications as expected.

Before you begin
Role required: admin

About this task
Fix scripts add, update, and delete data, including rules, scripts, and property settings.

ℹ️ Note: Do not test fix scripts on production instances.
Procedure

1. Open the fix script record.
2. Review the code design to ensure that it can run more than once on the same system without causing damage. For example, you may write a fix script that adds a role to a property by default. Design the script so that it can run multiple times on the same system without overwriting the existing data, even if it is not necessary to run the script again after the initial installation.
3. Click the Run Fix Script related link.
   Tests enforce the Unloadable option.
4. Confirm how to run the script.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proceed in Background</td>
<td>Use this option for long-running scripts, or if you do not know the expected execution time.</td>
</tr>
<tr>
<td>Proceed</td>
<td>Use this option to run the script immediately and display the results in a confirmation window.</td>
</tr>
</tbody>
</table>

5. Review the results from the Progress Workers related list, and make any necessary changes. To manually stop a running fix script:
   a. From the Progress Workers related list, select a worker in the Running State.
   b. Select the Cancel job related link.
Run fix scripts

After you transfer an application to another instance, you must manually run any necessary fix scripts.

Before you begin
Role required: admin

About this task
To run a fix script:

Procedure
1. Navigate to System Definition > Fix Scripts.
2. Edit the filter to search for your application name. For example, Application | is | Book Rooms).
3. Open the fix script record.
4. Click Run Fix Script.

Fulfillment tables

To enable a production instance to enforce entitled usage of your ServiceNow Store App, you configure the tables where only record owners or subscribed app users can make updates.

For any table that you, the developer, create or extend to support a custom application, you can specify that the table is a fulfillment table. In a fulfillment table, only a subscribed fulfiller user can perform a fulfiller action (typically, create/update/delete a not-own record).

In contrast, for a table that is not a fulfillment table, any user—e ven a user who is not subscribed—can act as a requester. The intent is to allow the usage admin to enable subscription enforcement on any production instance that implements the application.

Ownership of records in a fulfillment table

To enable the system to identify a fulfiller action, you define how to determine ownership of any record in the table. The developer of the application specifies the set of conditions that determine whether a user owns the record. For example, UserA owns a record in a task table if UserA opened the record or another resource opened the record on behalf of UserA.

For task-extended tables, time cards, and apps that require a subscription, the system sets the table as a fulfillment table by default and auto-assigns the ownership condition. For tables that you create to support your app,
you can mark the table as a fulfillment table and can specify the ownership condition (for example, use the filter `opened_by[[is][currentUser] OR [caller_id][is][currentUser]`).

System default conditions for ownership

<table>
<thead>
<tr>
<th>Action</th>
<th>Ownership condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>task extension</td>
<td>opened_by (read-only)</td>
</tr>
<tr>
<td>catalog request</td>
<td>requested_for (read-only)</td>
</tr>
<tr>
<td>other tables in apps that require a subscription</td>
<td>sys_created_by (read-only)</td>
</tr>
<tr>
<td>tables created by developer for app that requires a subscription</td>
<td>Specified by developer</td>
</tr>
</tbody>
</table>

Specify that a table is a fulfillment table

You configure a table as a fulfillment table to enable the system to prevent updates by users who are not subscribed to the app. For ServiceNow Store apps, you configure a table as a fulfillment table to enforce that fulfillment usage complies with your subscription use policy.

Before you begin
Role required: usage_admin, admin

Procedure

1. While working on a table, open the Subscription Management related list.
2. Select Fulfillment table.
3. Specify how the system determines ownership of records in the table so that both end users who own a record and subscribed fulfiller users can update the record: Specify the Ownership condition. For example, set the filter as `[opened_by][is][currentUser] OR [caller_id][is][currentUser]`.

Contextual development environment

The platform is a contextual development environment that displays the currently selected application, identifies the scope of every application artifact, and prevents any changes that violate the access settings for an application.
Application developers can use the contextual development environment to:

- Determine the application context
- View and select applications
- Enforce application version standards
- Enforce application resource throttling
- Enforce script protection policies

**Application context**

When application developers create new records, the system automatically assigns the records to the currently selected application in the application picker.

When application developers attempt to change existing records, the platform checks whether the currently selected application matches the scope of the application artifact. If they match, the application developer can save changes to the artifact. If they differ, the system makes the following changes to the user interface:

- Makes all the fields on the current record read-only.
- Displays a warning message that the application artifact belongs to another scope.

**Application access settings**

Application access settings determine whether one application can access resources from another application.

Application access settings are similar to access controls (ACLs) in that they allow you to restrict access to certain resources, but instead of restricting tables and records from users they restrict applications resources from other applications. There are several ways to set cross-scope access.

<table>
<thead>
<tr>
<th>Setting type</th>
<th>Use</th>
<th>Description</th>
<th>Access setting location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application design and runtime</td>
<td>Your application or application script requires access to a</td>
<td>Application designers can configure the following application access settings for the entire application.</td>
<td>Access determined by the cross-scope privilege record owned</td>
</tr>
<tr>
<td>Setting type</td>
<td>Use</td>
<td>Description</td>
<td>Access setting location</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td><strong>cross-scope resource.</strong></td>
<td></td>
<td>• Specify whether cross-scope tables can be selected during design-time activities. • Specify when scripts can run on cross-scope resources. • Specify what JavaScript standard the application supports.</td>
<td>by the calling application.</td>
</tr>
<tr>
<td><strong>Table design and runtime settings</strong></td>
<td>A cross-scope application or web service requires access to perform CRUD operations on a table.</td>
<td>Application developers can also configure application access settings for individual tables. • Specify whether the table is available to other application scopes. • Specify what runtime operations from other application scopes the table supports. • Specify whether the table can be selected during design-time activities. • Specify whether the table can be accessed from web services.</td>
<td>Access determined by settings on the target table.</td>
</tr>
<tr>
<td><strong>Restricted caller access privilege settings</strong></td>
<td>A cross-scope application or script requires access to your application or application resource.</td>
<td>Admin users can configure the following application access settings for an entire application scope or application resource. • Specify whether a cross-scope script can access your application scope or an application resource.</td>
<td>Access determined by the restricted caller access record owned by the target application.</td>
</tr>
</tbody>
</table>
### Setting type

<table>
<thead>
<tr>
<th>Use</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Track cross-scope requests for access to application resources.</td>
</tr>
<tr>
<td></td>
<td>• Approve or deny cross-scope requests to access application resources.</td>
</tr>
<tr>
<td>Access setting location</td>
<td></td>
</tr>
</tbody>
</table>

### Application design and runtime settings

The application design and runtime settings determine whether an application can access cross-scope resources.

### Design and Runtime fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JavaScript Mode</td>
<td>The application’s supported JavaScript standard. Select <strong>ES5 Standards Mode</strong> to support features in ECMAScript 5th edition. Select <strong>Compatibility Mode</strong> to support earlier ECMAScript editions.</td>
</tr>
<tr>
<td>Runtime Access Tracking</td>
<td>The application’s handling of script access requests to resources in other applications. Select <strong>None</strong> to authorize all access requests to cross-scope resources without logging them. Select <strong>Tracking</strong> to log and authorize all access requests to cross-scope resources. Select <strong>Enforcing</strong> to log access requests to cross-scope resources but require an administrator to authorize each request.</td>
</tr>
<tr>
<td>Restrict Table Choices</td>
<td>The availability of cross-scope tables when designing the application. Clear the option to allow the application to see tables from other application scopes. Select the option to restrict design choices to only tables in the same application.</td>
</tr>
</tbody>
</table>

### Runtime access tracking

Runtime access tracking allows administrators to manage script access to application resources by creating a list of script operations and targets that the system authorizes to run.

Runtime access tracking provides the following options:
## Runtime access tracking options

<table>
<thead>
<tr>
<th>Option</th>
<th>Tracking done</th>
<th>Authorization done</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>The system does not track runtime access requests.</td>
<td>The system does not require authorization to run access requests.</td>
<td>Application scripts can access resources from other applications as long as the table-level access settings allow it.</td>
</tr>
<tr>
<td>Tracking</td>
<td>During development, the system creates a Cross-Scope Privilege record for each runtime access request. After installation, the system no longer tracks new runtime access requests.</td>
<td>The system sets the status of the Cross-Scope Privilege record to <strong>Allowed</strong>.</td>
<td>The system runs the tracked operation as long as the table-level access settings allow it.</td>
</tr>
<tr>
<td>Enforcing</td>
<td>During development, the system creates a Cross-Scope Privilege record for each runtime access request. After installation, the system no longer tracks new runtime access requests.</td>
<td>The system sets the status of the Cross-Scope Privilege record to <strong>Requested</strong>.</td>
<td>The system blocks the tracked operation from running until an Administrator manually changes the status to <strong>Allowed</strong> and the table-level access settings allow it.</td>
</tr>
</tbody>
</table>

During development, application designers must run all of an application’s script logic to ensure the system tracks and authorizes the access requests to other applications.
Cross-scope privilege record

Runtime access tracking uses cross-scope privilege records to determine which script operations and targets the system allows to run.

The system creates cross-scope privilege records when:

• Runtime access tracking is set to **Tracking** or **Enforcing**.
• A script attempts to access another application.

Each cross-scope privilege record contains the following information.

**Cross-scope privilege fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Scope</td>
<td>The application requesting runtime access to another application's resources.</td>
</tr>
<tr>
<td>Target Scope</td>
<td>The application whose resources are being requested.</td>
</tr>
<tr>
<td>Target Name</td>
<td>The name of the table, script include, or script object being requested.</td>
</tr>
<tr>
<td>Target Type</td>
<td>The type of request: table, script include, or script object.</td>
</tr>
<tr>
<td>Operation</td>
<td>The operation the script performs on the target. The target type determines the available operations. Tables support the read, write, create, and delete operations. Script includes and script objects only support the execute API operation.</td>
</tr>
<tr>
<td>Status</td>
<td>The authorization for this record: requested, allowed, or denied</td>
</tr>
</tbody>
</table>

Administrators can manually create cross-scope privilege records for application developers in advance to communicate which cross-scope resources they expect developers to access. For example, administrators could create these cross-scope privilege records to permit application developers access to resources from Incident Management.

**Sample cross-scope privilege records**

<table>
<thead>
<tr>
<th>Source Scope</th>
<th>Target Scope</th>
<th>Target Name</th>
<th>Operation</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>My App</td>
<td>Global</td>
<td>incident</td>
<td>Read</td>
<td>Allowed</td>
</tr>
<tr>
<td>My App</td>
<td>Global</td>
<td>incident</td>
<td>Write</td>
<td>Allowed</td>
</tr>
</tbody>
</table>
Sample cross-scope privilege records (continued)

<table>
<thead>
<tr>
<th>Source Scope</th>
<th>Target Scope</th>
<th>Target Name</th>
<th>Operation</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>My App</td>
<td>Global</td>
<td>ScopedGlideRecord</td>
<td>Execute API</td>
<td>Allowed</td>
</tr>
</tbody>
</table>

During testing, application developers should run all of their application scripting logic to ensure the system creates any necessary cross-scope privilege records. After application publication, the system only allows runtime requests to run that have a valid cross-scope privilege record.

**Note:** Table privilege granting is limited to, at most, the permissions set on the table object (sys_db_object) record. For example, granting a scope privilege to delete for table incident would not be allowed if the table object for incident did not allow Can delete scopes.

Application design access record

Administrators use application design access records to specify which other applications are available to developers during application creation.

When administrators restrict an application’s table choices, they can then create application design access records to grant developers access to the tables of selected applications.

Each application design access record contains the following information.

**Application design access fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Scope</td>
<td>The application requesting design access to another application’s tables.</td>
</tr>
<tr>
<td>Target Package</td>
<td>The application whose tables will be available for design time access.</td>
</tr>
<tr>
<td>Application</td>
<td>The application to which this record belongs.</td>
</tr>
</tbody>
</table>

Application design access records allow administrators to have complete control of the resources available to application developers. When creating configuration records, application developers can only select tables from another application if there is an application design access record granting them access to the application. For example, administrators could create the following application design access records to grant developers access to tables from Incident Management and Problem Management.
Sample application design access records

<table>
<thead>
<tr>
<th>Source Scope</th>
<th>Target Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>My App</td>
<td>Incident</td>
</tr>
<tr>
<td>My App</td>
<td>Problem Management</td>
</tr>
</tbody>
</table>

After developers create configuration records to other applications, the system displays these applications as dependencies.

**Table design and runtime settings**

The Application access fields determine whether a table is accessible to other applications during design-time or run-time operations.

**Application access fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can read</td>
<td>Select the check box to allow script objects from other application scopes to read records stored in this table. This option offers runtime protection. For example, a script in another application can query data on this table. You must first select read access to grant any other API record operation.</td>
</tr>
<tr>
<td>Can write</td>
<td>Select the check box to allow script objects from other application scopes to modify records stored in this table. This option offers runtime protection. For example, a script in another application can modify a field value on this table. This option is available only when the Can read check box is selected. Clear the check box to prevent script objects from other application scopes from modifying data stored in this table.</td>
</tr>
<tr>
<td>Can create</td>
<td>Select the check box to allow script objects from other application scopes to create records in this table. This option offers runtime protection. For example, a script in another application can insert a new record in this table. This option is available only when the Can read check box is selected. Clear the check box to prevent script objects from other application scopes from creating records in this table.</td>
</tr>
<tr>
<td>Can delete</td>
<td>Select the check box to allow script objects from other application scopes to delete records from this table. This option offers runtime protection. For example, a script in another application can remove</td>
</tr>
</tbody>
</table>
Application access fields (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a record from this table</td>
<td>This option is available only when the <strong>Can read</strong> check box is selected. Clear the check box to prevent script objects from other application scopes from deleting records from this table.</td>
</tr>
<tr>
<td>Allow access to this table</td>
<td>Select the check box to allow users to make inbound web service queries to this table. This option offers both design-time and runtime protection. The user performing the query must have the correct permissions to access this table, even when this check box is selected. Clear the check box to prevent users from making web service queries to this table.</td>
</tr>
<tr>
<td>via web services</td>
<td></td>
</tr>
</tbody>
</table>

Runtime access to applications tables

Runtime access determines if an API or web service call can run against an application table.

Access permissions can be set for the following access points.

Runtime access points

<table>
<thead>
<tr>
<th>Access points</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Script API</td>
<td>Any supported object or method call from the scoped system API such as a GlideRecord call.</td>
</tr>
<tr>
<td>Web services</td>
<td>Any supported web service call such as a REST, JSON, or SOAP web service.</td>
</tr>
</tbody>
</table>

The system does not prevent you from creating API or web service calls to the application tables, rather it determines if the API or web service call is allowed to run against the application table. API or web service calls that violate the access permissions for an application table produce an error. For example, making a web service call to a protected application table produces a 403 Forbidden HTTP error.

Default runtime access permissions

The default runtime access permissions apply to new application data tables.
By default, new application tables only allow read access from other application scopes.

### Default runtime access

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessible from</td>
<td>All application scopes</td>
</tr>
<tr>
<td>Can read</td>
<td>Enabled</td>
</tr>
<tr>
<td>Can create</td>
<td>Disabled</td>
</tr>
<tr>
<td>Can update</td>
<td>Disabled</td>
</tr>
<tr>
<td>Can delete</td>
<td>Disabled</td>
</tr>
<tr>
<td>Allow access to this table via web services</td>
<td>Enabled</td>
</tr>
</tbody>
</table>

### Application access permissions for a table record

**Procedure**

1. Navigate to **System Applications > Applications**.
2. Click the button for the application type you want to edit.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developed</td>
<td>displays applications created on this instance.</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>Downloaded</td>
<td>displays applications downloaded on this instance.</td>
</tr>
</tbody>
</table>

3. Click the application name or the **Edit** button for the application you want to work on.

4. From the **Tables** related list, select the table whose access permission you want to set.

5. From the **Application Access** section, set the runtime access permissions.

6. Click **Update**.

**Example denying all runtime access to a table**

You can prevent script API and web service calls from other application scopes. Typically, this is to prevent any other application from creating or modifying data in the table. Denying access requires setting the following value in the table record.

<table>
<thead>
<tr>
<th>Denying all runtime access</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Field</strong></td>
</tr>
<tr>
<td>Accessible from</td>
</tr>
<tr>
<td>Can read</td>
</tr>
<tr>
<td>Can create</td>
</tr>
<tr>
<td>Can update</td>
</tr>
<tr>
<td>Can delete</td>
</tr>
<tr>
<td>Allow access to this table via web services</td>
</tr>
</tbody>
</table>
Limiting runtime access to this application scope only

The following diagram illustrates the effect of denying other application scopes access to application tables from script API and web service calls.

Deny all runtime access permissions to application tables

Example granting all runtime access to a table
You can permit some or all runtime script API and web service calls from other application scopes.
Granting access requires setting the following values in the table record.
Granting all runtime access

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessible from</td>
<td>All application scopes</td>
</tr>
<tr>
<td>Can read</td>
<td>Enabled</td>
</tr>
<tr>
<td>Can create</td>
<td>Enabled</td>
</tr>
<tr>
<td>Can update</td>
<td>Enabled</td>
</tr>
<tr>
<td>Can delete</td>
<td>Enabled</td>
</tr>
<tr>
<td>Allow access to this table via web services</td>
<td>Enabled</td>
</tr>
</tbody>
</table>

Granting other application scopes all runtime access permissions

The following diagram illustrates the effect of granting access to application tables from API calls and web services in other application scopes.
Granted access to application tables

As the application developer, you can grant or deny other applications the permission to create configuration records, also known as application files, that extend the functionality of an application.

The permission applies to any platform feature that extends the functionality of an application data table such as:

- Business rules
- UI actions
- Client scripts

These access permissions protect the application data table at design-time. The system prevents you from creating configuration records by hiding the application data table as an option in the **Table** field. For example, a protected application table does not appear as an option when you create configuration records such as UI actions and client scripts.

Even when permission is granted to create configuration records, some configuration records have additional restrictions to protect application data from unwanted changes from other application scopes.
Default design access permissions

By default, new application tables prevent other application scopes from creating configuration records on application data tables. This prevents any other applications from changing the functionality of a table.

Default design-time access

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessible from</td>
<td>All application scopes</td>
</tr>
<tr>
<td>Can read</td>
<td>Enabled</td>
</tr>
<tr>
<td>Can create</td>
<td>Disabled</td>
</tr>
<tr>
<td>Can update</td>
<td>Disabled</td>
</tr>
<tr>
<td>Can delete</td>
<td>Disabled</td>
</tr>
<tr>
<td>Allow access to this table via web services</td>
<td>Enabled</td>
</tr>
<tr>
<td>Allow configuration</td>
<td>Disabled</td>
</tr>
</tbody>
</table>

Default access permissions to configuration records

Set design-time access to application tables

Set these access permissions to protect application tables at design-time.

About this task

To set runtime access permissions:
Procedure

1. Navigate to **System Applications > My Company Applications**.
2. Click the button for the application type you want to edit.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developed</td>
<td>displays applications created on this instance.</td>
</tr>
<tr>
<td>Downloaded</td>
<td>displays applications downloaded on this instance.</td>
</tr>
</tbody>
</table>

3. Click the application name or the **Edit** button for the application you want to work on.
4. From the Tables related list, select the table whose access permission you want to set.
5. From the Application Access section, set the design-time access permissions.
6. Click **Update**.

Related information
- Example denying all design access to a table
- Example allowing configuration records for a table

**Example denying all design access to a table**

You can prevent other application scopes from creating configuration records on application data tables.

Typically, this is to prevent any other applications from changing the functionality of a table. Denying access requires setting the following value in the table record.

Denying all design-time access

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessible from</td>
<td>This application scope only</td>
</tr>
<tr>
<td>Can read</td>
<td>Disabled</td>
</tr>
<tr>
<td>Can create</td>
<td>Disabled</td>
</tr>
<tr>
<td>Can update</td>
<td>Disabled</td>
</tr>
</tbody>
</table>
Denying all design-time access (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can delete</td>
<td>Disabled</td>
</tr>
<tr>
<td>Allow access to this table via web services</td>
<td>Disabled</td>
</tr>
<tr>
<td>Allow configuration</td>
<td>Disabled</td>
</tr>
</tbody>
</table>

Limiting design-time access to this application scope only

The following diagram illustrates the effect of denying other application scopes the ability to create configuration records.

Example allowing configuration records for a table

You can permit other application scopes to create configuration records on application data tables.
You can grant access to the following configuration records with these settings.

### Granting access to configuration records

<table>
<thead>
<tr>
<th>Configuration record</th>
<th>Setting required to grant access</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access controls</td>
<td>• Accessible from set to All application scopes</td>
</tr>
<tr>
<td></td>
<td>• Can read is selected</td>
</tr>
<tr>
<td>Business rules</td>
<td></td>
</tr>
<tr>
<td>Client scripts</td>
<td>• Accessible from set to All application scopes</td>
</tr>
<tr>
<td>Dictionary entry (new field only)</td>
<td>• Can read is selected</td>
</tr>
<tr>
<td>UI actions</td>
<td>• Allow configuration is selected</td>
</tr>
</tbody>
</table>

### Granting other application scopes design access permission

The following diagram illustrates the effect of granting other application scopes the ability to create configuration records.
Granting access to configuration records

Restricted caller access privilege settings

By using restricted caller access privilege settings, you can define cross-scope access to an application, application resource (such as an access control role, a business rule, a UI action, or a script include), or event. You can even use these settings to allow or deny requests for access. Restricted caller access [sys_restricted_caller_access] records track cross-scope applications or scripts that request access to an application, application resource, or event in the Now Platform.

Restricted caller access privilege settings overview

The Now Platform creates sys_restricted_caller_access records when one of these actions occurs:

- Caller access is set to **Caller Restriction** or **Caller Tracking**.
- A cross-scope script attempts to access an application resource or event.

**Note:** A system scope to target scope is an example of a cross-scope.

By using these records, you can do these tasks:

- Track cross-scope requests for access to an application resource.
- Approve or deny any cross-scope requests for access to application resources or events. For example, you can create a Restricted Caller Access record to allow access for all scope-to-scope requests.

For more information, see: Requested RCA feature
Restricted caller access privilege setting combinations
You can define various combinations of privilege settings for restricted caller access and specify whether access is allowed or restricted for each relationship. This process ensures that your application has the privileges to access the correct scope and not one you don’t want to access. You can define various combinations of the following settings:

Scope
All application resources in a selected source or target scope. To learn more about application scopes, see Application scope.

Source
A specific application resource in a selected source scope.

Target
A specific application resource in a selected target scope.

These restricted caller access privilege settings combinations include, but are not limited to following combinations:

- Scope-to-scope
- Scope-to-target
- Source-to-scope
- Source-to-target

Note: For more information about these access setting combinations and to learn how to create each combination, see Set the application scope, application resource, and event access.

Activating application restricted caller access
You can activate application restricted caller access through one of the following methods:

- Activate the Scoped Application Restricted Caller Access plugin (com.glide.scope.access.restricted_caller).
- Request the HR Service Delivery or Security Incident Response applications. By default, restricted caller access is active in these applications.

For more information, see: Activate RCA.

Activate application restricted caller access
You can activate the Scoped Application Restricted Caller Access plugin (com.glide.scope.access.restricted_caller) if you have the admin role.
Before you begin
Role required: admin

Procedure

1. Navigate to System Applications > All Available Applications > All.

2. Find the plugin using the filter criteria and search bar.

   You can search for the plugin by its name or ID. If you cannot find a plugin, you might have to request it from ServiceNow personnel. For more information, see Request a plugin.

3. Click Install, and then in the Activate Plugin dialog box, click Activate.

   Note: When domain separation and delegated admin are enabled in an instance, the administrative user must be in the global domain. Otherwise, the following error appears: Application installation is unavailable because another operation is running: Plugin Activation for <plugin name>.

Related information

List of plugins (madrid)

Define cross-scope access to an application resource
Track cross-scope requests for access to an application resource and approve or deny requests.

Before you begin
If you enable application administration for the target application, only application administrators of the target application can set access to an application. If application administration is not enabled, an admin user can set access to an application.
Role required: admin or application admin

Note: To learn about application-specific administrator roles and delegated development, see Access control rules in application administration apps and Delegated development and deployment.

Procedure

1. To define access to an application resource, navigate to the application resource record. Available application resources include the following:
   - Table
   - Script Include
2. Set the **Accessible from** field to **All application scopes**. If set to **This application scope only**, no other application scopes can access the resource.

3. Select the appropriate access level in the **Caller Access** field.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Cross-scope calls to the resource are approved or denied based on the value of the <strong>Accessible from</strong> field.</td>
</tr>
<tr>
<td>Caller Restriction</td>
<td>Calls to the resource must be manually approved. Access requests are tracked in the Restricted Caller Access table with a status of Requested.</td>
</tr>
<tr>
<td>Caller Tracking</td>
<td>Calls to the resource are automatically approved. Calls are tracked in the Restricted Caller Access table with a status of Allowed.</td>
</tr>
</tbody>
</table>

4. Allow or deny an access request from a calling application. If a cross-scope application attempts to access a resource set to Caller Restriction, a record is created in the Restricted Caller Access table with a Requested status. An admin user or application administrator must allow or deny the request.

If a calling resource changes (such as when a business rule’s script changes), the restricted caller access record status changes to Invalidated. An admin user or application administrator must update the status to Allowed or Denied.

a. In the application record, navigate to the **Restricted Caller Access Privileges** tab.

b. In the **Status** column, set the value from **Requested** to **Allowed** or **Denied**. Once a calling script is allowed, all subsequent calls are allowed.

**Set the application scope, application resource, and event access**

Create a record in the Restricted Caller Access Privileges [sys_restricted_caller_access] table to set cross-scope resource access requests. Approve or deny requests from a source scope or source scope application resources to a target scope or to target scope application resources.

**Before you begin**

If you enable application administration for the target application, only application administrators of the target application can set access to an application. If application administration is not enabled, an admin user can set access to an application.
Role required: application admin or admin

ℹ️ Note: To learn about application-specific administrator roles and delegated development, see and .

About this task
You can set the following restricted caller access privilege settings combinations:
• Scope-to-Scope
• Scope-to-Target
• Source-to-Scope
• Source-to-Target

ℹ️ Note: In the Rome release, we have enforced that an RCA privilege record must be present in the target application to grant access to a resource. This means that the target scope must match the application scope.

Procedure
1. Navigate to System Applications > Application Restricted Caller Access.
2. On the form, fill in the fields.

Restricted Caller Access fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation</td>
<td>Operation on the target resource.</td>
</tr>
<tr>
<td>Source</td>
<td>Record of the calling script.</td>
</tr>
<tr>
<td>Source Scope</td>
<td>Scope of the calling application.</td>
</tr>
<tr>
<td>Source Table</td>
<td>Table that contains the Source record.</td>
</tr>
<tr>
<td>Source Type</td>
<td>Type of record that is calling the application resource:</td>
</tr>
<tr>
<td></td>
<td>• ACL</td>
</tr>
<tr>
<td></td>
<td>• Business Rule</td>
</tr>
<tr>
<td></td>
<td>• Document Title</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>• GlideScopedEvaluator</td>
<td></td>
</tr>
<tr>
<td>• Inbound Email Script</td>
<td></td>
</tr>
<tr>
<td>• Orchestration RunScript Activity</td>
<td></td>
</tr>
<tr>
<td>• Service Portal Widget</td>
<td></td>
</tr>
<tr>
<td>• Scheduled Script</td>
<td></td>
</tr>
<tr>
<td>• Scope</td>
<td></td>
</tr>
<tr>
<td>• Script Include</td>
<td></td>
</tr>
<tr>
<td>• UI Action</td>
<td></td>
</tr>
<tr>
<td>• UI Macro</td>
<td></td>
</tr>
<tr>
<td>• UI Page</td>
<td></td>
</tr>
<tr>
<td>• Workflow Activity</td>
<td></td>
</tr>
</tbody>
</table>

For example, to allow access from an entire application, select **Scope**.

<table>
<thead>
<tr>
<th>Status</th>
<th>Status of the access request:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Requested</td>
<td></td>
</tr>
<tr>
<td>• Denied</td>
<td></td>
</tr>
<tr>
<td>• Allowed</td>
<td></td>
</tr>
<tr>
<td>• Invalidated</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** If a calling resource changes, the restricted caller access record status changes to **Invalidated**. If you enable application administration, only application administrators of the target application can update the status of a request.

<table>
<thead>
<tr>
<th>Target</th>
<th>Record of the requested resource.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Scope</td>
<td>Scope of the requested resource.</td>
</tr>
<tr>
<td>Target Table</td>
<td>Table that contains the Target record.</td>
</tr>
<tr>
<td>Target Type</td>
<td>Type of requested resource.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Event         | • **Note:** An event is a special type of target for restricted caller access. By selecting an event in a target scope, you give a source application permission to queue an event that is registered as part of a target application. However, if you set the caller access on the event registry to None, it prevents cross-scope access calls to an event. This setting combination is a one-to-one relationship. To learn more about events, and their function, see Events.  
  
  • **Note:** If you set caller access to None on the event registry, the cross-scope access calls to an event are denied.  
  
  • Scope  
  • Table  
  • Script Include  
  
  For example, to allow access to an entire application, select **Scope**. |

**Note:** Refer to the following sections for instructions on defining each type of restricted caller access privilege setting combination.

**Scope-to-scope settings**

Allow or deny access of all application resources in a source scope to all application resources in a target scope. This setting combination is a many-to-many relationship.

Enter the following field settings for Scope-to-Scope restricted caller access.
<table>
<thead>
<tr>
<th>Field</th>
<th>Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Scope</td>
<td>Scope of the calling application that contains the source application resources.</td>
</tr>
<tr>
<td>Source Type</td>
<td>Type of record calling the application resource. To allow access from an entire application, select Scope.</td>
</tr>
<tr>
<td>Status</td>
<td>Status of the access request. Select Allowed to allow access, or select Denied to restrict access for this source-target resource relationship.</td>
</tr>
<tr>
<td>Target Scope</td>
<td>Scope of the requested resource that contains the target application resources that the source application resource requests access to.</td>
</tr>
<tr>
<td>Target Type</td>
<td>Type of requested resource. Select Scope to include all application resources in the target scope.</td>
</tr>
</tbody>
</table>

**Scope-to-target settings**

Allow or deny access of all application resources in a source scope to a specific application resource (business rule, table, script include, or event) in a target scope.

This setting combination is a many-to-one relationship. For example, you can specify that all application resources in source Scope A can access a script include in target Scope B.

Enter the following field settings for Scope-to-Target restricted caller access.

<table>
<thead>
<tr>
<th>Field</th>
<th>Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Scope</td>
<td>Scope of the calling application that contains the source application resources.</td>
</tr>
<tr>
<td>Field</td>
<td>Entries</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Source Type</td>
<td>Type of record calling the application resource. Select <strong>Scope</strong> to include all application resources in the source scope.</td>
</tr>
<tr>
<td>Status</td>
<td>Status of the access request. Select <strong>Allowed</strong> to allow access, or select <strong>Denied</strong> to restrict access for this source-target resource relationship.</td>
</tr>
<tr>
<td>Target Scope</td>
<td>Scope of the requested resource that contains the target application resources that the source application resource requests access to.</td>
</tr>
<tr>
<td>Target Type</td>
<td>Type of requested resource. Select the specific application resource (for example, business rule, script include, UI page, event) the source application resource requests access to.</td>
</tr>
<tr>
<td>Operation</td>
<td>Type of operation (for example, Read, Write) in the target application resource the source application resource requests access to.</td>
</tr>
</tbody>
</table>

**Source-to-scope settings**

Allow or deny access of a specific application resource in a source scope to all application resources in a target scope.

This setting combination is a one-to-many relationship. For example, you can specify that a particular business rule in source Scope A can access all application resources in target Scope B.

Enter the following field settings for Source-to-Scope restricted caller access.
<table>
<thead>
<tr>
<th>Field</th>
<th>Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Scope</td>
<td>Scope of the calling application that contains the source application resource requesting access to the target scope application resource.</td>
</tr>
<tr>
<td>Source Type</td>
<td>Type of record calling the application resource. Select the specific application resource (for example, business rule or script include) requesting access to the specified target scope application resource.</td>
</tr>
<tr>
<td>Status</td>
<td>Status of the access request. Select <strong>Allowed</strong> to allow access, or select <strong>Denied</strong> to restrict access for this source-target resource relationship.</td>
</tr>
<tr>
<td>Target Scope</td>
<td>Scope of the requested resource that contains the target application resources that the source application resource requests access to.</td>
</tr>
<tr>
<td>Target Type</td>
<td>Type of requested resource. Select <strong>Target</strong> to include all application resources in the selected target scope.</td>
</tr>
<tr>
<td>Operation</td>
<td>Type of operation (for example, Read, Write) in the target application resource the source application resource requests access to.</td>
</tr>
</tbody>
</table>

**Source-to-target settings**

Allow or deny access of a specific application resource in a source scope to a specific application resource in a target scope.

This setting combination is a one-to-one relationship. For example, you can specify that a specific business rule in source Scope A can access a specific application resource, such as a business rule, table, script include or event, in a target scope.
Enter the following field settings for Source-to-Target restricted caller access.

<table>
<thead>
<tr>
<th>Field</th>
<th>Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Scope</td>
<td>Scope of the calling application that contains the source application resources.</td>
</tr>
<tr>
<td>Source Type</td>
<td>Type of record calling the application resource. Select the specific application resource (for example, business rule, script include, or UI page) requesting access to the specified target scope application resource.</td>
</tr>
<tr>
<td>Status</td>
<td>Status of the access request. Select <strong>Allowed</strong> to allow access, or select <strong>Denied</strong> to restrict access for this source-target resource relationship.</td>
</tr>
<tr>
<td>Target Scope</td>
<td>Scope of the requested resource that contains the target application resources that the source application resource requests access to.</td>
</tr>
<tr>
<td>Target Type</td>
<td>Type of requested resource. Select the specific application resource (for example, business rule, Script Include, event) the source application resource requests access to.</td>
</tr>
</tbody>
</table>

**Requested restricted caller access (RCA)**

You can use a requested RCA to grant store apps access to protected resources in the Now Platform without the need to wait for the next family release. If you have the system admin or application admin role, you can review requested RCAs and approve and deny them.
RCAs are classified into two categories:

- Real RCA: sys_scope==target_scope
- Requested RCA: sys_scope!=target_scope

For example: A real RCA record is where the application scope and target scope match. A requested RCA is a record that is still awaiting approval for access to the target scope.

When you install an application, your scheduled jobs generate RCA records with the status of Requested in the target application for each requested RCA record that is packaged in the source application.

Note: The jobs are generated once Upgrade Summary has run.

Example of how a store app accesses a table

Let's say that a store app called HR Integrations Framework wants to access an HR Core Case table. The table is in the business rule called Find Case in the Integration Service table.

To request access, the HR Integrations Framework app requires that an RCA privilege is packaged in its own scope as follows:

- sys_scope = HR Integrations Framework
- target = HR Core Case
- status = Allowed
- target_scope = Human Resource: Core
- source = Find Case

App development example for developers

When you are developing an application, real RCAs are generated with the status of Requested when the target has a caller restriction. If the target has caller tracking, the status becomes Allowed. The developer can review and finalize all the real RCA records that are required for the application to work. For example, those RCAs with a status of Allowed.

A developer can click the Generate RCA Privileges in Current App in the related links to generate requested RCAs that are packaged in the current application. Requested RCAs are synchronized with real RCAs, which means that if a real RCA is updated or deleted, a requested RCA is updated or deleted too.

Now, the HR Integration Framework application can be packaged and installed on a customer instance.
**App installation example for administrators**

When you are installing an app on a customer’s instance, real RCAs are generated in the target application. A real RCA would have the Human Resource: Core with a status of Requested. This process is done asynchronously in a scheduled job, where some lag time can occur.

To notify the target app admin about an RCA’s pending review, messages have been added to application pages. An example is as follows:

---

**Store App backward compatibility**

If a store app is compatible and can be installed on an instance that is pre-Rome, then you must package the RCA records in their own scope with the status of Allowed.

ℹ️ **Note:** This process ensures that the store app works on all versions.

When upgrading to Rome, you can configure a one-time fix script to move RCAs in the source scope to the target scope. In Rome, if the target app already has the necessary RCA records, no RCA records are generated for the RCAs that are packaged by the source app.

---

**Application list**

The applications list allows application developers to view and select applications.

Application developers can use the applications list to open a custom application record. If the contextual development environment detects that you are editing an application artifact in another application scope, it displays a warning message you can use to switch to another application.

Administrators have the following options from the applications list.
### Sample list of applications

#### Application list options

<table>
<thead>
<tr>
<th>Tab name</th>
<th>Options available</th>
<th>Application source</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Develop</strong></td>
<td>• Create new application</td>
<td>Applications developed on this instance</td>
</tr>
<tr>
<td></td>
<td>• Edit existing application</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Share application</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Delete application</td>
<td></td>
</tr>
<tr>
<td><strong>Downloads</strong></td>
<td>• Install</td>
<td>• Applications shared on the company application repository</td>
</tr>
<tr>
<td></td>
<td>• View installed files</td>
<td>• Applications shared on the ServiceNow Store</td>
</tr>
<tr>
<td></td>
<td>• Edit installed files</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Uninstall</td>
<td></td>
</tr>
<tr>
<td><strong>Updates</strong></td>
<td>• Update</td>
<td>• Applications shared on the company application repository</td>
</tr>
<tr>
<td></td>
<td>• View version information</td>
<td>• Applications shared on the ServiceNow Store</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The application list displays the following information.

### Applications list

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developed</td>
<td>Display applications created on this instance.</td>
</tr>
<tr>
<td>Downloaded</td>
<td>Display applications downloaded on this instance.</td>
</tr>
<tr>
<td>Updates</td>
<td>Display available updates for downloaded applications on this instance.</td>
</tr>
</tbody>
</table>

### Application picker

The application picker allows application developers to view and select the application where their changes apply.

The application picker is available from the system menu under the gear icon in the banner frame.

### Lists and forms in scoped applications

The current application context determines what customization and form design options are available when working with lists and forms in scoped applications.

The user interface uses visual indicators to identify a list or form in the same or different application scope.
Available layout and design actions

The system allows the following layout and design actions when working on lists or forms in custom applications.

<table>
<thead>
<tr>
<th>Action</th>
<th>Access granted</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a list view</td>
<td>Always allowed</td>
<td>This action is always available to users with access to customization.</td>
</tr>
<tr>
<td>Create a form view</td>
<td>Always allowed</td>
<td>This action is always available to users with access to customization.</td>
</tr>
<tr>
<td>Create a form section</td>
<td>Always allowed</td>
<td>This action is always available to users with access to customization.</td>
</tr>
<tr>
<td>Select fields to display in a view</td>
<td>Only allowed for sections in the view that match the current scope.</td>
<td>This restriction is independent of a user's role. Administrators cannot bypass this restriction.</td>
</tr>
<tr>
<td>Change the order of sections in a view</td>
<td>Only allowed for views that match the current scope.</td>
<td>This restriction is independent of a user's role. Administrators cannot bypass this restriction.</td>
</tr>
<tr>
<td>Select fields to display in a section</td>
<td>Only allowed for sections that match the current scope.</td>
<td>This restriction is independent of a user's role. Administrators cannot bypass this restriction.</td>
</tr>
<tr>
<td>Add or remove section columns</td>
<td>Only allowed for sections that match the current scope.</td>
<td>This restriction is independent of a user's role. Administrators cannot bypass this restriction.</td>
</tr>
<tr>
<td>Delete a form section</td>
<td>Only allowed for sections that match the current scope.</td>
<td>This restriction is independent of a user's role. Administrators cannot bypass this restriction.</td>
</tr>
<tr>
<td>Create new fields</td>
<td>Only allowed for sections that match the current scope and</td>
<td>This restriction is independent of a user's role. Administrators cannot bypass this restriction.</td>
</tr>
</tbody>
</table>
Available layout and design actions (continued)

<table>
<thead>
<tr>
<th>Action</th>
<th>Access granted</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>when the <strong>Allow configuration</strong> option is enabled.</td>
<td>role. Administrators cannot bypass this restriction.</td>
</tr>
</tbody>
</table>

Form design visual indicators

The UI displays the following visual indicators when designing forms in custom applications.

You can only edit views and sections when you are in the same application scope as the form. Editable sections display:

- Section headings with a solid color background.
- A solid line around the section.
- A control to set the number of columns.
- A **Delete this section** button.
- Grip icons beside section headings.
- Grip icons beside fields.

Views and sections in another application scope display as read only. Read-only sections have:

- Section headings with a gray background.
- A gray line around the section.
- No control to set the number of columns.
• No **Delete this section** button.
• No grip icons beside section headings.
• No grip icons beside fields.

**Visual indicators of read-only sections**

![Visual indicators of read-only sections](image)

**Default form design permissions**

By default, new application data tables have the following form design permissions.

<table>
<thead>
<tr>
<th>Default form design permissions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Form design action</strong></td>
<td><strong>Permission setting</strong></td>
</tr>
<tr>
<td>Create new sections in tables</td>
<td>Allowed</td>
</tr>
<tr>
<td>belonging to another application scope</td>
<td></td>
</tr>
<tr>
<td>Create new fields in sections</td>
<td>By default, denied. Requires application designer to set <strong>Allow configuration</strong> for application table.</td>
</tr>
<tr>
<td>belonging to the same application scope</td>
<td></td>
</tr>
<tr>
<td>Add or remove fields from</td>
<td>Allowed</td>
</tr>
<tr>
<td>sections belonging to the same application scope</td>
<td></td>
</tr>
<tr>
<td>Change the order of fields in sections belonging to the same application scope</td>
<td>Allowed</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Default form design permissions (continued)

<table>
<thead>
<tr>
<th>Form design action</th>
<th>Permission setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change the order of sections belonging to the same application scope</td>
<td>Allowed</td>
</tr>
<tr>
<td>Add or remove fields from sections belonging to another application scope</td>
<td>Denied</td>
</tr>
<tr>
<td>Change the order of fields in sections belonging to another application scope</td>
<td>Denied</td>
</tr>
<tr>
<td>Change the order of sections belonging to another application scope</td>
<td>Denied</td>
</tr>
<tr>
<td>Create new fields in sections belonging to another application scope</td>
<td>Denied</td>
</tr>
</tbody>
</table>

### Layout visual indicators

The UI displays the following visual indicators when configuring the layout of a custom application’s list or form.

You can only edit fields, views, and sections when you are in the same application scope as the form. Editable sections display:

- Field selections with a white background.
- Buttons to add or remove fields.
- Save button with a blue background.
- Editable fields in the **Create new field** section.
Views and sections in another application scope display as read only. Read-only sections have:

- A warning message about the currently selected application scope and the scope of the form.
- Field selections with a gray background.
- No buttons to add or remove fields.
- Save button with a gray background.
- Read only fields in the Create new field section.
Visual indicators of read-only sections

The 'Conference Rooms' section is in the Book Rooms application, but Marketing Events is selected in your application picker. To edit this form:

- Create a new Marketing Events section
- Select an existing Marketing Events section
- Switch to Book Rooms to edit this section

Related reference

Available layout and design actions

Related information

Lists and forms in scoped applications
Form design visual indicators

Contextual development edit messages

The platform displays a message whenever a user attempts to edit an application artifact belonging to a different application.

Application context edit message

This message can be used to:
• Open the application to which the current configuration record belongs.
• Open the application of the currently selected application in the application picker.
• Temporarily switch to the application to which the current configuration record belongs and edit it.

ℹ️ **Note:** The system returns you to the current application context after you save or cancel out of the record.

The system also displays a message when a user attempts to configure a list or form layout while working from another application scope.

**Application context edit message for form layout or design**

The message provides a list of valid options:

• Edit the current section by temporarily switching to the application that owns it.
• Create a new section in the current application context.
• Create a new view in the current application context.

ℹ️ **Note:** The system returns you to the current application context after you save or cancel out of the record.

**Script protection policy**

Application developers can set a protection policy for script includes published as part of a custom application. The policy determines whether someone can view or edit the script include after the application is installed on their instance.

Application developers have these options to protect their custom application script includes:

<table>
<thead>
<tr>
<th>Protection policy</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Allows anyone who installs your published application to view and modify this script include on their instance.</td>
</tr>
</tbody>
</table>
### Script include protection policy Options (continued)

<table>
<thead>
<tr>
<th>Protection policy</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>Select this option to allow other application developers to customize your script include.</td>
</tr>
<tr>
<td>Read-only</td>
<td>Allows anyone who installs your published application to view this script include on their instance. Select this option to allow other application developers to see your script logic, but not to change it.</td>
</tr>
<tr>
<td>Protected</td>
<td>Prevents anyone who installs your published application to modify this script included on their instance. Select this option to prevent other application developers from changing your intellectual property.</td>
</tr>
</tbody>
</table>

### Application administration

Protect sensitive application data by using application administration to restrict how users acquire application-specific roles.

### Functions of application administration

Use application administration to:

- Prevent unauthorized users from accessing sensitive data such as financial records or personally identifiable information.
- Restrict who can assign application-specific roles such as the administrator and designated developers for the application.
- Prevent users with the system-level admin role from:
  - Assigning themselves a protected application role.
  - Assigning themselves to a group containing a protected application role.
  - Bypassing existing access controls to a protected application by creating access controls.
  - Changing the password of users who have a protected application role.
  - Impersonating a user who has a protected application role, unless the developer or administrator also has that role.
  - Inheriting a protected application role.
  - Overriding existing access controls to a protected application.
  - Running scripts that access protected application records.
Roles in application administration
You can make any role an application-specific administrator by selecting the Application Administrator check box in Role Configuration. To learn more, see Restrict access to an application. By convention, create the following roles:

### Application administration roles

<table>
<thead>
<tr>
<th>Role name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application-specific admin</td>
<td>Users with this role can assign other users to an application-specific role for that application. For example, you can create a role named my_application.admin. It should include the name of the restricted application, with a suffix of &quot;admin&quot; to indicate that it is the admin role for the application.</td>
</tr>
<tr>
<td>Application-specific developer</td>
<td>Users with this role can access the restricted application. For example, you can create a role named my_application.developer. It should include the name of the restricted application, with a suffix of &quot;developer&quot; to indicate that it is the developer role for the application. The developer role needs both application administrator and delegated development permissions to modify the application files. To learn more, see Delegated development and deployment and Delegate development and deployment permissions to personnel.</td>
</tr>
</tbody>
</table>

**Application-specific admin role**
The application-specific admin role enables a user to access a specific application, but does not grant the user any other admin rights. Assign the system-level admin role to a user before that user can do these tasks:

- Configure form and list layouts.
- Change application tables and fields.
- Assign the application-specific admin role to new users.
If you do not want a user with the application-specific admin role to have the system-level admin role:

- Do not assign the system-level admin role to the user. Assign only the application-specific admin role.
- Have the user assign themselves the application-specific developer role.

As an application-specific developer, the user can perform a subset of administrative tasks without having the system-level admin role.

⚠️ **Note:** Assign the application-specific admin role to more than one user. Then if a user with the application-specific admin role leaves the company, you are not prevented from changing the application.

### Enabling application administration and assigning application-specific roles

You can enable application administration for an application from the application record and restrict the assignment of application-specific roles from the user role record.

⚠️ **Note:** Enable application administration and assign application-specific roles after completing development of the application, but before adding application records. This practice protects sensitive data in the application records from access by unauthorized users.

The target instance must have at least one authorized user with the application-specific admin role.

- If you enable application administration for an application but do not assign the application-specific roles, no user can access the application.
- If you assign only one application-specific role, you cannot delete that role.

A warning appears if you enable application administration for an application, but no users have the application-specific admin role required to assign roles for the application. The warning reminds you to assign the application-specific roles for administrators and developers of the application.

Set the `[scoped_app_name].min_admin_account` property to require that more than one user must have the application-specific admin role. Assigning the application-specific admin role to multiple users reduces the risk of getting locked out of the scoped application. The `[scoped_app_name].min_admin_account` property has the following limitations:
If you specify an invalid value for the property, the default requirement for assigning at least one application-specific admin is enforced.

If you specify a valid value for the property, you can’t delete any application-specific admins unless you exceed the specified value. For example, if you specify a value of two and you have three application-specific admins, you can delete only one of those roles.

You can specify a value higher than the actual number of assigned application-specific admins. However, you can’t delete any application-specific admins until you exceed the specified value. For example, if you specify a value of six, but have only three application-specific admins, you can’t delete any of those roles.

For procedures, see Restrict access to an application.

Deploying applications with application administration

You must have the system-level admin role in both your developer and production instances to deploy an application protected by application administration. The process is outlined in the following steps.

1. Develop the application on a development instance.
2. Create the application-specific admin role.
3. Grant the application-specific admin role to all system-level admin users.
4. Update the application record to enable application administration and restrict access to the application.
5. Publish the application to the application repository.
6. From a production instance, install the application from the application repository.
7. As a system-level admin on the production instance, grant the application-specific admin role to the appropriate users.
8. Remove the application-specific admin role from all users with the system-level admin role.

For procedures to enable application administration and restrict the assignment of application-specific roles, see Restrict access to an application.

Training

The ServiceNow® Developer Site has training for Securing Applications.
Related information

Delegated development and deployment

Restrict access to an application

Restrict management of an application and access to that application to prevent unauthorized users from assigning administrative rights to the application or accessing sensitive information in the application records.

Before you begin

- Records required:
  - Role record to designate a role as the application-specific admin role
  - User record to assign an application-specific admin role to a user
  - Application record to enable application administration for a specific application
- Role required: admin

If an application-specific admin role does not already exist, create it before beginning this procedure. For example, you can create a role named my_application.admin that includes the name of the restricted application with the suffix "admin" to indicate that it is the admin role for the application.

Procedure

1. Navigate to System Security > Users and Groups > Roles or User Administration > Roles.
2. Open the role record for the application-specific admin role.
3. Configure the form to add the Application Administrator field.
   
   Note: The Application Administrator check box replaces the Assignable by field. By default, when you upgrade from a Jakarta or earlier release to a Kingston or later release, any role that was in the Assignable by field is defined as the application-specific admin role, and the Application Administrator check box is selected.

4. In the role record, select the Application Administrator check box, and then click Update.
6. Open the user record for the admin user.
7. On the Roles tab, add the application-specific role.
Only users with the application-specific admin role can enable application administration for an application.

**Note:** Assign the application-specific admin role to more than one user. Then, if a user with the application-specific admin role leaves the company, you are not prevented from performing changes to the application.

8. Click **Update**.

9. Log out and then log in with the application-specific admin role.

10. Navigate to **System Applications > Applications**.

11. Select the application for which you want to enable application administration.

12. In the application record, select **Application administration**.

13. Click **Update**.
   The system validates that the following requirements have been met:

   - The application has an application-specific admin role (there is at least one role with **Application Administrator** selected).
   - The current user has the application-specific admin role.

   If the validation passes, the system updates the application record. Otherwise, the system displays this error message and does not update the application record:

   Application Administration uses the 'Application Administrator' role to define what users are application administrators. None of the roles defined by this application have 'Application Administrator' enabled.

14. **Optional:** From the Related Links, you can optionally select one of the following options:

<table>
<thead>
<tr>
<th>Related Link</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manage Developers</td>
<td>Modal that enables the application-specific admin to manage these tasks:</td>
</tr>
<tr>
<td></td>
<td>• Designate developers for the application.</td>
</tr>
<tr>
<td></td>
<td>• Make themselves a delegated developer. After the application-specific</td>
</tr>
<tr>
<td></td>
<td>admin becomes a delegated developer, the application-specific admin can</td>
</tr>
<tr>
<td></td>
<td>perform a subset of administration tasks without having the system-level</td>
</tr>
<tr>
<td></td>
<td>admin role.</td>
</tr>
<tr>
<td>Related Link</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Learn more: Delegated development and deployment</td>
<td></td>
</tr>
<tr>
<td>Modal that creates a Contained Role [sys_user_role_contains] record for the system-level admin role. This adds the application-specific admin role as a contained role of the system-level admin role.</td>
<td></td>
</tr>
</tbody>
</table>

Note: When you publish the application with this record, users with the application-specific admin role can access the application after installing it.

Access control rules in application administration apps

By default, when application administration is enabled for a scoped application, ACL rules for the scoped application are applied. If no ACL rules for the scoped application are found, global ACL rules can apply.

This behavior applies to configuration records created in tables that extend the Application File [sys_metadata] table only. You can also change the default behavior.

When no access control (ACL) rules for an application administration app are defined, global ACL rules can apply to the configuration records of the application administration apps. See Application files for more information.

To allow a table in an application administration app to inherit global ACL rules, check that the system property is true and add the table to the Application Administration ACL Inheritances table [sys_scoped_admin_acl_inheritance].

- glide.security.scoped_administration.honor_global_acl system property: If no scoped ACL rules are defined, application administration apps can inherit global ACL rules. By default, this property is enabled for new and upgraded instances.

- Application Administration ACL Inheritances [sys_scoped_admin_acl_inheritance] table: If no ACL rules for the application administration app are found, tables added to this list inherit global ACL rules. Only the administrator for the application administration app can add, remove, read the records owned by the application administration app in this configuration table.
Configure a table in an application administration app to inherit global ACL rules

To avoid duplicating global access control rules (ACLs) in your applications, you can configure application file tables in application administration apps to inherit global ACLs when no ACL rules for the scoped application are found.

**Before you begin**
Role required: admin role for the application

**Procedure**

1. Enter `sys_scoped_admin_acl_inheritance.list` in the navigation filter.

2. Click **New**.

3. In the **Table** field, select an application file table (a table that extends the Application File [sys_metadata] table) from the application administration app.
   Only the administrator for the application administration app can add/remove/read records owned by the application administration app in this configuration table.

4. Click **Submit**.
   If no ACL rules from the application administration app are found, tables added to this list inherit global ACL rules.

**Access enforcement for ServiceNow Store apps**

All production instances monitor and generate reports on usage patterns for ServiceNow Store apps. When subscription enforcement is enabled, users who are not subscribed to the app are blocked from performing fulfiller actions in the app.

**Overview**

The following actions are required to enable a production instance to enforce entitled usage of your ServiceNow Store App:

1. The usage admin at your organization uses the Subscription Management application to allocate fulfiller users to the subscription.

2. You decide on the enforcement mode, either:
• Monitor and report usage with no enforcement (default)

• In addition to monitoring and reporting usage, enforce that all usage must be by subscribed fulfiller users.

3. To enforce usage only by subscribed users, you configure the tables where only record owners or subscribed fulfiller users can make updates as fulfillment tables.

Creation restrictions across application scopes

The system restricts the creation of some configuration records when the current application scope does not match the application scope of the configuration record's target table.

Configuration record creation restrictions prevent one application from making unwanted changes to another application's data tables. These restrictions only apply when you create a configuration record whose target table belongs to another application. Configuration records that belong to the same application scope do not have these restrictions.

The system always enforces the following creation restrictions when a developer adds a configuration record belonging to another application scope.

<table>
<thead>
<tr>
<th>Configuration record type</th>
<th>Creation restrictions when target table is in another application scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access controls</td>
<td>• You can only create field-level access controls with a role-based requirement.</td>
</tr>
<tr>
<td></td>
<td>• You cannot create table-level access controls for a table in another application scope.</td>
</tr>
<tr>
<td></td>
<td>• You cannot create field-level access controls that apply to all fields.</td>
</tr>
<tr>
<td></td>
<td>• You cannot create access controls that use conditions.</td>
</tr>
<tr>
<td></td>
<td>• You cannot create access controls that use a script-based condition.</td>
</tr>
</tbody>
</table>
### Configuration record creation restrictions (continued)

<table>
<thead>
<tr>
<th>Configuration record type</th>
<th>Creation restrictions when target table is in another application scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business rules</td>
<td>• You can create a rule where <strong>When</strong> is async with any of the following options:&lt;br&gt;  ◦ <strong>Insert</strong>, <strong>Update</strong>, and <strong>Delete</strong> database operations. You cannot select <strong>Query</strong>.&lt;br&gt;  ◦ <strong>Set field values</strong> actions and scripts (the <strong>Script</strong> field).&lt;br&gt;• You can create a rule where <strong>When</strong> is <strong>before</strong> with any of the following options:&lt;br&gt;  ◦ <strong>Insert</strong>, <strong>Update</strong>, and <strong>Delete</strong> database operations. You cannot select <strong>Query</strong>.&lt;br&gt;  ◦ <strong>Set field values</strong> actions only. You cannot write scripts and you cannot abort the database transaction.</td>
</tr>
<tr>
<td>Calculated fields</td>
<td>You cannot create calculated fields for tables in another application scope.</td>
</tr>
<tr>
<td>Data policies</td>
<td>• You cannot create data policy rules for fields in another application scope.&lt;br&gt;• You cannot make a field mandatory.</td>
</tr>
<tr>
<td>Field styles</td>
<td>You cannot create field styles for fields in another application scope.</td>
</tr>
<tr>
<td>Form sections</td>
<td>• You cannot modify existing form sections created in another application scope.&lt;br&gt;• You can create new form sections.</td>
</tr>
<tr>
<td>Record producers</td>
<td>You must have create access to the application table to create records from a record producer.</td>
</tr>
<tr>
<td>UI policies</td>
<td>• You cannot create UI policy rules for fields in another application scope.&lt;br&gt;• You cannot make a field mandatory.</td>
</tr>
<tr>
<td>UI script</td>
<td>You cannot create a global UI script from a scoped application.</td>
</tr>
</tbody>
</table>
Configuration record creation restrictions (continued)

<table>
<thead>
<tr>
<th>Configuration record type</th>
<th>Creation restrictions when target table is in another application scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>Views</td>
<td>• You can create new views.</td>
</tr>
<tr>
<td></td>
<td>• You cannot modify existing views created in another application scope.</td>
</tr>
</tbody>
</table>

Delegated development and deployment

Delegated development allows designated users without a system admin role to develop or deploy applications on the Now Platform.

If you have the application-specific admin role or the system-level admin role, you can delegate application development to designated developers at the application level.

Delegated deployment tasks

You can also delegate deployment tasks (application publishing, first-time installation, or update) to developers or non-admin users, such as Change Management personnel. You delegate deployment tasks to specific users at the application level, or through assignment of specific user roles at the instance level.

<table>
<thead>
<tr>
<th>Assignment Method</th>
<th>Applies to</th>
<th>Available options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting deployment permissions in Manage Developers.</td>
<td>Specific applications</td>
<td>Publishing and upgrades of specific applications. Publishing options include the application repository, ServiceNow Store, and update sets.</td>
</tr>
<tr>
<td>See Delegate development and deployment permissions to personnel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assignment of deployment user roles to specific persons.</td>
<td>Local non-production instance (for example, Development or QA)</td>
<td>First-time installations and upgrades of all applications that contain the same company as the current instance. For example, applications for ABC Company and XYZ</td>
</tr>
</tbody>
</table>
### Application-specific permissions

Developer and deployment permissions are application-specific. For example, a developer who has permission to access all file types for one application does not necessarily have any developer permissions for another application. Administrators must set developer (and optionally deployment) permissions for each application. Administrators must be familiar with application files and the system table structure to set developer permissions. For example, a developer expected to create advanced business rules needs both the **All File Types** and **Allow Scripting** developer permissions.

**Important:** If Application administration is enabled, only an application administrator for the target application can delegate developers for an application. Application administrators do not have system admin privileges. To enable a delegated developer to perform the functions granted in the developer permissions, the delegated developer must also be given the application administrator role.

Setting each permission grants one or more system-managed delegated development roles, allowing system admins to retain control over the system. System admins no longer have to elevate developers (or users who deploy applications) to the system admin role to enable them to develop or deploy applications.

### Developer and deployment permissions example

As a system administrator, you want to assign Abel Tuter certain developer and deployment permissions for a specific application in your Development instance. For more details on developer and deployment permissions, see Delegate development and deployment permissions to personnel.
Related information

Application files
Install a ServiceNow Store application
Install an update to a ServiceNow Store application
Application sharing

Delegate development and deployment permissions to personnel

A system administrator can assign a non-administrator user or group as a developer or deployment resource for a specific application. You can set permissions that designate what specific actions the assigned user can perform in the current instance.

Before you begin

- Role required: admin or application administrator

  If Application administration is enabled, only an application administrator of the target application can delegate developers to an application. If application administration is not enabled, an admin user can delegate developers.

- Records required:
  - Application
  - User
  - Group
Procedure

1. Navigate to **System Applications > My Company Applications**.

2. Click the name of the application to which you want to add developers. The system opens the application record.

3. Click **Manage Developers**.
   The system displays the Developer Permissions window.

4. Select the type of developer (or user without system admin roles) you want to assign.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developers</td>
<td>To add individual users as developers or deployers.</td>
</tr>
<tr>
<td>Groups</td>
<td>To add all members of a group as developers or deployers.</td>
</tr>
</tbody>
</table>

5. In **Developer Name** or **Group Name**, enter the name of the user or group you want to grant developer or deployment permissions.

6. Select specific developer and deployment permissions for the application. For details, see Developer and deployment permissions.

7. Click **Save**.
   The system assigns the designated developer and deployment permissions for the application.

Developer and deployment permissions

Using Manage Developers, administrators can assign one or more developer and deployment permissions to a group or user for a specific application. These permissions designate the specific actions the assigned user can perform for the application.

For example, you might grant permissions that enable a user to upgrade the application, publish to the application repository and ServiceNow Store, but prevent publishing to an update set.

Developer permissions

<table>
<thead>
<tr>
<th>Permission</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete Application</td>
<td>Grants the assigned developer within a scoped app rights to delete the application.</td>
</tr>
<tr>
<td>Permission</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Source Control</td>
<td>Grants the assigned developer full access source control.</td>
</tr>
<tr>
<td>All File Types</td>
<td>Grants the assigned developer access to all application file types, including some not granted by the other options. This permission is equivalent to granting the user the admin role but with some limitations. Specifically, it provides access to all file types that are configured in your application per the Manage Developers task in the Application Creator. For an example of such file types, see the permissions example in Delegated development and deployment.</td>
</tr>
<tr>
<td>Integrations</td>
<td>Grants the assigned developer access to web service APIs, REST APIs, and data sources.</td>
</tr>
<tr>
<td>Reporting</td>
<td>Grants the assigned developer access to reports and scheduled reports.</td>
</tr>
<tr>
<td>Mobile Studio</td>
<td>Grants the assigned developer access to Mobile Studio.</td>
</tr>
<tr>
<td>UI Builder</td>
<td>Grants the assigned developer access to UI Builder to create pages for experiences.</td>
</tr>
<tr>
<td>Workflow</td>
<td>Grants the assigned developer access to the Workflow Editor and Activity Creator.</td>
</tr>
<tr>
<td>Service Catalog</td>
<td>Grants the assigned developer access to catalog related file types such as catalog items, record producers, and variables.</td>
</tr>
<tr>
<td>Flow Designer</td>
<td>Grants the assigned developer access to the Flow Designer design environment to create flows and actions. Script action steps require the Allow Scripting permission.</td>
</tr>
<tr>
<td>Service Portal</td>
<td>Grants the assigned developer access to Service Portal editors and tools.</td>
</tr>
<tr>
<td>Tables &amp; Forms</td>
<td>Grants the assigned developer access to model and layout related file types such as table columns, form layout, and list layout.</td>
</tr>
<tr>
<td>Manage ACLs &amp; Roles</td>
<td>Grants the assigned developer access to security-related file types such as access controls and user roles.</td>
</tr>
<tr>
<td>Allow Scripting</td>
<td>Grants the assigned developer write access to script fields such as those in business rules, client scripts, and Flow Designer script action steps.</td>
</tr>
<tr>
<td>Permission</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Manage Collaborators</td>
<td>Grants the assigned developer the ability to invite and manage users and groups. This permission allows the delegated developer to further invite and manage other developers to the application.</td>
</tr>
<tr>
<td>Invite Collaborators</td>
<td>Grants the assigned developer the ability to invite users and groups. This permission allows the delegated developer to further invite other developers to the application.</td>
</tr>
</tbody>
</table>

**Deployment permissions**

<table>
<thead>
<tr>
<th>Permission</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upgrade App</td>
<td>Grants a user with an assigned delegated developer role permission to upgrade the associated application after it has been installed in the current instance.</td>
</tr>
<tr>
<td>Publish To Update Set</td>
<td>Grants a user with an assigned delegated developer role permission to publish the associated application to an update set in the current instance.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Users with this permission cannot also have the <strong>Manage Update Set</strong> permission.</td>
</tr>
<tr>
<td>Publish To App Store</td>
<td>Grants a user with an assigned delegated developer role permission to publish associated application to the ServiceNow Store in the current instance.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The <strong>Upgrade App</strong>, <strong>Publish To App Repo</strong>, and <strong>Publish To App Store</strong> permissions display by default. The <strong>Publish To Update Set</strong> permission only displays if manually enabled by a system administrator. For more details, see Display or hide update set deployment permissions.</td>
</tr>
<tr>
<td>Manage Update Set</td>
<td>Grants a user with an assigned delegated developer role permission to manage local and retrieved update sets. This permission allows users to create, update, and delete local update sets as well as preview, resolve conflicts, and commit retrieved update sets.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Users with this permission cannot also have the <strong>Publish To Update Set</strong> permission.</td>
</tr>
<tr>
<td>Permission</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Publish To App Repo</td>
<td>Grants a user with an assigned delegated developer role permission to publish the associated application to the application repository in the current instance.</td>
</tr>
<tr>
<td>Submit for Deployment</td>
<td>Grants a user with assigned delegated developer role permission to Submit the associated application for review and deployment.</td>
</tr>
</tbody>
</table>

⚠️ **Note:** The update set deployment permissions are hidden by default and require a system administrator to enable them with system properties. See Display or hide update set deployment permissions for more information.

⚠️ **Note:** The Submit for Deployment, Manage Collaborators, and Invite Collaborators delegated development permission sets are only available with the Developer Collaborator feature. They will not be shown in Manage Developers.

### Remove a developer

Removing a user as a developer prevents the user from developing, changing, or deploying the application in the current instance.

**Before you begin**

Role required: admin or application administrator

If Application administration is enabled, only an application administrator of the target application can delegate developers to an application. If application administration is not enabled, an admin user can delegate developers.

**Procedure**

1. Navigate to **System Applications > Applications**.
2. Click the application name of the application from which you are removing developers.
   - The system opens the application record.
3. Click **Manage Developers**.
   - The system displays the Developer Permissions window.
4. Point to the developer you want to remove.
   - The system displays a minus icon next to the developer name.
5. Click the minus icon next to the developer name. The system removes the developer and any associated application roles.

**Display or hide update set deployment permissions**

Display or hide deployment permissions for update sets from the Manage Developers dialog.

**Before you begin**

Role required: admin

**About this task**

System properties control the visibility of the Publish To Update Set and Manage Update Sets permissions. By default, both deployment permissions for update sets are hidden.

**Procedure**

1. In the Navigation filter, enter `sys_properties.list`

   The entire list of properties in the System Properties [sys_properties] table appears.

   **Tip:** For details on how to add and enable system properties, see Add a system property.

2. Enable or disable specific deployment permissions by adding or updating the following system properties.

**Deployment permissions**

<table>
<thead>
<tr>
<th>System property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.snc.dd.manage_update_set_enabled</td>
<td>Enables or disables display of the Manage Update Set permission.</td>
</tr>
<tr>
<td>System property</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>com.snc.dd.publish_to_app_repo_enabled</td>
<td>Enables or disables display of the <strong>Publish To App Repo</strong> permission. The default value for this system property is <strong>true</strong>. To disable display of this permission, set this value to <strong>false</strong>.</td>
</tr>
<tr>
<td>com.snc.dd.publish_to_app_store_enabled</td>
<td>Enables or disables display of the <strong>Publish To App Store</strong> permission. The default value for this system property is <strong>true</strong>. To disable display of this permission, set this value to <strong>false</strong>.</td>
</tr>
<tr>
<td>com.snc.dd.publish_to_update_set_enabled</td>
<td>Enables or disables display of the <strong>Publish To Update Set</strong> permission. The default value for this system property is <strong>false</strong> by default, which disables display of this permission. To enable the display of this permission, set this value to <strong>true</strong>.</td>
</tr>
<tr>
<td>com.snc.dd.upgrade_app_enabled</td>
<td>Enables or disables display of the <strong>Upgrade App</strong> permission. The default value for this system property is <strong>true</strong>. To disable display of this permission, set this value to <strong>false</strong>.</td>
</tr>
</tbody>
</table>
Instance-specific deployment user roles

Assign roles that enable non-admin users install or upgrade all applications in specific instances. You delegate these tasks by manually assigning specific user roles per instance.

For example, you can assign user roles to Change Management personnel that allow them to perform application installations in non-production (development or QA) instances.

**Important:** Give careful thought to whom, and in what instances you assign installation and upgrade user roles. You may only want to assign these roles to non-system administrators in non-production instances (for example, Developer or QA), but decline to do so in production instances. Given the potential impact of installing and upgrading applications, you may want to leave the responsibility for installations and upgrades in production instances to a system administrator.

Add deployment user roles

A system administrator can assign user roles to specific personnel that allow them to perform first application installations only, or install and upgrade applications in a local instance.

**Before you begin**

Role required: admin

**Procedure**

1. Navigate to User Administration > Users and then open a user record.

   ![Note](#) For details on how to assign a role to a user, refer to Assign a user role.

2. In the Roles related list, click Edit.

3. In the Collection list, select the desired deployment roles, and then click Add.

**Delegated deployment roles**

<table>
<thead>
<tr>
<th>User Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_appclient.app_client_company_installer</td>
<td>Allows a designated person to perform first-time installations of applications displayed on the Application Client page that contain the same company as the current instance. A user with this role cannot install</td>
</tr>
<tr>
<td>User Role</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>an application for another company. For example, applications for ABC Company and XYZ Company display on the Application Client page. A user with this role can only install XYZ Company applications when logged in to a XYZ Company instance. The user cannot install applications for ABC Company.</td>
</tr>
<tr>
<td>sn_appclient.app_client_user</td>
<td>Allows a designated person to install and upgrade all applications displayed on the Application Client page.</td>
</tr>
</tbody>
</table>

4. Click **Save**.

**System-managed developer and deployment roles**

Although system admins can still manually assign and remove the user roles, they are encouraged to let the system manage the following delegated developer roles.
<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>delegated_developer</td>
<td>User has one or more developer permissions.</td>
</tr>
<tr>
<td>Roles that start with an sn_dd prefix (for example,</td>
<td>User has an application-specific development permission. The role name indicates an application scope to which it applies. For example, after a user with a <code>sn_appclient.app_client_company_installer</code> role installs a company application, the system automatically grants a <code>sn_dd_&lt;app_name&gt;_upgrade_app</code> delegated deployment role. This role allows the user to upgrade the application when future updates are published on the Application Client page.</td>
</tr>
<tr>
<td>glide.security.add_admin_contained_roles_to_system</td>
<td>Default Value: true if the property is true, then all the roles, directly or indirectly contained by the admin role, are added to the system user, including the scoped-admin roles. <strong>Note:</strong> Setting the property to false results in the old behavior, where the system user has the admin role, but not any of the scoped-admin roles contained by the admin role. <img src="https://via.placeholder.com/15" alt="important" /> <strong>Example:</strong> The admin role contains the <code>sn_templated_snip.template_snippet_admin</code> role Old behavior: The system user does not have the <code>sn_templated_snip.template_snippet_admin</code> role. New behavior: The system user has the <code>sn_templated_snip.template_snippet_admin</code> role and other scoped roles that it contains.</td>
</tr>
</tbody>
</table>
Note: Users with delegated developer roles cannot add or remove the system admin role.

Assign source control permissions
Ability to assign full access to source control for a particular scope to a delegated developer. The Source Control menu is only visible if you have the correct permissions for the application that you are working in.

Before you begin
Role required: admin

Procedure
2. Open an application for which you want to assign delegated developer permissions.
3. As a System Admin, click File > Manager Developers.
4. Choose a developer to apply delegated developer permissions.
5. Toggle on Source Control, and then click Save.
6. The delegated developer can now access the Source Control menu options. For more information on source control options, see Available source control operations.

**Assign delete permissions**

Ability to assign the ability to delete an application to a delegated developer.

**Before you begin**

Role required: admin

**Procedure**

2. Open an application for which you want to assign delegated developer permissions.
3. As a System Admin, click File > Manager Developers.
4. Choose a developer to apply delegated developer permissions.
5. Toggle on Delete Application, and then click Save.
6. The delegated developer can now delete the application. For more information on deleting an application, see Delete an application.

Domain separation and Delegated Development

Domain separation is unsupported in the Delegated Development feature. Domain separation enables you to separate data, processes, and administrative tasks into logical groupings called domains. You can control several aspects of this separation, including which users can see and access data.

Support level: No support

- The domain field may exist on data tables but there is no business logic to manage the data.
- This level is not considered domain-separated.

For more information on support levels, see Application support for domain separation.
How domain separation works with Delegated Development

Delegated Development is not a full application but rather, a feature in the Custom Business Applications suite, meaning it works alongside other features, including domain separation. Delegated development can be used in a domain environment only by the instance owner’s developers.

Related information

Domain separation for service providers

Application collaboration

With application collaboration, you can manage the permissions for the collaborators who develop a platform application. If you have the admin role, you can manage delegated development permissions to designated developers at the application level.

Collaboration descriptors

You can use the predefined collaboration descriptors that are standard with activation, or create a custom collaboration descriptor. By using a collaboration descriptor, you can assign, manage, and monitor delegated development permissions.

<table>
<thead>
<tr>
<th>Descriptors</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Owner</td>
<td>Owner of the application that manages other collaborators and can delete the application.</td>
</tr>
<tr>
<td>Editor</td>
<td>Standard descriptor to invite collaborators.</td>
</tr>
<tr>
<td>Custom</td>
<td>Non-standard (custom) descriptor created by the user.</td>
</tr>
</tbody>
</table>

Development permissions

By setting permissions, you can retain control over the system. You assign permissions to developers (or users who deploy applications) so that they can develop or deploy applications.
Development permissions

<table>
<thead>
<tr>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Management</td>
<td>Capabilities to delete the application and version source control access.</td>
</tr>
<tr>
<td>File type Access</td>
<td>File types that the user has access to.</td>
</tr>
<tr>
<td>Security / Entitlement</td>
<td>Capabilities to manage application security.</td>
</tr>
<tr>
<td>Programming Tools</td>
<td>Capabilities for script fields access.</td>
</tr>
<tr>
<td>Deployment</td>
<td>Deployment capabilities.</td>
</tr>
</tbody>
</table>

Application collaboration users

You can assign a collaboration descriptor for an individual by creating an app collaboration user. The user can only be assigned one collaboration descriptor for any application.

Application collaboration groups

You can assign a collaboration descriptor for a group of users by creating an app collaboration group. The group can only be assigned one collaboration descriptor for any application.

Install the app collaboration application

Install the app collaboration application so that you can view the collaboration feature in the UI.

Before you begin

Download the app collaboration application so that you can view the collaboration feature.
Role required: admin

Procedure

1. Navigate to **System Applications > All Available Applications > All**.

2. Click the search icon (🔍) in the middle of the screen to search for the application collaboration application.

3. Click **Install**.
Note: If app collaboration is already installed, the Install button is inactive.

4. Click Activate.

Results
The app collaboration feature is active.

Note: After you install and activate the app collaboration plugin, all delegated developers will receive custom collaboration descriptors based on their current permissions and applications. The base system table provides standard Owner and Editor collaboration descriptors which are used in the collaboration feature.

Note: If you install this plugin without App Engine Studio (AES) for an application, you should manage the delegated development permissions with this plugin or use the old Manage Developers screen to manage the permissions. Do not use both the plugin and the Manage Developers screen to manage the delegated development permissions.

Note: To use the old Manage Developers screen, select your application and select Manage Developers in the related links on the form.

Create collaboration descriptors to assign permissions
Create descriptors in the collaboration application so that you can assign permissions to users or groups.

Before you begin
Role required: admin

About this task
Use the predefined collaboration descriptors that are standard with activation, or create a custom collaboration descriptor. Collaboration descriptors enable you to assign, manage, and monitor delegated development permissions for each application or uniformly across multiple applications. You can assign each collaborator with one collaboration descriptor for an application. However, users can have multiple collaboration descriptors simultaneously if they belong to groups where collaboration descriptors have been assigned.

Procedure
1. Navigate to App Engine > Collaboration > Descriptors.
2. Click New.
The collaboration descriptor form appears.

3. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name for the descriptor.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the descriptor that includes permissions capabilities.</td>
</tr>
<tr>
<td></td>
<td>Examples are as follows:</td>
</tr>
<tr>
<td></td>
<td>• Owner</td>
</tr>
<tr>
<td></td>
<td>• Editor</td>
</tr>
<tr>
<td>Application</td>
<td>Scope of the collaboration descriptor. An example is the application the descriptor will be used for.</td>
</tr>
<tr>
<td>Standard</td>
<td>Option for inviting other collaborators with this role.</td>
</tr>
</tbody>
</table>

4. Click Submit.

Add permissions to collaboration descriptors

Add permissions to collaboration descriptors to manage your user's capabilities, such as the ability to delete the application or manage collaborators.

Before you begin
Role required: admin

Procedure
1. Navigate to App Engine > Collaboration > Descriptors.
2. To modify the permissions, select a collaboration descriptor from the related list.
3. To view the permission's options, click Edit... in the related list.
4. To add or remove permissions to the descriptor, click the arrows.
5. Click Save and Update.
Assign collaboration descriptors to users
Assign collaboration descriptors to your users for a specific application so that you can define specific permissions for these users.

Before you begin
Role required: admin

Procedure
1. In the related list, click Application collaboration users.
2. Click New.
3. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collaboration Application</td>
<td>Application that the user works on.</td>
</tr>
<tr>
<td>Collaboration User</td>
<td>User that works on the application.</td>
</tr>
<tr>
<td>Collaboration Descriptor</td>
<td>Descriptor for the user on this application.</td>
</tr>
</tbody>
</table>

4. Click Submit.

Assign collaboration descriptors to groups
Assign collaboration descriptors to user groups for a specific application so that you can define specific permissions for these user groups.

Before you begin
Role required: admin

Procedure
1. In the related list, click Application collaboration groups.
2. Click New.
3. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collaboration Application</td>
<td>Application that the group will work on.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>Collaboration Group</td>
<td>Name of the collaboration group.</td>
</tr>
<tr>
<td>Collaboration Descriptor</td>
<td>Descriptor for the group on this application.</td>
</tr>
</tbody>
</table>

4. Click **Submit**.

**Application Management**

The system offers several ways to manage applications. You must have the admin role to perform these procedures.

**Application management actions**

<table>
<thead>
<tr>
<th>Action</th>
<th>Options</th>
<th>Use case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Install an application</td>
<td>• Install new application</td>
<td>Application testing.</td>
</tr>
<tr>
<td></td>
<td>• Install updates</td>
<td></td>
</tr>
<tr>
<td>Select the current application</td>
<td>• Select an application from the applications list</td>
<td>Edit an application.</td>
</tr>
<tr>
<td></td>
<td>• Select an application from the application picker</td>
<td></td>
</tr>
<tr>
<td>Update the application record</td>
<td>• Add tables</td>
<td>Add application logic.</td>
</tr>
<tr>
<td></td>
<td>• Add roles</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Add menus</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Add dependencies</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Add Update Sets</td>
<td></td>
</tr>
<tr>
<td>Set the application state</td>
<td>• Deactivate an application</td>
<td>Retire a legacy application.</td>
</tr>
<tr>
<td></td>
<td>• Activate an application</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Delete an application</td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Application management actions (continued)

<table>
<thead>
<tr>
<th>Action</th>
<th>Options</th>
<th>Use case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Share an application</td>
<td>• Make available on another instance &lt;br&gt;• Upload to the ServiceNow Store</td>
<td>Move an application from development to testing.</td>
</tr>
<tr>
<td>Remove an application</td>
<td>• Deactivate &lt;br&gt;• Delete &lt;br&gt;• Uninstall</td>
<td>Retire an obsolete or legacy application.</td>
</tr>
<tr>
<td>Move application files from one global application to another</td>
<td>None</td>
<td>Edit a legacy application.</td>
</tr>
</tbody>
</table>

Indicators are color coded to indicate if it is an error, warning, or info message.
- An indicator in red color, indicates that it is an error or warning message. This requires you to perform a relevant action.
- An indicator in green color, indicates that it is an info message.

### Opt in to the ServiceNow Store products

Opt in to the required product available on ServiceNow Store by navigating to the ServiceNow Products module.

**Before you begin**
Role required: admin

**Procedure**

1. **Navigate to** System Applications > All Available Applications > ServiceNow Products.  
   ServiceNow products that are available on ServiceNow Store are displayed. The products are organized based on the product family they are part of.

2. To opt in to the ServiceNow products:
   - a. On the tile of the required product, click Opt-in.
   - b. Enter your Now Support credentials.
c. Click Opt-in. A confirmation message is displayed and the Opted-in label is displayed on the product tile.

d. To view the applications that are part of the product, click the product tile. All applications that are part of the product are displayed in the Licensed and Not Licensed tabs.

Note:
- If the application is already available (that is, installed or available for update) in your ServiceNow instance, click Manage Application to navigate to the application page and install, uninstall, or repair the application.
- If the application isn’t available in your ServiceNow instance, click View Detail to navigate to the application listing page, view the application details and dependencies, and navigate to ServiceNow Store.

3. To opt in to the ServiceNow product family:

a. Click Product Family Opt-in.

b. Enter your Now Support credentials.

c. Click Opt-in. You have opted in to all product in the required family.

Manage entitlements from ServiceNow instance
Manage entitlement of applications and products from your ServiceNow instance.

Before you begin
Role required: admin

Entitle an application from ServiceNow instance
Manage entitlement of applications from your ServiceNow instance.

Before you begin
Purchase the application on ServiceNow Store.
Procedure

1. Navigate to System Applications > All Available Applications > All.

2. From the filter options, select the Instance not entitled indicator. All the purchased unentitled apps are displayed.

3. Click the menu icon (⊕) on the application tile, and click Entitle this instance.

4. In the Application entitlement dialog box, click OK.

Results
Application entitlement is provided to your ServiceNow instance.

What to do next
Install the application if it is compatible with the version of ServiceNow instance.

Entitle a product from ServiceNow instance
Manage entitlement of products from your ServiceNow instance.

Before you begin
Purchase the product on ServiceNow Store and opt in from your ServiceNow instance.

Procedure

1. Navigate to System Applications > All Available Applications > All.

2. From the filter options, select the Product not entitled indicator. All the purchased unentitled products are displayed.

3. Click the menu icon (⊕) on the product tile, and click Entitle this instance.
4. In the Application entitlement dialog box, click **OK**.

**Results**
Product entitlement is provided to your ServiceNow instance.

**What to do next**
*Install* the product if it is compatible with the version of ServiceNow instance.

**Skipped records that occur during application installation**

Some records may be skipped when you make local updates to global or scoped applications. Skipped records can occur either when you modify the metadata records in the instance to which you're deploying or when you apply an update set. Depending on the deployment model you use and the state of applicable properties, you may risk "skipped records," which are generated in the sys_upgrade_history_log. Learn what to expect on your instance when you upgrade an application using various different methods.

**General use cases**
Under most circumstances, these general use cases occur during application installation:

<table>
<thead>
<tr>
<th>Deployment type</th>
<th>Application type</th>
<th>Expected outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Control</td>
<td>Scoped</td>
<td>No skips, application loads from Source Control. If local update xml exists, you are prompted to stash those changes.</td>
</tr>
<tr>
<td>Source Control</td>
<td>Global</td>
<td>No skips, application loads from Source Control. If local update xml exists, you are prompted to stash those changes.</td>
</tr>
<tr>
<td>Deployment type</td>
<td>Application type</td>
<td>Expected outcome</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>xml exists, you are</td>
<td></td>
<td>prompted to stash those changes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>App Repository</td>
<td>Scoped</td>
<td>Skipped records are generated and customizations preserved.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>App Repository</td>
<td>Global</td>
<td>Application changes applied and skipped records are generated only when a superior claim is found. Refer to Claim Outcomes to Review related list for more details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Store</td>
<td>Scoped</td>
<td>Skipped records are generated, customization changes applied.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source Control</td>
<td>Customization</td>
<td>No skips, customization loads from Source Control. If local update xml exists, you are prompted to stash those changes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>App Repository</td>
<td>Customization</td>
<td>No skipped records are generated, customization changes applied.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Store/App Repository</td>
<td>Scoped and</td>
<td>No skipped records are generated unless the customization updates the base application record. In this case, a log entry of a skipped update for the base application is created.</td>
</tr>
<tr>
<td></td>
<td>Customization</td>
<td></td>
</tr>
</tbody>
</table>
Author Elective Updates

It's important to understand the purpose of the **author_elective_update** folder.

When ServiceNow packages your application for the repository or for committing to Source Control, additional elements (which might have been previously ignored), are unloaded to the **author_elective_update** folder. (These deleted elements are often referred to simply as "deletes.") When your application is installed on your own instance, the deleted files are automatically loaded from the **author_elective_update** folder.

For example, if you changed the schema of your application by deleting a table or a column, those files are tracked in the folder but are not applied by default. There are specific rules that apply to author elective updates that you can change, based on the properties you can set. This folder contains metadata deleted files, including schema changes, and choice set unloads that you can apply or ignore. Whether you apply or ignore them, and whether corresponding skipped records are generated, depends on the state of the following properties and processing flow:

<table>
<thead>
<tr>
<th>Property name</th>
<th>Behavior</th>
<th>Default</th>
<th>Used in Source Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.glide.apps.include_only_sys_choice</td>
<td>Loads only deleted elements and updates to sys_choice fields from author_elective_update</td>
<td>False</td>
<td>No</td>
</tr>
<tr>
<td>com.glide.apps.include_my_schema</td>
<td>Loads only deleted elements to schema files from author_elective_update. Applies to customer application installations and not third-party apps.</td>
<td>False</td>
<td>No</td>
</tr>
<tr>
<td>com.glide.apps.force_skips</td>
<td>Creates skipped records for all of author_elective_update</td>
<td>False</td>
<td>No</td>
</tr>
<tr>
<td>Property name</td>
<td>Behavior</td>
<td>Default</td>
<td>Used in Source Control</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------</td>
<td>------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>com.glide.apps.include_my_deletes</td>
<td>Process author_elective_update</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>com.glide.apps.include_global_deletes</td>
<td>Process author_elective_update for global applications</td>
<td>False</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Note:**

When there is no property for `com.glide.apps.include_my_schema`, it defaults to False. However, since the Orlando release, new instances have a default entry in the table to set it as True.
Claims

Claims apply to global applications and application-customizations. In global applications, claims allow the system to choose a record deterministically should the same file be included in multiple applications. If you see a superior claim, an upgrade history log for the skipped record is created to identify a conflicting claim status. See Claim Outcomes to Review related list for details.

Accessing unpurchased recommended apps on ServiceNow Store

Administrators can access the list of unpurchased ServiceNow Store apps, view recommended apps, and view details of the required app without navigating to the ServiceNow Store.

To access the list of unpurchased ServiceNow Store apps, navigate to System Applications > All Available Applications > Available To Obtain From Store. The recommended apps, have the Recommended indicator on the application tile. You can also use the filter options available under Additional Filters to populate the required apps. The filter options include:

<table>
<thead>
<tr>
<th>Filter option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Free Trial</td>
<td>App is available for a free trial.</td>
</tr>
<tr>
<td>Service Graph Connector</td>
<td>App type is Service Graph Connector.</td>
</tr>
<tr>
<td>Purchase In Process</td>
<td>Request to purchase the app is in process and the app has not been purchased on ServiceNow Store yet.</td>
</tr>
<tr>
<td>Rejected Request</td>
<td>Request for app trial or purchase is rejected.</td>
</tr>
<tr>
<td>Request in Progress</td>
<td>Request for app trial or purchase has been initiated and response from the vendor is awaited.</td>
</tr>
</tbody>
</table>

• To view the app details that are available on ServiceNow Store, click View Details.

• To purchase or try the required app, on the app details page, click:
  ◦ Get for free apps.
  ◦ Buy for paid apps.
- **Request App** for free or paid apps that require approval from vendor to purchase the app.
- **Request Trial** for free or paid apps that require approval from vendor to try the app.
- **Try** before procuring the application directly from the application manager. You will be redirected to the ServiceNow Store from where you can purchase the app.

**Request and install free application trial**

Request and install a free application trial of an application before procuring the application directly from your ServiceNow instance, without navigating to ServiceNow Store.

**Before you begin**
Role required: admin

**Procedure**

1. **Optional:** Obtain approval from the vendor to try the application.
   - **Note:** Some applications require vendor approval to try the application.
     - a. Navigate to **System Application > All Available Applications > Available To Obtain From Store**.
     - b. Search for the required application and click the application record.
     - c. Click **Request Trial**.
     - d. Enter your Now Support credentials.
     - e. Click **Sign in**.
       A confirmation message is displayed that the trial request is sent to the vendor. Once the vendor approves the request, you will be notified through an email.

2. Navigate to **System Application > All Available Applications > Available To Obtain From Store**.
   - **Note:** If the application requires vendor approval, perform these steps after you have been notified through email that your request is approved.

3. Search for the required application and click the application record.
4. Click **Try**.
5. In the pop-up window, review information in these tabs and click **Continue**.
• Usage Analytics and Contact
• Platform Licenses
• Terms and Conditions

Note: Select the I read and accept terms of use above check box.
You can download the terms and conditions documents.
In the Confirmation tab, a confirmation message is displayed that the application can be installed.

An application trial is available for 30 days. A green indicator indicates the trial expiration date. After the trial period has expired, a red indicator is displayed to indicate that the trial period has expired.

6. Click Go to install.

Request a free or paid application
Request a free or paid application that requires approval from the vendor to purchase the application directly from the ServiceNow instance, without navigating to ServiceNow Store.

Before you begin
Role required: admin

Procedure
1. Navigate to System Application > All Available Applications > Available To Obtain From Store.
2. Search for the required application and click the application record.
3. Click Request App.
   A confirmation message is displayed that the request is sent to vendor. Once the vendor approves the request, you will be notified through an email.

Get a free application
Get a free application from your ServiceNow instance, without navigating to the ServiceNow Store.

Before you begin
Role required: admin
Procedure
1. Navigate to System Application > All Available Applications > Available To Obtain From Store.
2. Search for the required application and click the application record.
3. Click Get.
4. In the pop-up window, review information in these tabs and click Continue:
   • Usage Analytics and Contact
   • Platform Licenses
   • Terms and Conditions

   Note: Select the I read and accept terms of use above check box.
   You can download the terms and conditions documents.
   In the Confirmation tab, a confirmation message is displayed that the application can be installed.
5. Click Go to install.

Buy a paid application
Request a quote and buy a paid application from your ServiceNow instance, without navigating to the ServiceNow Store.

Before you begin
Role required: admin

Procedure
1. Navigate to System Application > All Available Applications > Available To Obtain From Store.
2. Search for the required application and click the application record.
3. Click Buy.
4. In the pop-up window, review information in these tabs and click Continue:
   • Usage Analytics and Contact
   • Platform Licenses
Terms and Conditions

Note: If Payment Type is Purchase Order, a quote with the displayed payment details is requested. You can cancel or resent quote.

- Review the payment and tax details.
- Select the I read and accept terms of use above check box.

You can download the terms and conditions documents.

- In the Confirmation tab, a confirmation message is displayed that the quote request is sent. An email notification is sent when the request is processed.

- When you close the pop-up window, these buttons are displayed in the application page:
  - Resend Quote
  - Cancel Quote
  - Purchase with PO

5. Optional: If you want to resend quote, click Resend Quote.

6. Optional: If you want to cancel the quote:
   - Click the Cancel Quote button.
   - Provide cancellation reason.
   - Click Cancel Quote.

7. Click Purchase with PO to purchase the application.

- If applicable, select the I authorize ServiceNow to invoice my company without having received a Purchase Order, and I represent that I have authority to represent my company to provide such authorization check box.

- Else, enter PO Number and choose file for Upload PO Document.

8. Click Place an Order.

   An email notification is sent when the PO request is reviewed and approved. Then, you can install the application.

Complete the offline application purchase transaction

Complete the application purchase transaction from your ServiceNow instance after purchasing the application offline, without navigating to the ServiceNow Store.

Before you begin
Role required: admin
Procedure

1. Navigate to System Application > All Available Applications > All.

2. Search for the required application and click the application record and click Install.

   Note: Option to install the application is available after the application is purchased offline. Also, the indicator Click install to complete purchase is displayed in the application tile.

3. Enter your Now Support credentials.

4. Click Sign in.

5. Select the I read and accept terms of use above check box.

   You can download the terms and conditions documents.

6. Click Continue.

   Application is purchased and is ready to be installed.

   Note: If the application isn't compatible with the version of your ServiceNow instance, an error message is displayed informing you the same.

7. Click Install.

Schedule installation

Schedule the installation of a plugin, application, or product at the required time from your ServiceNow instance. You can simultaneously schedule multiple installations.

Before you begin
Role required: admin

Procedure

1. Navigate to System Definition > Plugins.

2. Search for the required plugin, application, or product.

3. Click the schedule installation icon ( ).

   Note:
   - If the app is blocked from installation, the installation and schedule installation icons are disabled.
   - You can't schedule the installation of an in-development plugin.
4. Specify the time at which you want to schedule the installation.

![Schedule Installation dialog box](image)

5. Click **Next**.

   **Note:** If another installation is scheduled at the same time, a message is displayed that the installation can't be scheduled at the mentioned time and auto-populates the next available slot.

![Schedule Installation dialog box with notification message](image)

6. In the Schedule Activation window, choose the option to reschedule if the scheduled installation doesn't complete. That is, the admin can choose to retry the installation once or exit the installation.
The installation is scheduled and a confirmation message is displayed.

**What to do next**

Click the schedule summary icon (📅) to view the installation status and schedule calendar.
• To view the status of the installation, click **View Status**.
  ◦ If the installation is successful, the details of the plugin, application, or product are displayed.
  ◦ If the installation fails, the reason for the failure is displayed along with other details.

• To reschedule or delete the scheduled installation, click **Reschedule** or **Delete** in the app tile.

Note: If you have scheduled the installation of a product, all apps in the product will also be scheduled for installation. However, you can't modify or delete the schedule of an individual app. You can only modify or delete the schedule of the product.

• To view the schedule of installations, click **Full schedule**. A calendar displays the details of the scheduled installations.

**Install or update all applications in a ServiceNow product**

Install or update all applications that are part of a ServiceNow product from your ServiceNow instance at a time.

**Before you begin**
Role required: admin

**Procedure**

1. Navigate to **System Applications > All Available Applications > All**.
2. Select the **ServiceNow Products** option in the **Listing type** filter options.
   The list of the ServiceNow products in displayed.
3. Search for the required ServiceNow product and open the record.
   Use the **Licensed** and **Not Licensed** tabs to view details about applications in the product.
4. Click **Install/Update All**.
Note: By default, application of the latest version is installed. If you want to install the application of another version, select the required version and select the application check box.

5. Optional: Select the Load demo data check box to load demo data for all the applications displayed in the Install pop-up window.

6. Click Install.

- If there are any errors or conflicts, the details are displayed in the Install pop-up window. To proceed with installation, you must address the errors and resolve the conflicts.
- If there are no errors or conflicts, the applications are installed or updated, and a confirmation message is displayed.

7. Optional: While installation is in progress, you can have application installed in the background by clicking Continue in background. After refreshing the page, this message is displayed: Product installation in progress, click here to view details. Click the link to see the installation progress.

Results
A confirmation message is displayed if all the applications are installed. Otherwise, the number of successful and unsuccessful installations are displayed.

Install multiple applications in a batch
The Batch Installation feature helps you install multiple applications, plugins, or application-customizations together, instead of having to install them serially one at a time. You can access this feature either from the Continuous Integration and Continuous Delivery (CI/CD) APIs via three new endpoints, or via the CI/CD Spoke on IntegrationHub using Flow Designer subflows and actions.

Overview
When managing many types of applications, it can be a challenge to have separate pipelines for each application and to wait for them to serially deploy to the same environment. With the Batch Installation feature, you can install several applications together by bundling them into a single JSON manifest file. These applications include scoped or global applications, plugins, and application-customizations.

A batch can contain any combination of the following types:
• A ServiceNow plugin
• Application: Scoped applications in the ServiceNow Store that your company doesn’t own, scoped or global applications in the Application Repository owned by your company, and your application-customizations for Store applications stored in the Application Repository.

Batch installation methods
For details on installing applications in a batch via the REST API see Continuous Integration/Continuous Delivery (CI/CD) API.

For details on installing applications in a batch with the Flow Designer application, see Continuous Integration and Continuous Delivery (CICD) Spoke.

Install a ServiceNow Store application
Install an application that you purchased from the ServiceNow Store to make it available on your instance.

Before you begin
• Purchase an application from the ServiceNow Store.
• Ensure that the application and all of its associated ServiceNow Store applications have valid ServiceNow entitlements. For more information, see Get entitlement for a ServiceNow product or application.

Role required: admin, sn_appclient.app_client_company_installer (can only install applications that match the instance company), or sn_appclient.app_client_user

About this task
Applications that you purchase from the ServiceNow Store are added to the Application Manager automatically. You must install the application from the Application Manager to activate it in an instance.

To install an application that you published to the ServiceNow application repository, follow the steps in Install an application from the application repository instead.

Procedure
1. Navigate to System Applications > All Available Applications > All.
2. Find the application using the filter criteria and search bar.
You can search for the application by its name or ID. If you cannot find an application, you may have to request it from the ServiceNow Store.

Visit the ServiceNow Store website to view all the available apps and for information about submitting requests to the store. For cumulative release notes information for all released apps, see the ServiceNow Store version history release notes.

3. Click **Install**.

4. In the Application installation dialog box, review the application dependencies.

   If dependent plugins and applications will be installed, are currently installed, or need to be installed, they appear in this dialog box. If there are any plugins or applications that need to be installed, you must install them before you can install the ServiceNow Store application.

5. **Optional:** If demo data is available and you want to install it, click **Load demo data.**

   (Optional) Demo data comprises sample records that describe application features for common use cases. Load demo data when you first install the application on a development or test instance.

   **Important:** If you don't load the demo data during installation, it's unavailable to load later.

6. Click **Install**.

7. **Optional:** Click **Go to Subscription Management** to map the installed application to a subscription.

**Related information**

- Install customizations from an application repository
- System-managed developer and deployment roles

**Install an update to a ServiceNow Store application**

Install an update to an application that you purchased from the ServiceNow Store.

**Before you begin**

Role required: admin
Procedure

1. Navigate to System Applications > All Available Applications > All.
2. Find the application with the filter criteria and search bar.
   If they are available, you can select customized versions to be installed.
3. Next to the application listing, select the version to install.
4. Click Update.

⚠️ Note: When customers upgrade to this new version of your application, the Deletes in your author_elective_update folder will be written as Skips in the Upgrade History entry for this application upgrade. Your users can browse the Skip list, and apply the changes on a case-by-case basis. Your customers have the option of setting com.glide.apps.include_my_deletes to false, which will skip writing Skip records as well.

Related information

- Developer and deployment permissions
- System-managed developer and deployment roles

View the upgrade history of a ServiceNow Store application

View the upgrade history of an application that you purchased and installed from the ServiceNow Store.

Before you begin

Role required: admin, sn_appclient.app_client_company_installer (can only install applications that match the instance company), sn_appclient.app_client_user, unified_plugin_read_only

Procedure

1. Navigate to System Applications > All Available Applications > All.
2. Find the application using the filter criteria and search bar.
3. Click the menu icon () on the application tile, and then click **View upgrade history**.

*i Note:* When customers upgrade to this new version of your application, the Deletes in your author_elective_update folder will be written as Skips in the Upgrade History entry for this application upgrade. Your users can browse the Skip list, and apply the changes on a case-by-case basis. Your customers have the option of setting `com.glide.apps.include_my_deletes` to false, which will skip writing Skip records as well.

**Select an application from the application picker**

Use the application picker to select the application to which any new records and updates apply.

**About this task**

Application developers must select an application as their current scope context. Any new records become part of this application. The Now Platform also uses the current application scope to determine if the developer can view or make changes to records in other scope contexts.

**Procedure**

1. Click the gear icon from the banner frame.
2. From **Application**, select the application scope where you want to save changes.
Repair a ServiceNow application

Repair a ServiceNow application by reinstalling it.

Before you begin
Role required: admin

Procedure
1. Navigate to System Applications > All Available Applications > All.
2. From the filter options, select the Installed indicator. All the installed apps are displayed.
3. Click the menu icon ( ) on the application tile, and click Repair.
4. In the Activate Plugin dialog box, click Repair.

Results
The application and its dependencies are reinstalled.

Note:
- When repairing an application with a customization, the customization is repaired as well, unless it’s linked to a source control repository.
- It is assumed the complete set of customizations for an application is within the application customization. When repairing, installing, or upgrading the underlying application or plugin that has an associated app customization, changes made outside of that application customization are overwritten.

With a repair, all the files with sys_update_xml records are added to the remote update set. The remote update set is named with the format: "Repair/Upgrade app <app scope name> at <timestamp>".

Preserve unpublished applications during a system clone

Application developers must manually save a copy of each application currently in development prior to cloning over their development instance.

Before you begin
- Role required: admin
- Write access to the application record
- A source control repository
About this task
The cloning process does not preserve version differences for applications in development. Instead, the system clone only copies the application version installed on the source instance onto the target instance. If the target instance had a development version of the same application, the application will be editable after the clone, but it will be at whatever version was installed on the source instance. If the application was missing from the source instance, the cloning process deletes the application from the target instance.

Procedure
1. Use one of these actions to preserve the application on the clone target instance.

<table>
<thead>
<tr>
<th>Application version state</th>
<th>Action to take</th>
</tr>
</thead>
<tbody>
<tr>
<td>The application version on the clone target instance is different than the source instance version.</td>
<td>Export each application from the clone target instance. Choices include:</td>
</tr>
<tr>
<td>The application is only available on the clone target instance.</td>
<td>• (Recommended) Link each application to a source control repository.</td>
</tr>
<tr>
<td>The application version on the clone target instance is the same as the source instance.</td>
<td>None. The system clone process will copy this application version onto the target instance during the clone.</td>
</tr>
</tbody>
</table>

2. Request a system clone of the source instance over the target instance.

Example
For example, clone your production instance over your development instance.

3. After the clone process finishes, log in to the clone target instance.

4. If you saved each application to a source control repository, use one of these actions to retrieve them from the source control repository.
5. If you saved each application to an update set, use one of these actions to retrieve them from the update set.

**Retrieve applications from a source control repository**

<table>
<thead>
<tr>
<th>Application installation state</th>
<th>Action to take on clone target</th>
</tr>
</thead>
<tbody>
<tr>
<td>The application was previously installed on the source instance.</td>
<td>Apply remote changes from source control repository.</td>
</tr>
<tr>
<td>The application was never installed on the source instance.</td>
<td>Import the application from source control repository.</td>
</tr>
</tbody>
</table>

**Retrieve applications from an update set**

<table>
<thead>
<tr>
<th>Application installation state</th>
<th>Action to take on clone target</th>
</tr>
</thead>
</table>
| The application was previously installed on the source instance. | a. Delete the application version cloned from the source instance.  
  b. Load the update set containing the current application version. |
| The application was never installed on the source instance. | Load the update set containing the current application version. |

**Results**

The applications previously in development are available for further development on the clone target instance.

**Example: Preserve the Marketing Events application**

Suppose your company previously created version 1.0 of a custom application called Marketing Events. You have already published version 1.0 of the Marketing Events application to the application repository and installed it on your production instance.

Over time, users have submitted enhancement requests for the application, and you decide to develop version 2.0 of the Marketing Events application on a non-production instance to address these requests. As development nears completion, you want to update your non-production instance to the latest copy of production for some comprehensive testing.

Since you previously used a source control integration to develop version 1.0 of the Marketing Events application, you have already linked the Marketing Events application to a source control repository. You commit version 2.0 of the Marketing Events application to the source control repository.
You schedule a clone of the production instance over the development instance. After completion, you log in to the development instance and see that it has version 1.0 of the Marketing Events application, because that was the version installed on the source instance.

Since the application was already installed on the source instance, you apply remote changes from the source control repository to receive the latest application version. The development instance now has version 2.0 of the Marketing Events application available for further development and testing.

**Opt out of the ServiceNow Store products**

Opt out of the required product available on ServiceNow Store by navigating to the ServiceNow Products module.

**Before you begin**

- Opted in product
- Role required: admin

**Procedure**

1. Navigate to **System Applications > All Available Applications > ServiceNow Products**. ServiceNow products that are available on ServiceNow Store are displayed. The products are organized based on the product family they are part of.

2. To opt out of the ServiceNow products:
   
   a. Click the required product tile.
   b. Click **Opt-out**.
   c. Enter your Now Support credentials.
   d. Click **Opt-out**.

**Application sharing**

Administrators can share applications that are complete and are ready for use on other instances.

Application developers can share applications using one of the following methods.
### Application sharing methods

<table>
<thead>
<tr>
<th>Sharing method</th>
<th>Makes available to</th>
<th>Typical use case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Publish to the application repository</td>
<td>All instances assigned to the same company</td>
<td>Transfer an application to a test or production environment.</td>
</tr>
<tr>
<td>Publish to the ServiceNow Store</td>
<td>All ServiceNow customers</td>
<td>Share or sell applications to other companies.</td>
</tr>
<tr>
<td>Publish to an Update Set</td>
<td>Any instance with access to the Update Set file</td>
<td>Save a version of an application for compliance or backup reasons.</td>
</tr>
<tr>
<td>Push to team development instances</td>
<td>Other instances in the team development environment</td>
<td>Push developer changes to the parent instance.</td>
</tr>
</tbody>
</table>

**Note:**

“Tracking schema”: Deleting a table or a column in a scoped application is enabled by default for freshly zbooted instances. This is done by having the system property `com.glide.apps.include_my_schema` set to “true”.

For upgraded instances, if you have no custom applications installed or in development, “tracking schema deletes” is enabled by default. Otherwise the property is set to “false” so that customers get the same experience for schema deletes in their applications as in previous releases before Paris. To learn more see the New York and Scoped Applications New Features article on the ServiceNow Community site.

### Custom licensing for ISV applications

For applications that you are sharing, you can create a definition to track usage metrics on your application. For more information, see Custom licensing for ISV applications.

### Publish an application to the application repository

Publish a custom application to the application repository so that it can be installed on other instances in your organization.

**Before you begin**

To allow a developer to publish an application to the application repository, delegate the Publish to App Repo permission to the developer. For more
Delegate development and deployment permissions to personnel.

Role required: admin, or delegated_developer with Publish To App Repo permission enabled

**Procedure**

1. Navigate to **System Applications > My Company Applications**.
2. Open the In Development tab.
3. Open the application record that you want to publish to the application repository.
4. Click the **Publish to My Application Repository** related link.
5. Click **Submit**.

**What to do next**

Install the application on company instances so that your organization can start using it. For more information, see Install an application from the application repository.

By default, after you publish an application to the application repository, all your company instances are entitled to the application automatically. To limit which company instances are entitled to the application, access the application repository by going to [https://apprepo.service-now.com](https://apprepo.service-now.com), and then change the entitlement type for the application.
Note:
The Can Edit Application in Studio property defaults to true for new applications, but you can set it false before publishing.

In Studio, when an application customization has the Can Edit Application in Studio property set to false, the user sees this warning:

For more information, see Manage application entitlements from the application repository and Access ServiceNow Studio.

Publish an application to the ServiceNow Store

Publishing an application to the ServiceNow Store makes it available to everyone.

Before you begin
To publish an application to the ServiceNow Store:

• Create an application within a private application scope.
• Join the Technology Partner Program.
• Have the application certified.

Note: Applications in the global scope cannot be published to the ServiceNow Store.

Role required: admin, or delegated_developer with Publish To App Store permission enabled
About this task
After you have met the prerequisites, you can publish the application to the ServiceNow Store

ℹ️ Note:
The Can Edit Application in Studio property defaults to true for new applications, but you can set it false before publishing.

In Studio, when an application customization has the Can Edit Application in Studio property set to false, the user sees this warning:

Procedure
1. Navigate to System Applications > Applications.
2. Click the In Development tab.
3. Open the application record you want to publish to the ServiceNow Store.
4. Click the Publish to Store related link.
5. Optional: Fill in the fields, as appropriate (see table).
6. Enter your HI credentials.

7. Click **Submit**.
   The system uploads the current version of the application to the ServiceNow Store allowing other users to download it.

**Related information**
Delegate development and deployment permissions to personnel

**Create application files to include sample data**
Include sample records from an application data table when sharing a custom application.

**About this task**
The system can export selected records as application files that are included as part of the application update set when you share it. Including application files in an update set is not intended for the export and import of large numbers of
records between instances. If you are trying to move data between instances, see Import from another instance instead.

The application data only includes the version of the records that existed when the records were shared. The system does not update this snapshot of the application data when the records change. Application designers can include data on a table by table basis.

Procedure

1. Navigate to the list for an application data table.
2. Filter the list to display the records you want to include.
3. Perform the appropriate action for the list version.

<table>
<thead>
<tr>
<th>Version</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>List v2</td>
<td>Open any column context menu and select Create Application Files.</td>
</tr>
<tr>
<td>List v3</td>
<td>Open the list title menu and select Create Application Files.</td>
</tr>
</tbody>
</table>

4. For Load When, select when the application record includes application data.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Install and Upgrades</td>
<td>Includes application data whenever the application is installed or upgraded.</td>
</tr>
<tr>
<td>New Install</td>
<td>Includes application data only when the application is installed.</td>
</tr>
<tr>
<td>New Install with Demo Data</td>
<td>Includes application data only when the application is installed with demo data.</td>
</tr>
</tbody>
</table>

5. Click OK.
The system adds the records to the application files related list.
6. Repeat steps 1–5 for each application data table you want to include.

Publish an application to an Update Set

Publishing an application creates an update set containing the current version of all application configuration records.

Before you begin
Role required: Role required: admin, or delegated_developer with Publish To Update Set permission enabled

About this task
You can use this update set as a backup file for auditing purposes or to transfer the application to another instance.

Procedure
1. Navigate to **System Applications** > **My Company’s Applications**.
2. Click the **In Development** tab.
3. Open the application record you want to create an update set for.
4. Click the **Publish to Update Set** related link.
5. Optional: Fill in the fields, as appropriate (see table).
Publish to Update Set

Publishing an application lets you transfer it to another ServiceNow instance.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application name</td>
<td>[Read-only] Displays the name of the application that you are publishing.</td>
</tr>
<tr>
<td>Version</td>
<td>(Optional) Enter version information to append to the Update Set name in dot-notation such as 1.2.3. The platform saves the information you enter here in the application Version field. The Update Set has the name &lt;Application name&gt; - &lt;Version&gt;. If you leave this field blank, the initial Update Set has the name &lt;Application name&gt; and subsequent Update Sets have the name &lt;Application name&gt; - &lt;Sequential number&gt;.</td>
</tr>
<tr>
<td>Description</td>
<td>Enter a description for the Update Set. By default, this field contains the short description of the application.</td>
</tr>
<tr>
<td>Include data</td>
<td>(Optional) Select the check box to include a limited number of data records from each table in the application. Use this feature to package sample data with your applications.</td>
</tr>
</tbody>
</table>
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Warning:</strong></td>
<td>Using this feature to migrate large quantities of data records between instances can cause performance issues, as it is not intended for this purpose. To migrate data, use an instance-to-instance import. You can adjust the maximum number of data records to include with an application.</td>
</tr>
<tr>
<td></td>
<td>See Import sets.</td>
</tr>
</tbody>
</table>

| Note: | |
|-------| |
| • If your sample data includes tables with record numbering, the current counter value is also transferred. When the update set is applied on another instance, the counter is set to the larger of the sample data or the target instance counter. |
| • For translated fields, only records in English are transferred. |

6. Click **Publish**. 
A new update set is created and the latest update of each application file in the application is copied into it. The update set is marked as complete.

7. Transfer the update set to another instance according to your test process.
   • Retrieve the update set from the source instance.
   • Save the update set as an XML file.

8. Run any fix scripts that are included in the application.

### Related information

- Retrieve an update set
- Save an update set as a local XML file
- Run fix scripts
- Delegate development and deployment permissions to personnel

### Custom licensing for ISV applications

Monitor the usage of ISV applications with Subscription Management. Create a definition for your store application with the metadata you want collected. After publishing the application with the definition to the store, Usage Analytics runs and aggregates your defined metrics.
Overview
Applications that you create with the can be published to the ServiceNow Store. Monitor the usage of your applications with custom licensing.

You can use custom licensing on ISV applications that you publish to the ServiceNow Store if they meet the following criteria:

- The application is licensable.
- The application is not scoped in ServiceNow or Global.
- The application has a capacity subscription model.

Custom licensing functions through the usage analytics (com.glide.usageanalytics) and scoped app author (com.sn_appauthorrr) plugins. These plugins are active by default in the App Engine. Additionally, your instance must be connected to an application repository or application store instance.

Custom license definitions
Custom licensing is achieved by creating a definition. A definition is a set of licensing metrics that you define based on what usage information you want to collect for your application. See Create a definition for your store application.

Create a definition for your store application
Create a definition for your store app to define the licensing metrics you want collected.

Before you begin
Role required: admin or developer

About this task
Create a definition for an application in development or for an application you have already published. However, if you assign a definition to a published application, you will have to re-publish it with the new definition. Additionally, you can only assign one definition to an application.

⚠️ Note: You can only create a definition for capacity subscriptions. See Types of subscriptions.

Procedure
2. Select the application.
3. Under the Subscription Management section, set Subscription Model to Capacity.

4. Select the search icon next to License Definition. The Application License Definitions pop-up loads.

5. Select New.

6. Fill in the Application License Definition record.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the definition.</td>
</tr>
<tr>
<td>Description</td>
<td>Brief summary of the definition.</td>
</tr>
<tr>
<td>Metric Type</td>
<td>Type of collected metric.</td>
</tr>
<tr>
<td>Frequency</td>
<td>How often metrics are collected.</td>
</tr>
<tr>
<td>Performance Validated</td>
<td>Check the box if you have tested the performance of the definition.</td>
</tr>
<tr>
<td>Table</td>
<td>Table used to determine usage.</td>
</tr>
<tr>
<td>Note:</td>
<td>Only lists tables associated with the current application.</td>
</tr>
<tr>
<td>Query</td>
<td>Criteria to use on the table.</td>
</tr>
<tr>
<td>Aggregation</td>
<td>Type of aggregation.</td>
</tr>
<tr>
<td>Note:</td>
<td>Only lists tables associated with the current application.</td>
</tr>
<tr>
<td>Aggregation Column</td>
<td>Table column used for aggregation.</td>
</tr>
<tr>
<td>Note:</td>
<td>Column must be indexed to be aggregated.</td>
</tr>
<tr>
<td>Group By</td>
<td>Columns in the table to group the data by. Max of 3.</td>
</tr>
<tr>
<td>Warning:</td>
<td>Case sensitive.</td>
</tr>
</tbody>
</table>

7. Select Submit. Your new definition appears in the Application License Definitions pop-up.

8. Select the definition to apply it to the application.
What to do next
Create a new version of the application and publish it to the ServiceNow Store. While your application is in review, any changes to the definition may cause subsequent versions of your application to be rejected. Once the application is approved and published to the ServiceNow Store, the definition becomes read-only.

To change the definition of an application already on the ServiceNow Store, you must create a new definition, submit a new version of the application, and go through the review process.

Rolling back and removing applications
As an administrator, you can roll back and remove unwanted applications.

Roll back an application
As an administrator, you can roll back the last installation of a selected application. When you roll back an application, you remove all code, table, and file updates from the initial installation.

Before you begin
- Publish one or more applications to the application repository.
- The selected application must have been installed in the application repository or the application store. See Install a ServiceNow Store application.

Role required: admin

About this task
You can roll back the last installation or upgrade of all applications that are installed from the application store or repository. Rolling back an installation enables you to make needed changes to artifacts associated with an application before you publish and install it again.

Tip: This rollback does not affect the global application record. To remove the global application record, you must instead perform an uninstall of the application. See Uninstall an application.

Procedure
1. Log in to the instance in which you want to roll back the installed application.
3. Click the name or icon of the application that you want to roll back.
4. In the custom application record, click the Rollback related link.

Related information
Uninstall an application

Delete an application
As an administrator, you can delete custom applications that are no longer needed. Typically, you only delete applications that have never been shared with other instances.

Before you begin
• To ensure that you can reinstall the application at a later time, first publish a backup of your application to the ServiceNow application repository, ServiceNow Store, or an update set. See Application sharing.

Role required: admin

About this task
This process ensures that the system can support any published applications. When you delete an application, all the records that are associated with the application, such as tables, business rules, and menus, are also deleted.

Procedure
1. Log in to the instance in which you want to delete the installed application.
2. Navigate to System Applications > Applications.
3. Click the name or icon of the application that you want to delete.
4. In the application record, click Delete.
   A confirmation dialog box appears, indicating if any application files are associated with the application you are deleting.
5. Click the appropriate option.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show Files</td>
<td>Navigate to a listing of files associated with the application record. Review this listing and determine how to remove the files before deleting the application itself.</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>Delete</td>
<td>Delete the selected application.</td>
</tr>
<tr>
<td>Cancel</td>
<td>Cancel deletion of the files associated with the selected application.</td>
</tr>
</tbody>
</table>

6. In the confirmation dialog box, enter delete and click **OK**. Another confirmation dialog box appears, indicating the progress of the deletion. If any remaining metadata files are associated with the application record, the system prevents its deletion until all associated files are removed.

**Results**

Unless the application extends other applications, or files are still associated with the application, the system deletes the application.

**Note:** If an application is linked to Source Control (that is, the Git repository), you must also delete the Git configuration record for an application in the Repository Configurations (sys_repo_config) table. Without deleting this record, the application can’t be re-imported from Source Control on this instance.

**Uninstall an application**

As an administrator, you can uninstall applications that are no longer needed. Typically, you uninstall applications that have been shared with other instances and you want to remove them.

**Before you begin**

**Note:** For global scoped applications, you must first remove the associated files from the application before you delete the application record. These actions help protect the integrity of the instance by preventing the accidental deletion of files in the global scope. See Remove a file from a global application and Move an application file between global applications.

Role required: admin

**About this task**

When you uninstall an application, all application files associated with the application are also removed.
Procedure

1. Log in to the instance on which you want to uninstall an application.
2. Navigate to System Applications > All Available Applications > All.
3. Click the name or icon of the application that you want to uninstall.

   Note: You cannot uninstall applications on the In Development tab, but you can delete them. To learn more about what happens when you perform an application deletion, see Delete an application. You also cannot uninstall ServiceNow-provided system applications.

4. In the custom application record, click the Uninstall related link.
   The following information is displayed in the pop-up window:
   • List of dependent applications.
   • List of in-development applications.
   • Tables and records in the required application.
   • Option to retain the application tables and data.

   Note: If one of the dependent applications is uninstalled after retaining the tables and data, the Uninstalled applications with retained tables list is also displayed. In this case, the Retain tables and data check box is selected by default and can't be cleared.

   Note: When customers upgrade to this new version of your application, the Deletes in your author_elective_update folder will be written as Skips in the Upgrade History entry for this application upgrade. Your users can browse the Skip list, and apply the changes on a case-by-case basis. Your customers have the option of setting com.glide.apps.include_my_deletes to false, which will skip writing Skip records as well.

5. In the Uninstall confirmation dialog box, review the tables and records associated with the application before you uninstall that application.

6. To delete all data associated with this application, clear the Retain tables and data check box. Leave this check box selected to remove only application files.

7. Click OK.

8. In the confirmation dialog box, enter uninstall and click OK.

9. Click Done.
Domain separation and Application Management

Domain separation is unsupported for Application Management. Domain separation enables you to separate data, processes, and administrative tasks into logical groupings called domains. You can control several aspects of this separation, including which users can see and access data.

Support level: No support
- The domain field may exist on data tables but there is no business logic to manage the data.
- This level is not considered domain-separated.

For more information on support levels, see Application support for domain separation.

Application tools

The platform provides several tools to create, manage, and deploy applications.

App Engine Studio

ServiceNow® App Engine Studio is a development tool for creators of varying skill levels to build applications that meet the immediate needs of your organization.

Overview

Here’s an overview on how you can build apps with App Engine Studio. App Engine Studio Overview

Request apps on the Store

Visit the ServiceNow Store website to view all the available apps and for information about submitting requests to the store. For cumulative release notes information for all released apps, see the ServiceNow Store version history release notes.
Get started

<table>
<thead>
<tr>
<th>Explore</th>
<th>Configure</th>
<th>Build</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn about App Engine Studio concepts and features.</td>
<td>Configure environments, tools, and user access.</td>
<td>Build applications using App Engine Studio.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Review</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Review and publish applications that are built in App Engine Studio.</td>
<td>Get details about App Engine Studio components like fields, tables, and properties.</td>
</tr>
</tbody>
</table>

Troubleshoot and get help

- Ask or answer questions in the App Engine Studio community forum
- Search the Known Error Portal for known error articles
- Learn more about how to create your own apps on the developer site.
- Contact Customer Service and Support

Exploring App Engine Studio

Learn about App Engine Studio features that enable you to expand the development of powerful applications for your organization.

App Engine Studio is a guided, low-code tool for developing rich web applications to store information, automate business processes, and solve business problems. This means that you can delegate development work once assigned to administrators to employees with little to no training. By delegating development to business units in your organization, administrators are freed up to address more strategic, system-wide issues.
Overview

• Empower employees in your organization with a guided, intuitive app development environment.

• Build powerful apps to fit any business need by leveraging a suite of development tools, including UI Builder, Catalog Builder, Mobile Studio, and Flow Designer.

• Application development takes place on a non-production instance, your delegated developers can build applications safely without causing technical issues on your production instance.

• Build apps even faster using app templates for pre-built solutions.

• Control which apps to deploy by reviewing and testing the apps that App Engine Studio users submit for approval.

Key features

App Engine Studio applications are built on the Now Platform so users engage with real-time data.

Development tools

App Engine Studio provides instant access to several tools available in the Now Platform development suite. Developers can use any of the following tools to build applications in App Engine Studio:

**UI Builder**

Edit portal and workspace experiences using a highly configurable UI development tool. For more information on using this tool in App Engine Studio, see Editing an experience in App Engine Studio.

**Catalog Builder**

Edit record producer catalog items in a guided tool for building a Service Catalog. For more information on using this tool in App Engine Studio, see Editing an experience in App Engine Studio.

**Mobile Studio**

Edit mobile experiences. For more information on using this tool in App Engine Studio, see Editing an experience in App Engine Studio.

**Table Builder**
Edit data models in a table format. Table Builder is exclusive to App Engine Studio.

For more information on using this tool in App Engine Studio, see Editing data in App Engine Studio.

**Flow Designer**

Edit flows to automate application processes.

For more information on using this tool in App Engine Studio, see Editing logic and automation in App Engine Studio.

**App Engine Studio user interface**

Learn about the App Engine Studio user interface.

App Engine Studio is a web user interface that lets you build custom applications for your organization. You can build applications using templates which can be customized to fit your organizations needs.

**App Engine Studio home**

The App Engine Studio home page provides easy access to creating apps, viewing your app, quick start actions, and templates.

![App Engine Studio Home Page](image)

**Hi, System**

How do you want to get started?

**Quick start**

- Add a table
- Add an experience
- Add an automated flow
- Browse templates
- Learn the tools

**Templates**

- Emergency Alert Template: Accelerate reporting and communications during an emergency
- Team Contacts App Template: Manage employee contact information
- Performance Review Template: Application automates performance-appraisal process
- Time Off Template: Schedule employee vacations and time-off

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Use the top navigation bar to view your apps, access app templates, and find help resources. You can also access the help center by selecting the help icon in the top right.

Users can access the following in App Engine Studio:

- **Create app**— Create an application from scratch.
- **Quick start**— Links to add objects to apps, browse app templates, and learn more about the App Engine Studio tools.
- **My recent apps**— A list of applications you have accessed recently using App Engine Studio.
- **Templates**— A list of templates you can use to create an application with preconfigured data, experience, logic and automation, and security.

**My Apps tab**

View and search the applications you created.

**Templates tab**

Provides a list of available app templates.
Resources tab

View tutorials and helpful resources to get started in App Engine Studio.

App Home

Add data, experience, logic and automation, and security to your app from the app home.
Some objects require access depending on your role. You can request access by selecting **Contact your system administrator** to become a delegated developer. For more information on the delegated developer role, see [Delegated developer role in App Engine Studio](#).

**Application Properties**

View general settings of your application such as the name, description, and logo. You can also open your application in ServiceNow Studio or delete your application.
Repository configuration

View and edit the source control repository settings for your application.

Repository configuration

Edit your integration with your external repository.

Network Protocol

- **https**
- **ssh**

URL

- https://

Branch

- emergency_alert

Connect with a MID server

- **Yes**
- **No**

Default email

- [ ] Always use this email for commits from all developers

Credential

- Srav Github
Experiences that you can create in App Engine Studio
You can add several experiences to your application in App Engine Studio.

Workspace
A workspace is a suite of tools that provides agents, case managers, help desk professionals, and managers with tools to help answer customer questions and resolve customer problems.

App Engine Studio provides a basic workspace that includes a home page, lists, and search.

For more information on workspace features, see Workspace.

The workspace that you create in App Engine Studio includes several default pages. You can use these pages as-is or edit them to suit your business needs.
Search page

3 results for "iphone"

I need a replacement iPhone, please

<table>
<thead>
<tr>
<th>Number</th>
<th>Created</th>
<th>Updated</th>
<th>Owner</th>
<th>Category</th>
<th>Priority</th>
<th>State</th>
<th>Assigned to</th>
<th>Short description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INC001</td>
<td>2020-6-</td>
<td>2021-1-</td>
<td>Fred Lu</td>
<td>IT</td>
<td>5 - Plan</td>
<td>In Preg</td>
<td>ITIL User</td>
<td>Rollback Oracle Version</td>
</tr>
<tr>
<td>INC002</td>
<td>2020-6-</td>
<td>2021-1-</td>
<td>None</td>
<td>IT</td>
<td>1 - Critical</td>
<td>Closed</td>
<td>ITIL User</td>
<td>Switch Sales over to the new 555 prefix</td>
</tr>
<tr>
<td>INC003</td>
<td>2020-6-</td>
<td>2021-1-</td>
<td>None</td>
<td>IT</td>
<td>4 - Low</td>
<td>Review</td>
<td>ITIL User</td>
<td>Roll back Windows SP2 patch</td>
</tr>
<tr>
<td>INC004</td>
<td>2020-6-</td>
<td>2021-1-</td>
<td>None</td>
<td>IT</td>
<td>4 - Low</td>
<td>Implement</td>
<td>David Loo</td>
<td>Upgrade to Oracle 11i</td>
</tr>
<tr>
<td>INC005</td>
<td>2020-6-</td>
<td>2021-1-</td>
<td>None</td>
<td>IT</td>
<td>3 - Moderate</td>
<td>Scheduled</td>
<td>ITIL User</td>
<td>Install new PBX</td>
</tr>
<tr>
<td>INC006</td>
<td>2020-6-</td>
<td>2021-1-</td>
<td>None</td>
<td>IT</td>
<td>4 - Low</td>
<td>Authorize</td>
<td>ITIL User</td>
<td>Put another 100 Gb drive on the 2nd F...</td>
</tr>
<tr>
<td>INC007</td>
<td>2020-6-</td>
<td>2021-1-</td>
<td>None</td>
<td>IT</td>
<td>1 - Critical</td>
<td>Assess</td>
<td>ITIL User</td>
<td>R&amp;D wants to know what it’d cost to s...</td>
</tr>
<tr>
<td>INC008</td>
<td>2020-6-</td>
<td>2021-1-</td>
<td>None</td>
<td>IT</td>
<td>1 - Critical</td>
<td>New</td>
<td>Bow Ruggeri</td>
<td>Install new Cisco</td>
</tr>
<tr>
<td>INC009</td>
<td>2020-6-</td>
<td>2021-1-</td>
<td>None</td>
<td>IT</td>
<td>4 - Low</td>
<td>New</td>
<td>Don Goodlife</td>
<td>Apply patches 10.2.0.1 to 10.2.0.3</td>
</tr>
</tbody>
</table>

Knowledge (2 of 2)

How to configure VPN for Apple Devices

How to configure VPN for Apple Devices. For an iPhone or iPad running iOS 7. Select Settings > General > VPN > Click Add VPN. Enter ServiceNow VPN in the Description field. Enter vpn-xx.vj.server.com in the Server field. Enter your NetID in the Account field. Enter your NetID...

What is a cookie?

What is a cookie? Cookies are messages that web servers pass to your web browser when you visit internet sites. Your browser stores each message in a small file, called cookie.txt. When you request another page from the server, your browser sends the cookie back to the server. These files typically contain information about your visit to the web...

Simple List page

<table>
<thead>
<tr>
<th>Number</th>
<th>Priority</th>
<th>State</th>
<th>Assigned to</th>
<th>Short description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHG0000001</td>
<td>3 - Moderate</td>
<td>New</td>
<td>ITIL User</td>
<td>Rollback Oracle Version</td>
</tr>
<tr>
<td>CHG0000002</td>
<td>4 - Low</td>
<td>Canceled</td>
<td>ITIL User</td>
<td>Switch Sales over to the new 555 prefix</td>
</tr>
<tr>
<td>CHG0000003</td>
<td>1 - Critical</td>
<td>Closed</td>
<td>ITIL User</td>
<td>Roll back Windows SP2 patch</td>
</tr>
<tr>
<td>CHG0000004</td>
<td>4 - Low</td>
<td>Review</td>
<td>ITIL User</td>
<td>Upgrade to Oracle 11i</td>
</tr>
<tr>
<td>CHG0000005</td>
<td>4 - Low</td>
<td>Implement</td>
<td>David Loo</td>
<td>Install new PBX</td>
</tr>
<tr>
<td>CHG0000006</td>
<td>3 - Moderate</td>
<td>Scheduled</td>
<td>ITIL User</td>
<td>Put another 100 Gb drive on the 2nd F...</td>
</tr>
<tr>
<td>CHG0000007</td>
<td>4 - Low</td>
<td>Authorize</td>
<td>ITIL User</td>
<td>R&amp;D wants to know what it’d cost to s...</td>
</tr>
<tr>
<td>CHG0000008</td>
<td>1 - Critical</td>
<td>Assess</td>
<td>ITIL User</td>
<td>Install new Cisco</td>
</tr>
<tr>
<td>CHG0000009</td>
<td>1 - Critical</td>
<td>New</td>
<td>Bow Ruggeri</td>
<td>Apply patches 10.2.0.1 to 10.2.0.3</td>
</tr>
<tr>
<td>CHG0000010</td>
<td>4 - Low</td>
<td>New</td>
<td>Don Goodlife</td>
<td>Java Application Server change</td>
</tr>
</tbody>
</table>
Note: Users could go in to make changes to the Home page, but editing all other pages would entail creating a variant/copy. If you edit these pages directly, then they won’t get the latest code in a system upgrade.

Portal

A portal is a site where users inside of your organization can find information, submit requests, and complete business tasks.

The portal that you create in App Engine Studio includes several default pages. You can use these pages as-is or edit them to suit your business needs.

- Article page
- Catalog item page
- Landing page
- Login
- Order success
- Record page (request)
- Record page (task)
- Search page
- Settings page
- Task approved
- Task rejected

Home page
Note: Unlike Service Portal, the portal that you can create in App Engine Studio is built on the Now® Experience UI Framework. You edit the portal by configuring pages and components in UI Builder.

Catalog item
A catalog item is a form that your users can fill in to create requests or tasks. Users can fill out a form to request an item or service.

For example, an employee can fill in a catalog item form to request time-off for a holiday. After submitting the form, a request is created for a manager to approve or reject.

For more information on catalog items, see Service catalog items.

Mobile
A mobile experience allows users to access your application from a ServiceNow native mobile app.

For example, you can enable a time-off requests application for mobile so that managers can approve or reject requests by opening the Now Mobile app on their mobile device.

For more information on the Now Mobile app, see Now Mobile app.

Delegated development in App Engine Studio
Delegated development enables designated users without a system admin role to develop or deploy applications on the Now Platform. This enables administrators and delegated developers to work together to deliver custom applications through App Engine Studio.

Phase I: Development requested
A team in your organization requests an application. For example, a team may request that their IT department build an application to automate task assignments. An administrator receives this request via a catalog request or email.

Phase II: Development delegation
After an administrator reviews a request, they delegate the development work by providing a link to the developer that opens App Engine Studio in your organization’s development environment. The administrator can:
• Send the link directly to a developer via email
• Post the link on a portal page
• Create a catalog item for delegated developers to request App Engine Studio access

Phase III: Building the application
After receiving access to App Engine Studio, the developer can start building an application using a predefined application template or by creating an application from scratch.

After the developer finishes building the application, they submit their application for approval. For more information on developing in App Engine Studio, see Building applications in App Engine Studio.

Phase IV: Administrator review
After the developer submits their application for approval, an administrator receives a deployment request. They log in to your organization's production environment to review the request.

The deployment request contains detailed information about the application. After reviewing the deployment request, the administrator might send feedback to the developer and then deploy the application to your organization's test environment. The administrator then logs into the test environment and inspects the application. Then:

• If the application is ready to publish, the administrator logs into the production environment and accepts the deployment request.
• If the application is not ready to publish, the administrator logs into the production environment and rejects the deployment request. Optionally, they can send feedback to the developer to inform them why the application is not ready.

Phase V: Application revision
If the administrator rejects the application, the developer receives notification. A developer can use the administrator's feedback to improve their application. After addressing application issues, the developer submits the application again to the administrator.

An application may cycle through administrator review and developer revision several times before the application is ready to publish.
Phase VI: Publishing the application

After accepting a deployment request, the administrator publishes the application by deploying it to production.

The application is now available to users through your organization’s production environment. An administrator can then begin on-boarding the team that requested the application. For example, if the developer specified a certain security role for the application, the administrator assigns that role to the relevant users.

Delegated developer role in App Engine Studio

Delegated development allows designated users without a system admin role to develop and/or deploy applications on the Now Platform. The delegated developer role enables an admin to elevate user permissions to a dedicated role with specific access for App Engine Studio.

A delegated developer has more permissions than a user, but less than an admin. The delegated developers each have their own permissions.

An admin may have several delegated developer each with their own set of permissions. For example, you may want a designated developer for developing a time-off request, and another designed developer with separate permissions for creating support tickets.

As a delegated developer, some features of App Engine Studio may be inaccessible, depending on your permissions. Your role’s permissions enable you to access, edit, and add objects. If a feature of App Engine Studio is unavailable to you, contact your system administrator.

For more information about delegated development, see Delegated development and deployment

Instance strategy for App Engine Studio

Learn how to use multiple ServiceNow instances to implement App Engine Studio in your organization.

One of App Engine Studio’s major security features is the pipeline. A pipeline allows you to quickly move an application across development, test, and production environments. Using separate environments ensures that application development doesn’t cause issues in your production instance, and thus interfere with business operations.

To implement App Engine Studio in your organization, you need one production instance and two non-production instances. One non-production instance serves as the development environment, and the other non-production instance serves as the test environment. The production instance is where the application appears after it has been fully built and tested.
If you have more than two non-production instances, you must decide which instances to use as your development and test environments. You may ask the following questions to make this decision:

- Who is developing the application?
- Which instance does your developer already have access to?
- Has the instance reached or almost reached its table limit?
- Is the instance similarly configured to your production instance?

A non-production instance that is similarly configured to your production instance may be the best candidate for your test environment. An administrator can then more accurately find issues that may arise if the application is deployed to production.

We've provided a new pipeline to move apps around an organization’s instance, which provides admins the confidence that they can roll out citizen developer tools to their organization.

Admins will need to establish their company's instance strategy for App Engine Studio. They need to decide whether they want to allow anyone within their company to have access to App Engine Studio to start building. Admins can also choose to give access to a select group of people. Or, they can choose, on a case-by-case basis, who they give access to by setting up a form where individuals can complete information about the app that they are looking to build and then IT decides whether or not to give those individuals access to build that app.

After they have established their instance strategy, they will also need to establish and automate their approval or review process. AES is a product that runs on your sub-prod environment. For an organization with multiple sub-prod environments, they will need to decide which sub-prod environment App Engine Studio will run on. They will then also need to determine what the pipeline is for apps to be promoted from a particular sub-prod instance to their test instance, and then finally to production where the app will be running live.

**Accessing App Engine Studio for the first time**

App Engine Studio enables a frictionless login experience for developers.

An administrator grants access to App Engine Studio by adding users to the App Engine Studio Users group in your organization’s production instance. The user information is then queued up to synchronize with the instance that your organization is using as its development environment. You can track this queue by navigating to **App Engine Studio > Configuration > User Sync Queue**.
After the system successfully adds a user to the development environment, the user receives an email notification. Email recipients use a temporary password to log in to App Engine Studio for the first time. After resetting their password, a user can start building applications in App Engine Studio.
Only users who are added to the App Engine Studio Users group are synchronized. If the sn_app_eng_studio.user role is assigned directly to the user, then the user isn’t synchronized.

If a user is deleted in your production instance, then the user is deactivated in the development environment and removed from the App Engine Studio Users group.

User synchronization depends on the pipeline to correctly identify your production and development environments. For more information on configuring a pipeline, see Create a pipeline.

Alternatively, you may choose to create users in the development instance only. Add these users to the App Engine Studio Users group in the development instance. Each user then receives an email notification that includes a link to log in to App Engine Studio.

Related reference

- App Engine Studio properties

Related information

- Define environments
- Create a pipeline
- Grant access to App Engine Studio

Configuring App Engine Studio

Plan and configure your App Engine Studio implementation.

Configuring App Engine Studio involves setting up environments, tools, and user access. You perform most of these configuration tasks in each instance that you’re using.

<table>
<thead>
<tr>
<th>Configuration stage</th>
<th>Description</th>
<th>Instance to configure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Install App Engine Studio</td>
<td>You can install the App Engine Studio application (com.snc.app-engine-studio) if you have the admin role. The application installs related ServiceNow® Store applications and plugins if they are not already installed.</td>
<td>• Production</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Development</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Test</td>
</tr>
<tr>
<td>Configuration stage</td>
<td>Description</td>
<td>Instance to configure</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------</td>
<td>----------------------</td>
</tr>
</tbody>
</table>
| Set up instance credentials | For each instance that you're using, create a Connection & Credential alias. Ensure that the alias type is **Credential**, not **Connection and Credential**. Also, ensure that the credentials are configured in the App Engine Studio scope. | • Production  
• Development  
• Test |
| Define environments | Designate which instances to use for development, testing, and deployment. | • Production  
• Development  
• Test |
| Create a pipeline | Create a pipeline so that an administrator can quickly move an application from one environment to another. | • Production  
• Development  
• Test |
| Set up the Microsoft 365 Excel spoke | Allow developers to build applications that can integrate with Microsoft spreadsheets. | • Production  
• Development  
• Test |
| Set up the Google Sheets spoke | Allow developers to build applications that can integrate with Google spreadsheets. | • Production  
• Development  
• Test |
| Set up the Jira spoke | Allow developers to build applications that can integrate with Jira projects. | • Production  
• Development  
• Test |
| Set up the Slack spoke | Allow developers to build applications that can integrate with the Slack platform. | • Production  
• Development  
• Test |
<table>
<thead>
<tr>
<th>Configuration stage</th>
<th>Description</th>
<th>Instance to configure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set up the Microsoft Teams spoke</td>
<td>Allow developers to build applications that can integrate with the Microsoft Teams platform.</td>
<td>• Production</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Development</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Test</td>
</tr>
<tr>
<td>Set up the Gmail spoke</td>
<td>Allow developers to build applications that can integrate with the Gmail platform.</td>
<td>• Production</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Development</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Test</td>
</tr>
<tr>
<td>Set up the Twilio spoke</td>
<td>Allow developers to build applications that can integrate with the Twilio platform.</td>
<td>• Production</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Development</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Test</td>
</tr>
<tr>
<td>Set up the Zoom spoke</td>
<td>Allow developers to build applications that can integrate with the Zoom platform.</td>
<td>• Production</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Development</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Test</td>
</tr>
<tr>
<td>Set up the Twitter spoke</td>
<td>Allow developers to build applications that can integrate with the Twitter platform.</td>
<td>• Production</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Development</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Test</td>
</tr>
<tr>
<td>Review Flow Designer access settings</td>
<td>Review access settings for Flow Designer so that your developers can use the editing capabilities that best suit them.</td>
<td>• Development</td>
</tr>
<tr>
<td>Review Mobile Studio access settings</td>
<td>Review the sn_app_eng_studio/mobile_studio_access system property, which controls whether a developer add a mobile experience to their custom application in App Engine Studio. By default, this property is set to true. To hide this</td>
<td>• Development</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Configuration stage</th>
<th>Description</th>
<th>Instance to configure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Review Catalog Builder access settings</td>
<td>Review access settings for Catalog Builder so that your developers can add items to the appropriate catalogs and categories.</td>
<td>Development</td>
</tr>
<tr>
<td>Set up an instance scan cadence in Health Center</td>
<td>Optionally, schedule regular scans of your instance so that you can identify possible issues that arise from application development in App Engine Studio.</td>
<td>Production, Development, Test</td>
</tr>
<tr>
<td>Set up the administrator contact email</td>
<td>Define the contact email address of the App Engine Studio administrator so that they can receive notifications for deployment requests.</td>
<td>Production, Development, Test</td>
</tr>
<tr>
<td>Grant access to App Engine Studio</td>
<td>Let developers in your organization build applications in App Engine Studio by adding them to the App Engine Studio Users group.</td>
<td>Production</td>
</tr>
<tr>
<td>Create a catalog item for users to request access to App Engine Studio</td>
<td>Optionally, allow users to request App Engine Studio access via Service Catalog.</td>
<td>Production</td>
</tr>
</tbody>
</table>

**Using guided setup to implement App Engine Studio**

Use App Engine Studio guided setup to step through the initial configuration of App Engine Studio.

App Engine Studio guided setup provides a sequence of tasks that help you configure App Engine Studio on your ServiceNow instance. To open App
Engine Studio guided setup, navigate to **App Engine > App Engine Studio > Configuration > Guided Setup.**

Welcome to App Engine Studio Guided Setup

Getting started

**Introduction**

App Engine Studio is a fast, intuitive, low-code visual development environment that empowers developers to build applications to power the digital enterprise. Guided Setup will help you configure App Engine Studio.

**What you’ll do**

- **Configure environments**
  Decide which instances you will use for development, testing, and production.

- **Review and set up tooling**
  Set up App Engine Studio to easily access our available tools for building apps.

- **Set up user access**
  Configure the admin group and other general settings for App Engine Studio users.

**How to prepare for setup**

Setup is required for each instance you use. Keep this list on-hand and be prepared to do this a few times.

<table>
<thead>
<tr>
<th>Spokes to configure (included with App Engine Studio)</th>
<th>Email address of</th>
<th>URLs and Credentials for</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft 365 Excel</td>
<td>App Engine Studio administrator</td>
<td></td>
</tr>
<tr>
<td>Google Sheets</td>
<td>Production instance</td>
<td></td>
</tr>
<tr>
<td>Jira</td>
<td>Test instance</td>
<td></td>
</tr>
<tr>
<td>Slack</td>
<td>Development instance</td>
<td></td>
</tr>
<tr>
<td>Microsoft Teams</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ServiceNow Core</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gmail</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Twilio</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zoom</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Twitter</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The landing page provides information on the different categories, tools, and user access. Select the **Get Started** button in the top, right corner to start your configuration.
The App Engine Studio Guided Setup page provides a list of different categories. Select the **Get Started** button under each category to start configuring App Engine Studio.

For more information on **Guided Setup**, see Using guided setup.

**Set up user access in App Engine Studio**

Configure the admin group and other general settings for App Engine Studio users.

**Before you begin**

Role required: admin

**Procedure**

Navigate to **Application > Module**.

**Install App Engine Studio**

You can install the App Engine Studio application (com.snc.app-engine-studio) if you have the admin role. The application installs related ServiceNow® Store applications and plugins if they are not already installed.
Before you begin

- Ensure that the application and all of its associated ServiceNow Store applications have valid ServiceNow entitlements. For more information, see Get entitlement for a ServiceNow product or application.

Role required: admin

About this task
The following items are installed with App Engine Studio:

- Plugins
- Roles
- Tables

For more information, see Components installed with App Engine Studio.

Procedure

1. Navigate to System Applications > All Available Applications > All.

2. Find the App Engine Studio application (com.snc.app-engine-studio) using the filter criteria and search bar.
   
   You can search for the application by its name or ID. If you cannot find the application, you might have to request it from the ServiceNow Store.

   Visit the ServiceNow Store website to view all the available apps and for information about submitting requests to the store. For cumulative release notes information for all released apps, see the ServiceNow Store version history release notes.

3. In the Application installation dialog box, review the application dependencies.

   Dependent plugins and applications are listed if they will be installed, are currently installed, or need to be installed. If any plugins or applications need to be installed, you must install them before you can install App Engine Studio.

4. Optional: If demo data is available and you want to install it, click Load demo data.

   (Optional) Demo data comprises sample records that describe application features for common use cases. Load demo data when you first install the application on a development or test instance.

   **Important:** If you don’t load the demo data during installation, it’s unavailable to load later.

5. Click Install.
What to do next
Use App Engine Studio guided setup to step through the initial configuration of App Engine Studio. For more information, see Using guided setup to implement App Engine Studio.

Define environments
Designate which instances to use for development, testing, and deployment.

Before you begin
- Install App Engine Studio
- For each instance that you're using, create a Connection & Credential alias. Ensure that the alias type is Credential, not Connection and Credential. Also, ensure that the credentials are configured in the App Engine Studio scope. For more information on creating a credential alias, see Create a Connection and Credential alias.

Role required: admin

About this task
To set up App Engine Studio, you must define the following environments:

Development
Instance that a developer uses to build applications in App Engine Studio.

Test
Instance that an administrator uses to test applications built in App Engine Studio.

Production
Instance that an administrator uses to deploy applications built in App Engine Studio.

The following steps guide you through defining each environment.

Procedure
1. Navigate to App Engine > App Engine Studio > Configuration > Environment, and then select New.
2. On the form, fill in the fields.
Environment form

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the environment. Enter a name that distinguishes the instance as a development, test, or production environment. For example, if you’re defining a development environment, you would include dev in the name.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope of the environment.</td>
</tr>
<tr>
<td>Instance credential</td>
<td>Authentication data related to the instance that you're configuring as an environment. Select a Connection &amp; Credential alias to reference.</td>
</tr>
<tr>
<td>Instance url</td>
<td>Web address of your ServiceNow instance.</td>
</tr>
</tbody>
</table>

3. Select Submit.

4. Repeat the previous steps as you define each remaining environment. For example, if you defined a production environment, repeat the procedure again to define a development environment. Then, repeat the procedure once more to define a test environment.

What to do next
Create a pipeline so that an administrator can quickly move an app from one environment to another. For more information, see Create a pipeline.

Define environments in each instance that you’re using for App Engine Studio. For example, if you defined environments in your production instance, open your development instance and define environments again. Then, do so again in your test instance.

Setting up pipelines
Create a pipeline so that an administrator can quickly move an application from one environment to another.

Set up the administrator contact email
Define the contact email address of the App Engine Studio administrator so that they can receive notifications for deployment requests.

Before you begin
Role required: admin
**Procedure**

1. Navigate to **App Engine > App Engine Studio > Configuration > Properties**.
2. Open the record named **sn_app_eng_studio.aes_admin_contact**.
3. In the **Value** field, enter the email address of the App Engine Studio administrator.
   To enter multiple email addresses, enter a comma-separated list of addresses with no spaces. For example, you would enter multiple addresses as follows: `beth.anglin@example,joe.employee@example.com,abel.tuter@example.com`.
4. Select **Update**.

**Grant access to App Engine Studio**

Let developers in your organization build applications in App Engine Studio by adding them to the App Engine Studio Users group.

**Before you begin**

- Install App Engine Studio
- Define environments
- Create a pipeline

Role required: user_admin or admin

**About this task**

You grant access to App Engine Studio by adding users to the App Engine Studio Users group in your production instance. Each group member is assigned the **sn_app_eng_studio.user** role automatically. Users receive an email notification that includes a link to log in to App Engine Studio.

The user data is synchronized from your production environment to your development environment. User synchronization isn’t supported for non-members who have the **sn_app_eng_studio.user** role. Alternatively, you may choose to create users in the development instance only. Add these users to the App Engine Studio Users group in the development instance.

**Procedure**

1. Log in to your production instance.
2. Navigate to **User Administration > Groups**.
3. Find and open the record named **App Engine Studio Users**.
4. In the Group Members related list, select **Edit**...
5. On the Edit Members page, move each developer from Collection to Group Members List.

6. Select Save.

7. On the group record, select Update.

Results
Users who you add to the App Engine Studio Users group receive an email notification. Email recipients use a temporary password to log in to App Engine Studio for the first time. After resetting their password, a user can start building applications in App Engine Studio.

Note: ServiceNow authentication is not used when App Engine Studio is configured to use single sign-on.
Related reference

Components installed with App Engine Studio

Related information

Accessing App Engine Studio for the first time

Building applications in App Engine Studio

Meet the business needs of your organization by building custom applications in App Engine Studio.

Planning your application

Before you begin building, plan your application. Essentially, an application is a digital program that supports user tasks. You may ask the following questions as you plan:

• What are the goals, objectives, and outputs of your application?
• Who uses your application?
• Who has access to what?
• What tasks do people complete with your application?
• Where does the data come from?
• How do people interact with your application?
• What processes must the application support?
Your answers to these questions can help you decide whether to use an application template.

An application template provides predefined content to support a certain use case. For example, the Travel Request template provides application content for submitting and approving employee travel requests.

For a more streamlined development process, you would use the template that most closely fits your application goals. For more information on the available templates, see Template library.

If the available templates don’t fit your application goals, then you would create an application from scratch.

**Building your application**

To build an application in App Engine Studio, add the following content:

**Data**
Information that is stored in your application. For example, employee phone numbers or office locations. You configure application data using tables.

**Experience**
Graphical interface that your users interact with. For example, you can create a portal where users find information, submit requests, or complete business tasks.

**Logic and automation**
Business logic to automate application processes. You define logic and automation using flows. A flow includes a sequence of actions and a trigger. You can use a flow template or create a flow from scratch.

**Security**
Roles and access controls to limit who can use your application.

If you use an application template, the template automatically adds one or more of these items to your application. You can then edit them to tailor to your specific business needs. You can also add more data, experience, logic and automation, or security to an application that you created using a template.

To build an application from scratch, add data, experience, logic and automation, and security to your application and then edit each.

**Submitting your application**

After you’ve finished building your application, you start the process of getting the application published by submitting it for administrator approval.
The administrator reviews the submitted application and checks for potential issues. The administrator then rejects or approves the application.

You can check the status of your submitted application from the application dashboard in App Engine Studio.

"Pending Approval" status

If the administrator rejects the application, then it’s up to you to address the issues that the administrator discovered. You can use administrator feedback to guide you as you improve your application. After you’ve updated your application, re-submit it for administrator review.

After the administrator approves the application, they publish the application and on-board the relevant users.

For more information on submitting your application, see Submit an application for approval.

For more information on the application review process, see Delegated development in App Engine Studio.

Considerations
There are several considerations to keep in mind as you build applications in App Engine Studio:

• You access App Engine Studio by navigating to App Engine Studio or with the following web address: <instance-name>.servicenow.com/now/appenginestudio. If you don’t have a working link, contact your administrator.

• When you navigate back and forth between different applications in App Engine Studio, the application scope changes accordingly.

Intake request process for application development ideas
Application developers can easily submit application ideas through intake request forms. After ideas are submitted, the requests are automatically assigned to a designated group for review and approval.

After a request is approved or denied, an email notification is sent to the requester. If the request was approved, the developer is granted access to App Engine Studio to enable app development.
Benefits

The intake request process provides the following benefits:

• Streamlined process to submit ideas for app development.

• Automatic workflow process sends ideas to specified approvers or approval groups to accept or reject. Notification email messages are automatically sent once an idea is accepted or rejected.

• By exporting requests to a spreadsheet, approvers can easily check for duplicate or previously rejected requests.

Related concepts

• Refine the intake request form by modifying the related Service Catalog items. For more information, see Service catalog items

• Customize the approval flow. For more information, see Flows.

Submit an intake request form

Submit an idea for app development in your organization using an intake request form.

Before you begin

Role required: None

Procedure

1. Navigate to Service Catalog > Catalog.

2. Select Can We Help You?.

3. Select the Apply for Citizen Development catalog item.

4. On the form, fill in the fields.

Citizen Development form

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Name</td>
<td>Name for the application.</td>
</tr>
<tr>
<td>Describe your idea in a few sentences</td>
<td>Brief overview of the app development idea.</td>
</tr>
<tr>
<td>Is your process repeatable?</td>
<td>Option to indicate whether the app development process can be repeated.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>Do you have an email or excel based process?</td>
<td>Indicates whether the app development process is based on email or a Microsoft Excel spreadsheet.</td>
</tr>
<tr>
<td>How many users are involved in this?</td>
<td>Number of developers involved in developing or implementing the app development idea.</td>
</tr>
<tr>
<td>Does this involve any sensitive/PII data?</td>
<td>Indicates whether the app development idea requires access to sensitive customer information.</td>
</tr>
<tr>
<td>Do you need data from other departments?</td>
<td>Indicates whether the app developers require data from other departments.</td>
</tr>
<tr>
<td>Who are the users that will have access to the data in this application?</td>
<td>Lists groups or departments that require access to the app.</td>
</tr>
</tbody>
</table>

5. Click **Submit**.

Results
Your intake request is submitted to a designated group for approval. You will receive an email notification after a decision is made.

Review an intake request
After an idea for app development is submitted, review the intake request to approve or reject it.

Before you begin
Role required: admin
Procedure

1. Access the approval request record in one of the following ways.

<table>
<thead>
<tr>
<th>Option</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServiceNow instance</td>
<td>a. Navigate to Self-Service &gt; My Approvals.</td>
</tr>
<tr>
<td></td>
<td>b. Select the approval record that you want to review.</td>
</tr>
<tr>
<td></td>
<td>c. View the details by clicking Requested Items.</td>
</tr>
<tr>
<td>Email</td>
<td>Click the link in the Approval Request email you receive.</td>
</tr>
</tbody>
</table>

2. On the approval record, review the intake request and either approve or reject it.

<table>
<thead>
<tr>
<th>Option</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>The request is of value and you think it should be implemented</td>
<td>Click Approve.</td>
</tr>
<tr>
<td>The request is not viable. Possible reasons include:</td>
<td></td>
</tr>
<tr>
<td>• It has already been implemented.</td>
<td></td>
</tr>
<tr>
<td>• It's too difficult to implement.</td>
<td></td>
</tr>
<tr>
<td>• It's a duplicate.</td>
<td></td>
</tr>
<tr>
<td>• It has previously been rejected.</td>
<td></td>
</tr>
</tbody>
</table>

Results
A notification email is sent to the requester indicating the approval decision.

Use an application template
Build an application that uses predefined data, experience, logic and automation, and security.

Procedure

1. From the Template page, select a template.
2. On the template detail page, select Use Template.
3. Enter a name and description for your application.

4. **Optional:** Add a logo by dragging an image or browsing your computer.

5. Select **Continue**.

6. On the summary screen, select **Go to app home**.

7. From the app home, review the data, experience, logic and automation, or security that was created with the template. To tailor the application to your business needs, you can edit these predefined items or add your own. For more information, see the following topics:
   - Editing data in App Engine Studio
   - Adding data in App Engine Studio
   - Editing an experience in App Engine Studio
   - Adding an experience in App Engine Studio
   - Editing logic and automation in App Engine Studio
   - Adding logic and automation in App Engine Studio
   - Editing security in App Engine Studio
   - Adding security in App Engine Studio

**What to do next**

After you've finished building your application, submit the application for approval to get it reviewed and deployed by an administrator. For more information, see **Submit an application for approval**.

**Create an application from scratch for App Engine Studio**

If the available application templates don't fit your application goal, create an empty application to which you can add data, experience, logic and automation, and security.

**Before you begin**

Role required: admin, sn_app_engine_studio.user

**Procedure**

1. From the My Apps page, select **Create app**.

2. Enter a name and description for your application.

3. **Optional:** Add a logo by dragging an image or browsing your computer.

4. Select **Continue**.

5. On the summary screen, select **Go to app home**.
6. From the app home, add data, experience, logic and automation, and security to your app.
For more information, see the following topics:

- Adding data in App Engine Studio
- Adding an experience in App Engine Studio
- Adding logic and automation in App Engine Studio
- Adding security in App Engine Studio

What to do next
After you've finished building your application, submit the application for approval to get it reviewed and deployed by an administrator. For more information, see Submit an application for approval.

Adding data in App Engine Studio
Store information in an application by uploading a spreadsheet, extending a table, or building a custom table.

Data is a key starting point for any new application. Data populates a user interface. People who use your application may provide data, such as their name and phone number, when they fill in a form. Other people may refer to data as they fulfill a request.

In App Engine Studio, you store application data in a table format. Organizing data in a table makes it easier to track. When users update your application data, they create a row or change an existing row on your table.

The data that you use in your application may come from several sources. You may want to use data that you’re already storing in a spreadsheet. Or, you may want to use data that your organization is storing on the Now Platform.

Use any of the following methods to add a data table to your application:

- Upload a spreadsheet that has one sheet
- Upload a spreadsheet that has multiple sheets
- Create a table from an existing table
- Create a table from scratch

Application templates automatically add data to your application. If you use a template to create your application, you can edit the tables that were added or add different tables.
Upload a spreadsheet that has one sheet

Add a table to your application by uploading a Microsoft Excel spreadsheet that has one sheet.

Before you begin
Check that the spreadsheet meets the following requirements:
• Formatted with horizontal columns and a header label for each
• Saved as an XLSX file type
• Has only one sheet

Procedure
1. From the My Apps page, open your application.
2. In your application, next to Data, select the add icon (➕).
3. Select Upload a spreadsheet and then select Continue.
4. Upload your spreadsheet.
   You can drag the spreadsheet file onto App Engine Studio or browse your computer.
5. In the Enter a row number for the table header field, enter the row number in which column headers are located.
   For example, if the headers are in the second row, enter 2.
6. Optional: To upload the information below the column headers in your spreadsheet, select the Import spreadsheet data field.
   To upload only the column headers, leave the field blank.
7. Select Continue to review the data parsed from your spreadsheet.
8. Review the properties for each column header.

<table>
<thead>
<tr>
<th>Column header property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field Label</td>
<td>Unique label for the column.</td>
</tr>
<tr>
<td>Field name</td>
<td>Database name for the column.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of information that the column contains. For example, to contain plain text in the column, select String. Depending on the type that you select, fill in the additional fields to further define the table column. For</td>
</tr>
<tr>
<td>Column header property</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>example, if you select <strong>String</strong>, define the character limit of the string input. Or, if you select <strong>Choice</strong>, define the choices that users can choose from.</td>
<td></td>
</tr>
<tr>
<td>Character limit</td>
<td>Max length of the strings that can be stored.</td>
</tr>
<tr>
<td>Display</td>
<td>Option to set the column as the display value for the table. A reference field shows the display value of the table to which it is referring. For example, the <strong>Opened by</strong> column of the task table refers to the user table. Because the display value of the user table is the user name, the <strong>Opened by</strong> field shows something like <strong>Beth Anglin</strong> or <strong>Joe Employee</strong>. When you select a display value, choose the table column that would act as an appropriate title for individual records. Only one column can act as the display value for a table.</td>
</tr>
<tr>
<td>Mandatory</td>
<td>Option to require that the column must contain a value before a new record can be saved.</td>
</tr>
</tbody>
</table>

To add another column to your table, select **Add new field**.

To delete a column, select the trash icon (🗑️).

9. Select **Continue** to define table properties.

10. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table label</td>
<td>Unique label to identify the table.</td>
</tr>
<tr>
<td>Table name</td>
<td>Database name for the table. A table name is created automatically after you enter a table label. You can edit the name if needed.</td>
</tr>
<tr>
<td>Make extensible</td>
<td>Option to allow other tables to share data from this table. For more information on table extension, see Table extension and classes.</td>
</tr>
<tr>
<td>Auto number</td>
<td>Option to track table records with a unique number. If you select this option, define the <strong>Prefix</strong>, <strong>Starting number</strong>, and <strong>Number of digits</strong>.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Prefix</td>
<td>Abbreviated name of the table to append to the beginning of the record number. For example, if you are creating a &quot;Laptop&quot; table, then your prefix may be &quot;LPTP&quot; or &quot;LT.&quot;</td>
</tr>
<tr>
<td>Starting number</td>
<td>Number to identify the first record created for your table.</td>
</tr>
<tr>
<td>Number of digits</td>
<td>Maximum number of digits to allow in the record number. This value determines the highest possible record number. For example, if you enter 7, then the highest possible number is 9999999.</td>
</tr>
</tbody>
</table>

11. Select **Continue**, and then select **Done** on the summary screen.
   Alternatively, to view your table, you can select **Edit table** on the summary screen.

**Upload a spreadsheet that has multiple sheets**

If your Microsoft Excel spreadsheet has multiple sheets, you can upload each sheet to add multiple tables to your application.

**Before you begin**

Check that the spreadsheet meets the following requirements:

- Formatted with horizontal columns and a header label for each
- Saved as an XLSX file type
- Has more than one sheet

**Procedure**

1. From the My Apps page, open your application.

2. In your application, next to Data, select the add icon (⊕).

3. Select **Upload a spreadsheet** and then select **Continue**.

4. Upload your spreadsheet and then select **Continue**.
   You can drag the spreadsheet file onto App Engine Studio or browse your computer.

5. From the list of sheets, select a sheet to add as a table to your application.

   a. In the **Enter a row number for the table header** field, enter the row number in which the sheet column headers are located.
      For example, if the headers are in the second row, enter 2.
b. **Optional:** To upload the information below the column headers in the sheet, select the **Import spreadsheet data** field. To upload only the column headers, leave the field blank.

c. Select **Convert to table**.

6. Review the properties for each column header.

<table>
<thead>
<tr>
<th>Column header property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field Label</td>
<td>Unique label for the column.</td>
</tr>
<tr>
<td>Field name</td>
<td>Database name for the column.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of information that the column contains. For example, to contain plain text in the column, select <strong>String</strong>. Depending on the type that you select, fill in the additional fields to further define the table column. For example, if you select <strong>String</strong>, define the character limit of the string input. Or, if you select <strong>Choice</strong>, define the choices that users can choose from.</td>
</tr>
<tr>
<td>Character limit</td>
<td>Max length of the strings that can be stored.</td>
</tr>
<tr>
<td>Display</td>
<td>Option to set the column as the display value for the table. A reference field shows the display value of the table to which it is referring. For example, the <strong>Opened by</strong> column of the task table refers to the user table. Because the display value of the user table is the user name, the <strong>Opened by</strong> field shows something like <strong>Beth Anglin</strong> or <strong>Joe Employee</strong>. When you select a display value, choose the table column that would act as an appropriate title for individual records. Only one column can act as the display value for a table.</td>
</tr>
<tr>
<td>Mandatory</td>
<td>Option to require that the column must contain a value before a new record can be saved.</td>
</tr>
</tbody>
</table>

To add another column to your table, select **Add new field**.

To delete a column, select the trash icon (łuż).
7. Select **Continue** to define table properties.
8. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table label</td>
<td>Unique label to identify the table.</td>
</tr>
<tr>
<td>Table name</td>
<td>Database name for the table. A table name is created automatically after you enter a table label. You can edit the name if needed.</td>
</tr>
<tr>
<td>Make extensible</td>
<td>Option to allow other tables to share data from this table. For more information on table extension, see Table extension and classes.</td>
</tr>
<tr>
<td>Auto number</td>
<td>Option to track table records with a unique number. If you select this option, define the <strong>Prefix</strong>, <strong>Starting number</strong>, and <strong>Number of digits</strong>.</td>
</tr>
<tr>
<td>Prefix</td>
<td>Abbreviated name of the table to append to the beginning of the record number. For example, if you are creating a &quot;Laptop&quot; table, then your prefix may be &quot;LPTP&quot; or &quot;LT.&quot;</td>
</tr>
<tr>
<td>Starting number</td>
<td>Number to identify the first record created for your table.</td>
</tr>
<tr>
<td>Number of digits</td>
<td>Maximum number of digits to allow in the record number. This value determines the highest possible record number. For example, if you enter 7, then the highest possible number is 9999999.</td>
</tr>
</tbody>
</table>

9. For each sheet that you want to upload, repeat steps 5-8.

10. After you’ve uploaded your sheets, select **Done**.

    Alternatively, to view your tables, you can select **Edit Tables** on the summary screen.

**Create a table from an existing table**

Add a data table to your application by extending a table that you or someone else previously created.

**About this task**

There are several tables that your organization is already using in other applications. For example, your organization may be using the Incident [incident] table for its Incident Management application. You can create a table for your application by extending any of these existing tables.
Table extension means enabling a table to share fields and records with a parent table. You would extend a table if you expect users to fill in similar fields as an existing table. For example, to create a ticketing type of table, you would extend the Task [task] table. The Task table includes fields that are standard for most work tickets, such as **Number** and **Assigned to**.

After extending an existing table, you can further customize the new table by adding more columns.

For more information on extending tables, see **Table extension and classes**.

**Procedure**

1. From the My Apps page, open your application.
2. In your application, next to Data, select the add icon (➕).
3. Select **Create from an existing table**, and then select **Continue**.
4. Select a table to extend, and then select **Continue**.
5. On the form, fill in the fields and then select **Continue**.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table label</td>
<td>Unique label to identify the table.</td>
</tr>
<tr>
<td>Table name</td>
<td>Database name for the table. A table name is created automatically after you enter a table label. You can edit the name if needed.</td>
</tr>
<tr>
<td>Make extensible</td>
<td>Option to allow other tables to share data from this table. For more information on table extension, see <strong>Table extension and classes</strong>.</td>
</tr>
<tr>
<td>Auto number</td>
<td>Option to track table records with a unique number. If you select this option, define the <strong>Prefix</strong>, <strong>Starting number</strong>, and <strong>Number of digits</strong>.</td>
</tr>
<tr>
<td>Prefix</td>
<td>Abbreviated name of the table to append to the beginning of the record number. For example, if you are creating a &quot;Laptop&quot; table, then your prefix may be &quot;LPTP&quot; or &quot;LT.&quot;</td>
</tr>
<tr>
<td>Starting number</td>
<td>Number to identify the first record created for your table.</td>
</tr>
<tr>
<td>Number of digits</td>
<td>Maximum number of digits to allow in the record number. This value determines the highest possible record number. For example, if you enter 7, then the highest possible number is 9999999.</td>
</tr>
</tbody>
</table>
6. To review the columns that were added to your table, select **Edit table**.

7. In the new tab that opens, review the table columns.
   
   Each table column appears as a row in Table Builder.

   A lock icon (🔒) appears next to columns from the extended table. You can’t modify these columns.

8. Customize your table by adding table columns.

   a. In Table Builder, select **+ Add new field**.

   b. On the blank row, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Column label</td>
<td>Unique label for the column.</td>
</tr>
<tr>
<td>Column name</td>
<td>Database name for the column.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of information that the column contains. For example, to contain plain text in the column, select <strong>String</strong>. Depending on the type that you select, fill in the additional fields to further define the table column. For example, if you select <strong>String</strong>, define the character limit of the string input. Or, if you select <strong>Choice</strong>, define the choices that users can choose from.</td>
</tr>
<tr>
<td>Reference</td>
<td>Table that is associated with the column. This field applies only if the column type is <strong>Reference</strong>.</td>
</tr>
<tr>
<td>Max length</td>
<td>Maximum number of characters that users can enter in the field.</td>
</tr>
<tr>
<td>Default value</td>
<td>Value that populates the field automatically after a new record is created.</td>
</tr>
<tr>
<td>Display</td>
<td>Option to set the column as the display value for the table. A reference field shows the display value of the table to which it is referring. For example, the <strong>Opened by</strong> column of the task table refers to the user table. Because the display value of the user table is the user name, the <strong>Opened by</strong> field shows something like <strong>Beth Anglin</strong> or <strong>Joe Employee</strong>. When you select a display...</td>
</tr>
</tbody>
</table>
Create a table from scratch

If there are no spreadsheets or existing tables to use for your application, you can create a table that is completely custom.

Procedure

1. From the My Apps page, open your application.
2. In your application, next to Data, select the add icon (+).
3. Select Create from scratch, and then select Continue.
4. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table label</td>
<td>Unique label to identify the table.</td>
</tr>
<tr>
<td>Table name</td>
<td>Database name for the table. A table name is created automatically after you enter a table label. You can edit the name if needed.</td>
</tr>
<tr>
<td>Make extensible</td>
<td>Option to allow other tables to share data from this table. For more information on table extension, see Table extension and classes.</td>
</tr>
<tr>
<td>Auto number</td>
<td>Option to track table records with a unique number. If you select this option, define the Prefix, Starting number, and Number of digits.</td>
</tr>
<tr>
<td>Prefix</td>
<td>Abbreviated name of the table to append to the beginning of the record number. For example, if you are creating a “Laptop” table, then your prefix may be “LPTP” or “LT.”</td>
</tr>
<tr>
<td>Starting number</td>
<td>Number to identify the first record created for your table.</td>
</tr>
<tr>
<td>Number of digits</td>
<td>Maximum number of digits to allow in the record number. This value determines the highest possible</td>
</tr>
</tbody>
</table>
5. To begin building your table, select **Edit table**.

Table Builder opens in a new tab.

Your new table includes several default columns, including **Created** and **Updated**. You can’t delete these default columns.

6. Customize your table by adding table columns.

   a. In Table Builder, select **+ Add new field**.

   b. On the blank row, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Column label</td>
<td>Unique label for the column.</td>
</tr>
<tr>
<td>Column name</td>
<td>Database name for the column.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of information that the column contains. For example, to contain plain text in the column, select <strong>String</strong>. Depending on the type that you select, fill in the additional fields to further define the table column. For example, if you select <strong>String</strong>, define the character limit of the string input. Or, if you select <strong>Choice</strong>, define the choices that users can choose from.</td>
</tr>
<tr>
<td>Reference</td>
<td>Table that is associated with the column. This field applies only if the column type is <strong>Reference</strong>.</td>
</tr>
<tr>
<td>Max length</td>
<td>Maximum number of characters that users can enter in the field.</td>
</tr>
<tr>
<td>Default value</td>
<td>Value that populates the field automatically after a new record is created.</td>
</tr>
<tr>
<td>Display</td>
<td>Option to set the column as the display value for the table. A reference field shows the display value of the table to which it is referring. For example, the <strong>Opened by</strong> column of the task table refers to the user table. Because the display value of the user table is the user name, the <strong>Opened by</strong> field shows something like <strong>Beth</strong>.</td>
</tr>
</tbody>
</table>
Adding an experience in App Engine Studio

Create a portal, workspace, record producer, or mobile experience for users to interact with your app.

Application templates automatically add experiences to your application.

Add a workspace

Give your users a service desk-like experience to manage tickets.

Procedure

1. From the My Apps page, open your application.
2. In your application, next to Experience, select the add icon (+).
3. Select Workspace, and then select Begin.
4. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the workspace. By default, the workspace shares the same name as your application.</td>
</tr>
<tr>
<td></td>
<td><strong>Tip:</strong> Use a name that uniquely identifies the workspace from other experiences. For example, Office art requests workspace.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the workspace.</td>
</tr>
<tr>
<td>URL</td>
<td>Web address of the workspace. By default, the workspace URL is based on the application name.</td>
</tr>
<tr>
<td>Roles</td>
<td>User roles to limit who can access the workspace. To use a custom role for your workspace, you must create one in Security first. For more information, see Adding security in App Engine Studio.</td>
</tr>
</tbody>
</table>

5. Select Continue to define data for your workspace.
6. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary Table</td>
<td>Data table that workspace users view or update.</td>
</tr>
<tr>
<td>Secondary Tables</td>
<td>Additional data tables that workspace users view and update.</td>
</tr>
</tbody>
</table>

7. Select **Continue**, and then select **Done**.

**What to do next**
Customize the workspace to your organization's needs by editing in UI Builder.
For more information, see [Editing data in App Engine Studio](#).

**Add a portal**
Give your users a site where they can find information, create requests, and complete business tasks.

**Procedure**
1. From the My Apps page, open your application.
2. In your application, next to Experience, select the add icon (➕).
3. Select **Portal**, and then select **Begin**.
4. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the portal. By default, the portal shares the same name as your application.</td>
</tr>
<tr>
<td></td>
<td>☛ <strong>Tip:</strong> Use a name that uniquely identifies the portal from other experiences. For example, Office art portal.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the portal.</td>
</tr>
<tr>
<td>URL</td>
<td>Web address of the portal. By default, the workspace URL is based on the application name.</td>
</tr>
<tr>
<td>Roles</td>
<td>User roles to limit who can access the portal. To use a custom role for your portal, you must create one in</td>
</tr>
</tbody>
</table>
5. Select **Continue**, and then select **Done**.

**What to do next**
Customize the portal to your organization's needs by editing in UI Builder. For more information, see **Editing data in App Engine Studio**.

**Add a mobile experience**
Give your users the ability to perform application tasks from their mobile device.

**Procedure**
1. From the My Apps page, open your application.
2. In your application, next to Experience, select the add icon (➕).
3. Select **Mobile**, and then select **Begin**.
4. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the mobile experience. By default, the mobile experience shares the same name as your application.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of your mobile experience.</td>
</tr>
<tr>
<td>Tables</td>
<td>Data tables that mobile users view and update.</td>
</tr>
<tr>
<td>Roles</td>
<td>User roles to limit who can access the mobile experience. To use a custom role for your mobile experience, you must create one in <strong>Security</strong> first. For more information, see <strong>Adding security in App Engine Studio</strong>.</td>
</tr>
<tr>
<td>Available offline</td>
<td>Option to enable mobile users work on data without an internet connection. Updates that users make while offline are synced to the application after they connect to the internet again.</td>
</tr>
</tbody>
</table>

5. Select **Continue**, and then select **Done**.

**What to do next**
Customize the mobile experience to your organization's needs by editing in Mobile Studio. For more information, see **Editing data in App Engine Studio**.
Add a catalog item

Give your users a way to submit a request by filling out a form.

Before you begin

Create a data table that you plan to update using the request form. For more information, see Adding data in App Engine Studio.

Procedure

1. From the My Apps page, open your application.
2. In your application, next to Experience, select the add icon (+).
3. Select Catalog item, and then select Begin.
4. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the catalog item.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the catalog item.</td>
</tr>
</tbody>
</table>

5. Select Edit catalog item to open Catalog Builder.
6. Specify the required information.

Catalog Builder steps

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Details</td>
<td>• Basic info: Item name and short description.</td>
</tr>
<tr>
<td></td>
<td>• Item details: Description of the item.</td>
</tr>
<tr>
<td>Destination</td>
<td>Destination table in which a record producer creates records.</td>
</tr>
<tr>
<td>Location</td>
<td>Catalogs and categories that the item can belong to.</td>
</tr>
<tr>
<td></td>
<td>Note: You can add up to a maximum of 50 catalogs or categories.</td>
</tr>
<tr>
<td>Questions</td>
<td>Question sets (variable sets) or questions for the item. By default, the question sets included in the template are added and cannot be removed. Available question types are restricted to those specified in the template.</td>
</tr>
<tr>
<td>Step</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>From the <strong>Insert new question</strong> list, you can include additional question sets, questions, deactivated questions, single-column container, two-column container, and line break. For information on creating a question and supported question types, see <em>Create a question for a catalog item in Catalog Builder</em>. You can also add dynamic form behavior for a question. For information on adding dynamic form behavior, see <em>Edit a question in Catalog Builder</em>.</td>
</tr>
</tbody>
</table>

**Note:**
- The questions within a question set cannot be edited. The question sets can be reordered by dragging and dropping them.
- Question sets specified in the template cannot be removed, while the ones added by the user can be removed.
- Removal of question sets simply removes the association with the item and does not delete the question set.
- A single-column question set can be added to a single-column or a two-column container. A two-column question set cannot be added to a container.

<table>
<thead>
<tr>
<th>Settings</th>
<th>Settings to configure the catalog item behavior in Service Portal based on the request method.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>User criteria to specify users for whom the item is available or not available.</td>
</tr>
<tr>
<td>Review and submit</td>
<td>Review the item and submit it.</td>
</tr>
</tbody>
</table>

To save your changes in each step, select **Save**.

7. To preview a catalog item in Portal or Now Mobile, select **Preview**.

**Note:** When you preview an item, you can interact with it but not submit it.
a. To preview the item in Portal, select Portal for the View within field.

   Note: Portal preview is based on portal URL configured for the catalog builder.

b. To view an item representation in Now Mobile, select Now Mobile for the View within field.

c. To open the preview in a new tab, select Open preview in a new tab.

   Note: When you make any changes to the item and save it, the preview is dynamically refreshed in the opened tab.

8. Select Submit.

Adding logic and automation in App Engine Studio

Replace manual processes with application logic and automation.

In App Engine Studio, you define logic and automation by using flows. A flow is a sequence of actions that run when the flow starts. App Engine Studio provides pre-built flows you can apply to common use cases, or you can create your own custom flows. You can automatically start flows by defining a trigger or manually start flows from an API call. For example, you can create a flow that is triggered when a user fills in a request form. Then, you can define the flow to send a notification to another user who fulfills the request.

Application templates automatically add logic and automation to your application.

To add automation to your application, you can use a flow template or create a flow from scratch.

Use an automation template

Use a pre-built flow to automate application processes.

Procedure

1. From the My Apps page, open your application.

2. In your application, next to automation, select the add icon (➕).

3. From the gallery of automation templates, select a template and then select Begin.
Create an approval for a requested catalog item

This flow template creates an approval flow for a requested item from the Service Catalog. This flow automatically approves any item with a price lower than a specific amount. During configuration, you specify the automatic approval price amount.

4. Complete the wizard for the template that you selected.
5. On the summary screen, select Done.

What to do next
You can edit the flow to tailor it to your specific business needs. For more information on editing flows, see Editing logic and automation in App Engine Studio.

Create a flow from scratch
If an existing automation template doesn't fit your application goal, define custom automation by building a flow from scratch.

Procedure
1. From the My Apps page, open your application.
2. In your application, next to automation, select the add icon (➕).
3. Select Build from scratch.
4. On the form, fill in the fields and then select **Continue**.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name to uniquely identify your flow.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of your flow.</td>
</tr>
<tr>
<td>Protection</td>
<td>Access settings for the flow. You can select one of the following options:</td>
</tr>
<tr>
<td></td>
<td><strong>-- None --</strong></td>
</tr>
<tr>
<td></td>
<td>Option to allow other users to edit the flow.</td>
</tr>
<tr>
<td></td>
<td><strong>Read-only</strong></td>
</tr>
<tr>
<td></td>
<td>Option to allow other users to view the flow but not edit it.</td>
</tr>
</tbody>
</table>
| Run As      | Selection to specify if the flow runs as a system user or the user who initiates the session. If updates should come from the user who triggered the flow, select **User who initiates session**. For example, when you want incident
### Field Description

- **Run with roles**
  - Roles that the flow runs with. This option is available only when **Run As** is set to **User who initiates session**.
  - To use a custom role for your flow, you must create one in **Security** first. For more information, see Adding security in App Engine Studio.

5. On the summary screen, select **Edit this flow** to open Flow Designer.

6. Add a trigger to your flow.

   a. Under the **TRIGGER** section, select **Add a trigger**.

   b. From the Trigger list, select a trigger that will start running your flow.
      - For more information on trigger types, see Flow trigger types.
      - The system displays a set of fields depending on the type of trigger that you’ve selected.

   c. Set up your trigger by filling in the fields.
      - For a record-based trigger, for example, select a table and optionally set field conditions that, when met, will start running your flow.

   d. Click **Done**.
7. To add actions, flows, subflows, or flow logic, click **Select to add an Action, Flow Logic, or Subflow**.

   a. Select an option.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>Select the desired action. Flow Designer includes a set of Actions available to flows and subflows. Alternatively, a user with the action_designer role can create additional actions to add to flows. The IntegrationHub and spoke plugins install additional actions. To add draft actions from the More Actions menu, set <strong>Show draft actions</strong> to <strong>true</strong>. To view spokes available in the ServiceNow Store, set <strong>Show store spokes</strong> to <strong>true</strong> from the More Actions menu. <strong>Note:</strong> Under <strong>Not Installed Spokes</strong>, the system displays spokes available in the ServiceNow Store based on compatibility with the ServiceNow version and application dependency on Flow Designer.</td>
</tr>
<tr>
<td>Flow Logic</td>
<td>Select an option to specify conditional or repeated operations.</td>
</tr>
<tr>
<td>Subflow</td>
<td>Select a published subflow and define input values. In addition to adding a subflow as a flow action, you can enable the <strong>Show triggered flows</strong> option from the More Actions menu to select an activated flow and define the required inputs. Running a triggered flow ignores its trigger conditions and runs all actions.</td>
</tr>
</tbody>
</table>

To change the order of an action in a flow, drag the handle on the left side of the action to the desired location.

The system displays a set of fields depending on the option selected.

   b. To configure the action, flow logic, or subflow, fill in the fields.

   c. Click **Done**.

   d. Repeat adding actions until complete.
8. Select **Save**.  
Flow Designer saves a draft of the flow, trigger, and actions.

**What to do next**
Test your flow until you're ready to activate it. For more information on testing and editing flows, see [Editing logic and automation in App Engine Studio](#).

ℹ️ **Note:** Only activated flows can be triggered in your application.

**Adding security in App Engine Studio**
Control who can use or edit your application by adding a role.

You add security to your application by defining roles. A role is used to limit and control access to your application. The access that you define for a role is granted to all users who are assigned to the role.

An administrator assigns a role to the appropriate user based on the kind of work that the role permits. For example, to enable a manager to read and approve employee requests, an administrator would assign a role that has access to read and update records from the relevant table. The role may also need access to the portal or workspace where the manager would see the request.

Application templates automatically add security to your application. If you use a template to create your application, you can edit the roles that were added or add different roles.

App Engine Studio includes default roles that you can use for your application. For more information on the default roles, see [Base system roles](#).

Alternatively, you can build custom roles for your application.

After you submit your application, an administrator on-boards the requesting team by assigning the roles that you've defined.

**Build a new role**
Create a custom role for your application.

**Procedure**
1. From the My Apps page, open your application.
2. In your application, next to Security, select the add icon (➕).
3. Select **Build a new role** and then select **Continue**.
4. Enter a name and description for your role, and then select Continue.

Tip: For consistency with existing role names, enter a role name that is all lower-case and uses underscores instead of spaces. For example, for an office art requester role, you would enter office_art_requester.

5. For each table in your application, select one or more access controls for your role.

- **Create**
  Users with the role can create new records on the data table.

- **Read**
  Users with the role can read the data that is stored in the table.

- **Write**
  Users with the role can update existing records in the table.

- **Delete**
  Users with the role can delete table records.
6. Select the **Experience** tab, and then select the experiences that users with the role can access.

7. Select **Continue**, and then select **Done** on the summary screen.

**Use an existing role**

Use a previously created role for your application.

**About this task**

App Engine Studio includes default roles that you can use for your application. For more information on the default roles, see [Base system roles](#).

**Procedure**

1. From the My Apps page, open your application.

2. In your application, next to Security, select the add icon (+).

3. Select **Use an existing role**, and then select **Continue**.

   ![Option to use an existing role](#)

**ADD SECURITY**

**How do you want to add a role to your app?**

- ![Build a new role](#)
- ![Use an existing role](#)

4. Select the search field and then select a role from the list.
5. Select **Edit**.
   A new tab opens to set access controls for the existing role.

6. In the Experience section, select the experiences that users with the role can access.

7. In the Data section, set access controls for each available table.
   
   **Create**
   Users with the role can create new records on the data table.

   **Read**
   Users with the role can read the data that is stored in the table.

   **Write**
   Users with the role can update existing records in the table.

   **Delete**
   Users with the role can delete table records.

8. Select **Save**.

**Editing data in App Engine Studio**

Use Table Builder to edit data in your application.

Table Builder is a tool for editing data tables that you’ve added to your application. You edit a table by editing the columns of the table.

![Table Builder](image-url)
In Table Builder, each table column is represented as a row. For example, "Art title," "Image," and "Available" appear as column headers on the actual table. However, when you edit the columns in Table Builder, they appear as rows.

**Editing columns in Table Builder**

<table>
<thead>
<tr>
<th>Column label</th>
<th>Column name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Art title</td>
<td>art_title</td>
<td>String</td>
</tr>
<tr>
<td>Image</td>
<td>image</td>
<td>Image</td>
</tr>
<tr>
<td>Available</td>
<td>available</td>
<td>True/False</td>
</tr>
</tbody>
</table>

**Actual table**

<table>
<thead>
<tr>
<th>Art title</th>
<th>Image</th>
<th>Available</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search</td>
<td>Search</td>
<td>Search</td>
</tr>
</tbody>
</table>

As you edit table columns in Table Builder, think of each row as a field on a form. For example, if you change a column label, use a label that describes the data that you expect the user to enter in the form field. Change the column type to support this kind of form entry.
To open Table Builder, open an application in App Engine Studio. From the application home, select the menu icon (⋯) next to a table, and then select Edit.

In Table Builder, you can perform any of the following tasks:

- Edit table properties
- Add a table column
- Delete a table column
- Change a column label
- Change a column name
- Change a column type
- Add a default value to a table column
- Select a column as the table display value
- Preview a table
- Delete a table

**Edit table properties**
Change table properties such as the table label and access controls.

**About this task**
By editing table properties, you can relabel your table, make your table extensible, or add record numbers. You can also set application access settings.

Making a table extensible means allowing new tables to share columns from your table. For example, if you created an “Office location” column in your table, you can allow new tables to use the “Office location” column also. For more information on table extension, see Table extension and classes.

Adding record numbers means automatically creating a tracking number for each new table entry. For example, a new entry to the Travel Requests table would get a record number like TRV1234567. You can use this number to find table records more easily.

Application access settings determine whether script objects from other applications can access the table in your application. You can give these script objects access to read, create, update, or delete records on your table. Alternatively, you can disable access to your table from other applications. For more information on the application access controls, see Table design and runtime settings.
Procedure

1. Select **Properties**.

2. On the dialog box that appears, update the table properties.

<table>
<thead>
<tr>
<th>Option</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change the table label</td>
<td>In the General information tab, update the <strong>Table label</strong> field.</td>
</tr>
<tr>
<td>Make the table extensible</td>
<td>a. In the General information tab, select <strong>Advanced</strong>.</td>
</tr>
<tr>
<td></td>
<td>b. Select the <strong>Make extensible</strong> check box.</td>
</tr>
<tr>
<td>Add record numbers</td>
<td>a. In the General information tab, select <strong>Advanced</strong>.</td>
</tr>
<tr>
<td></td>
<td>b. Select the <strong>Add record number</strong> check box.</td>
</tr>
<tr>
<td></td>
<td>c. Define the record numbers by updating the Prefix, Starting number, and Number of digits fields.</td>
</tr>
<tr>
<td>Make the table accessible from other applications</td>
<td>a. Open the Access tab.</td>
</tr>
<tr>
<td></td>
<td>b. In the <strong>Accessible From</strong> field, select All Application Scopes.</td>
</tr>
<tr>
<td></td>
<td>c. In the <strong>Application Access Controls</strong> field, set the access controls for the table.</td>
</tr>
<tr>
<td>Disable other tables from accessing the table</td>
<td>a. Open the Access tab.</td>
</tr>
<tr>
<td></td>
<td>b. In the <strong>Accessible From</strong> field, select This Application Scope Only.</td>
</tr>
</tbody>
</table>

3. Select **Save** to close the dialog box.

**Add a table column**

Store more information in a table record by adding a table column.
Procedure
1. Select + Add new field.
2. On the blank row, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Column label</td>
<td>Unique label for the column.</td>
</tr>
<tr>
<td>Column name</td>
<td>Database name for the column.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of information that the column contains. For example, to contain plain text in the column, select String. Depending on the type that you select, fill in the additional fields to further define the table column. For example, if you select String, define the character limit of the string input. Or, if you select Choice, define the choices that users can choose from.</td>
</tr>
<tr>
<td>Reference</td>
<td>Table that is associated with the column. This field applies only if the column type is Reference.</td>
</tr>
<tr>
<td>Max length</td>
<td>Maximum number of characters that users can enter in the field.</td>
</tr>
<tr>
<td>Default value</td>
<td>Value that populates the field automatically after a new record is created.</td>
</tr>
<tr>
<td>Display</td>
<td>Option to set the column as the display value for the table. A reference field shows the display value of the table to which it is referring. For example, the Opened by column of the task table refers to the user table. Because the display value of the user table is the user name, the Opened by field shows something like Beth Anglin or Joe Employee. When you select a display value, choose the table column that would act as an appropriate title for individual records. Only one column can act as the display value for a table.</td>
</tr>
</tbody>
</table>

Results
If you preview the table, you can see the new column added to your table. For more information on previewing a table, see Preview a table.
Delete a table column
Delete a column from your table.

Procedure
Next to the row that represents your table column, select the trash icon (🗑).

Note: Your table includes several default columns, including Created and Updated, that you can’t delete. Also, you can’t delete columns that are extended from another table.

Results
If you preview the table, you can see that the table column is no longer used in the table. For more information on previewing a table, see Preview a table.

Change a column label
Change the label that uniquely identifies a table column.

About this task
A column label is the text that’s visible to your users when they’re reading or updating application data. For example, if you create a form for requesters to fill in, each column label appears as a form field label.

Use a column label that describes the specific information that you expect users to enter. For example, to prompt a user to add comments, you would use the column label Comments instead of Text.

Procedure
From the Column label column, change a column label by updating the text in the corresponding row.

For example, to change the label of the "Office location" column, you would select the cell that says "Office location" and enter a different label.

The column label updates automatically after you select another cell.

Change a column name
Change the database name for the column.

About this task
A column name is the text that an administrator uses to track column data in the database. A column name is created automatically after you enter a column label. In Table Builder, if you change the column label, the column name also changes automatically.
You may choose to change the column name manually if a reviewer finds that the column name was causing issues in testing. Changing the column name allows you to resolve the issue without having to change the text that's visible to users.

**Procedure**

From the **Column name** column, change the column name by updating the text in the corresponding row.

For example, to change the name of the "Office location" column, you would select the cell that says u_office_location and enter a different name.

The column name updates automatically after you select another cell.

**Change a column type**

Change the type of information that is stored in a table column.

**Procedure**

1. From the **Type** column, select a cell and clear the text.
2. From the list, select a column type.

![List of column types](image)

You can search the list by entering new text. For example, to find all the string types, you would enter *string*. 
3. If required, further define the column type. For example, if you select **Choice**, define the choices that users can select from.

![Choice Type Example](image)

The column type updates automatically after you select another cell.

**Add a default value to a table column**

Define the default value to populate a table column automatically after a user creates a record.
Procedure

From the **Default value** column, add a default value of a column by updating the content in the corresponding row.

You define the default value by selecting a cell and then entering the value. The value that you enter depends on the column type. For example, if the column requires a choice, you would select one of the choices that you defined in the **Type** column.

The column name updates automatically after you select another cell.

**Select a column as the table display value**

Select a column value that is displayed in the reference fields of other tables.

**About this task**

A reference field shows the display value of the table to which it is referring. For example, the **Opened by** column of the task table refers to the user table. Because the display value of the user table is the user name, the **Opened by** field shows something like **Beth Anglin** or **Joe Employee**. When you select a display value, choose the table column that would act as an appropriate title for individual records.

Only one column can act as the display value for a table.

**Procedure**

1. From the **Display** column, check that all selections are cleared.
   
   To clear a selection, toggle off the switch.

2. From the **Display** column, select one row to act as the table display value.
   
   To select a row, toggle on the switch.

3. Select **Save**.

**Preview a table**

See what your table looks like after you’ve edited the table columns.
About this task
Each row in Table Builder represents a column from your table. Previewing the table allows you to see what the columns look like in their proper place. You can also test the experience of creating records on your table.

Procedure
1. Select Preview.
2. From the new tab, review the table preview.

Example table preview

<table>
<thead>
<tr>
<th>Requested by</th>
<th>Requested time</th>
<th>Location</th>
<th>Preferred meal</th>
<th>Dietary restrictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bath Angel</td>
<td>2013-05-08 18:00:00</td>
<td>Luigi's Ristora</td>
<td>Pizza</td>
<td>Vegetarian</td>
</tr>
<tr>
<td>John Smith</td>
<td>2013-05-08 17:00:00</td>
<td>Call by the Sea</td>
<td>Seafood</td>
<td>Some allergies</td>
</tr>
<tr>
<td>Jane Doe</td>
<td>2013-02-23 17:30:00</td>
<td>Men's Texas Style BBQ</td>
<td>Barbecue</td>
<td>None</td>
</tr>
<tr>
<td>Joe Employee</td>
<td>2013-05-01 09:00:00</td>
<td>La Cina</td>
<td>Burritos</td>
<td>Vegan</td>
</tr>
</tbody>
</table>

What to do next
If your table doesn’t work as expected, continue editing in Table Builder. Then, preview the table again.

Delete a table
Delete a table from your application.

Procedure
1. Select the menu icon (・・・), and then select Delete Table.
2. On the dialog box that appears, enter delete and then select Delete.
Results
The table is no longer available in the app dashboard.

Editing an experience in App Engine Studio
Use UI Builder, Catalog Builder, or Mobile Studio to tailor your experience to your business needs.

Edit a workspace
Use UI Builder to edit a workspace experience.

Before you begin
Add a workspace to your application. If you created an application using a template, a workspace may already be added to your application. For more information on adding a custom workspace, see Adding an experience in App Engine Studio.

Procedure
1. From the My Apps page, open your application.
2. Next to a workspace, select the menu icon (≡≡≡) and then select Edit.

3. In UI Builder, edit the workspace by adding or configuring components on each page.
   You can also add more pages to the workspace, if needed.

   Note: The default workspace pages are read-only and can't be edited. To change the content of each page, create a page variant and edit the variant.

Edit a portal
Use UI Builder to edit a portal experience.
Before you begin
Add a portal to your application. If you created an application using a template, a portal may already be added to your application. For more information on adding a custom portal, see Adding an experience in App Engine Studio.

Procedure
1. From the My Apps page, open your application.
2. Next to a portal, select the menu icon (・・・) and then select Edit.

3. In UI Builder, edit the portal by adding or configuring components on each page. You can also add more pages to the portal, if needed.

Note: The default portal pages are read-only and can’t be edited. To change the content of each page, create a page variant and edit the variant.

Edit a mobile experience
Use Mobile Studio to edit a mobile experience.

Before you begin
Add a mobile experience to your application. If you created an application using a template, a mobile experience may already be added to your application. For more information on adding a custom mobile experience, see Adding an experience in App Engine Studio.

Procedure
1. From the My Apps page, open your application.
2. Next to a mobile experience, select the menu icon (・・・) and then select Edit.
3. Edit the mobile experience in Mobile Studio.

### Edit a request form

Use Catalog Builder to edit a request form.

### Before you begin

Add a request form to your application. If you created an application using a template, a request form may already be added to your application. For more information on adding a custom request form, see Adding an experience in App Engine Studio.

### Procedure

1. From the My Apps page, open your application.
2. Next to a catalog item, select the menu icon (≡≡≡) and then select **Edit**.

### Edit a catalog item

3. Edit the request form.
## Catalog Builder steps

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
</table>
| Details | - Basic info: Item name and short description.  
- Item details: Description of the item. |
| Destination | Destination table in which a record producer creates records. |
| Location | Catalogs and categories that the item can belong to.  
**Note:** You can add up to a maximum of 50 catalogs or categories. |
| Questions | Question sets (variable sets) or questions for the item. By default, the question sets included in the template are added and cannot be removed.  
Available question types are restricted to those specified in the template.  
From the **Insert new question** list, you can include additional question sets, questions, deactivated questions, single-column container, two-column container, and line break. For information on creating a question and supported question types, see [Create a question for a catalog item in Catalog Builder](#).  
You can also add dynamic form behavior for a question. For information on adding dynamic form behavior, see [Edit a question in Catalog Builder](#).  
**Note:**  
- The questions within a question set cannot be edited. The question sets can be reordered by dragging and dropping them.  
- Question sets specified in the template cannot be removed, while the ones added by the user can be removed.  
- Removal of question sets simply removes the association with the item and does not delete the question set.  
- A single-column question set can be added to a single-column or a two-column container. A two-column question set cannot be added to a container. |
<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Settings</td>
<td>Settings to configure the catalog item behavior in Service Portal based on the request method.</td>
</tr>
<tr>
<td>Access</td>
<td>User criteria to specify users for whom the item is available or not available.</td>
</tr>
<tr>
<td>Review and submit</td>
<td>Review the item and submit it.</td>
</tr>
</tbody>
</table>

To save your changes in each step, select **Save**.

**Preview an experience**

See what an application experience looks like to your users.

**Before you begin**

Add an experience to your application. For more information, see **Adding an experience in App Engine Studio**.

**Procedure**

1. From the My Apps page, open your application.
2. From the application home, next to an experience, select **Preview**.

3. On the browser tab that opens, review the experience.

**What to do next**

If the experience looks or behaves differently than expected, edit the experience.
Delete an experience

Delete a portal, workspace, mobile experience, or request form that you no longer need.

Procedure

1. From the My Apps page, open your application.
2. Next to an experience, select the menu icon (・・・) and then select Delete.

3. On the dialog box that appears, enter delete and then select Delete.

Editing logic and automation in App Engine Studio

Use Flow Designer to tailor your automation to your business needs.

Edit a flow

Edit an existing flow in Flow Designer.

Before you begin

Add logic and automation to your application. If you created an application using a template, logic and automation may already be added to your application. For more information on adding custom logic and automation, see Adding logic and automation in App Engine Studio.

Procedure

1. From the My Apps page, open your application.
2. Next to a flow, select the menu icon (・・・) and then select Edit.
3. Take the appropriate actions to edit the flow.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change the flow name, description, or roles</td>
<td>In the main header, select Properties, enter the values you want into the appropriate fields, and then select Update.</td>
</tr>
<tr>
<td>To edit the trigger</td>
<td>In your flow, select the trigger description, fill in the fields as desired, and then select Done.</td>
</tr>
<tr>
<td>Note:</td>
<td>Modifying triggers can result in the deletion of referenced action configurations.</td>
</tr>
<tr>
<td>To edit an existing action</td>
<td>In your flow, select the action description, fill in the fields as desired, and then select Done.</td>
</tr>
<tr>
<td>To add a new action</td>
<td>In your flow, select the plus icon in the ACTION section, then proceed as you would for adding an action to a new flow.</td>
</tr>
</tbody>
</table>

4. To save your changes, select Save.

Test a flow
Test a flow to make certain it works the way you expect.

Procedure
1. From the My Apps page, open your application.
2. Next to a flow, select the menu icon (⋮) and then select Edit.
3. In Flow Designer, select Test.
The system displays the Test flow dialog. The contents of the Test flow dialog depend on the type of trigger.

4. If the flow has a record trigger, create or select a record to use for the test. To create a record, select the **Create new record** button.

5. If the flow has a record **Updated** or **Created or Updated** trigger, specify which fields and values changed in the update. To specify a field value change, select the **Create new changed field** button for each field whose value you want to change. Complete the changed field details for each changed field.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field Name</td>
<td>The field updated by the test.</td>
</tr>
<tr>
<td>Previous Value</td>
<td>The field value prior to the update.</td>
</tr>
<tr>
<td>Current Value</td>
<td>The field value after the update.</td>
</tr>
<tr>
<td>Previous Display Value</td>
<td>The field display value prior to the update.</td>
</tr>
<tr>
<td>Current Display Value</td>
<td>The field display value after the update.</td>
</tr>
</tbody>
</table>

6. Select **Run Test**.

   **Note:** Select the **Run test in background** option to test a flow asynchronously in the background.

   The system tests the flow.

7. Select **Your test has finished running. View the flow execution details**.

   **Note:** This link is created irrespective of your choice for the **Run test in background** option. If you have selected the **Run test in background** option, the execution details are displayed only after the execution is completed asynchronously in the background. Also, the execution details are associated with the flow only after execution is completed.

   The system displays the flow execution details for the test.

**What to do next**

- Review the flow execution details. For more information, see Flow execution details.

**Activate a flow**

Activate a flow to make it available to other users.
About this task
To use a flow in your application, you must activate it.

Procedure
1. From the My Apps page, open your application.
2. Next to a flow, select the menu icon (•••) and then select Edit.

3. In Flow Designer, select Activate.

Delete a flow
Delete a flow that you no longer need.

Procedure
1. From the My Apps page, open your application.
2. Next to a flow, select the menu icon (•••) and then select Delete.

3. On the dialog box that appears, enter delete and then select Delete.

Editing security in App Engine Studio
Edit the application role that you added.

Editing a role entails changing the access settings for the role. You may choose to change access settings for any of the following reasons:
- Expand the access of a role
- Reduce the access of a role
- Define role access for a newly added table or experience

Application templates automatically add security to your application. If you use a template to create your application, you can edit the roles that were added.

**Change access settings for a role**
Control the application permissions for a role.

**Procedure**
1. From the My Apps page, open your application.
2. Next to a role, select the menu icon (･･･) and then select **Edit**.

![](image.png)

A new tab opens to edit the role.
3. In the Experience section, select the experiences that users with the role can access.
4. In the Data section, set the access controls for each available table.
   - **Create**
     Users with the role can create new records on the data table.
   - **Read**
     Users with the role can read the data that is stored in the table.
   - **Write**
     Users with the role can update existing records in the table.
   - **Delete**
     Users with the role can delete table records.
5. Select **Save**.

**Delete a role**
Delete a role that you no longer need.
Procedure

1. From the My Apps page, open your application.
2. Next to a role, select the menu icon (・・・) and then select Delete.

Deleting a role

3. On the dialog box that appears, enter delete and then select Delete.

Using the Collaboration feature

Developers can add additional developers to work on an application using the Collaboration feature. In App Engine Studio, you can add or remove users and groups to be collaborators on an application from the AES dashboard.

Users have varying levels of permissions for actions on the Collaboration feature. At a high level:

<table>
<thead>
<tr>
<th>User</th>
<th>Collaboration permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users with Manage Collaborators delegated development permission to the application</td>
<td>Can:</td>
</tr>
<tr>
<td></td>
<td>• See a list of Collaborators and their descriptors</td>
</tr>
<tr>
<td></td>
<td>• Search for users or groups</td>
</tr>
<tr>
<td></td>
<td>• Select/change user or group’s Collaboration descriptor (but users cannot change customized users or groups and cannot change the owner if there is only one owner)</td>
</tr>
<tr>
<td></td>
<td>• Add collaborators by sending invite</td>
</tr>
<tr>
<td></td>
<td>• Remove users or groups (but users cannot remove customized users or groups and cannot remove the owner if there is only one owner)</td>
</tr>
<tr>
<td></td>
<td>Cannot:</td>
</tr>
<tr>
<td>User</td>
<td>Collaboration permissions</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------</td>
</tr>
</tbody>
</table>
| Users with Invite Collaborators delegated development permission to the application | Can:  
- See a list of Collaborators and their descriptors  
- Search for users or groups  
- Add collaborators by sending invite  
Cannot:  
- Select/change user or group's Collaboration descriptor  
- Remove users or groups  
- Customize a user or group's permissions |
| Users who do not have any Collaborator related roles to the application | Can:  
- See a read-only list of Collaborators and their descriptors  
Cannot:  
- Search for users or groups  
- Select/change user or group's Collaboration descriptor  
- Add collaborators by sending invite  
- Remove users or groups  
- Customize a user or group's permissions |
| Administrators | Can:  
- See a list of Collaborators and their descriptors  
- Show/hide members of a group  
- Search for users or groups  
- Select/change user or group's Collaboration descriptor (includes changing customized users or groups) |
<table>
<thead>
<tr>
<th>User</th>
<th>Collaboration permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>groups and changing the owner even if there is only one owner)</td>
</tr>
<tr>
<td></td>
<td>• Add collaborators by sending invite</td>
</tr>
<tr>
<td></td>
<td>• Remove users or groups (includes removing customized users and groups and removing the owner even if there is only one owner)</td>
</tr>
<tr>
<td></td>
<td>• Customize a user or group’s permissions</td>
</tr>
</tbody>
</table>

When a user or group is added, an collaboration task is generated and an approval flow kicks off. To find all collaboration tasks, navigate to **App Engine > Collaboration > Collaboration Tasks**. The collaboration task provides information on which application a developer is being added to, and what permissions will be granted. Approvers will sometimes need to review these task records before developers can be added to the application.

If you are an admin, you can use flow designer to modify the Collaboration Request flow. The base system Collaboration Request flow handles collaboration requests as follows:

- If the user has AES or delegated developer permissions and is not new to the platform, the collaboration request approval record is auto-approved.
- If the user does not have AES or delegated developer permissions and is new to the platform, approval is required.

See **Flow Designer** for more information.

If you are an admin, you can modify the collaboration descriptors with which developers assign other developers delegated development permissions. The base system table provides Owner and Editor collaboration descriptors. By default, Owners have the manage collaborator delegated development permission set and Editors have the invite collaborator delegated development permission set. See **Application collaboration** for more information.

**Adding a user or group to collaboration**

You can add a user or group to the Collaboration feature.

**Before you begin**

Role required: admin, manage collaborator delegated development role, or invite collaborator delegated development role
Procedure

1. From the AES dashboard, select the Manage Collaborators button.

2. The Collaborate with others modal appears with the users and groups that are collaborators. To add another user or group as a collaborator, enter the user name or group name in the Invite field.

3. You can search for a specific user or group by entering the first few characters in the Invite field. A drop-down list with matching user names and groups displays and you can select the user or group you want to add. If a user or group displays in the drop-down list but cannot be selected, that user or group has already been added as a collaborator and cannot be selected again.

4. Select the collaboration descriptor for the user or group you are adding (users with invite collaborators do not have this ability and will default to the Editor option). Only App Collaboration Descriptors that are defined in the global scope and have the standard option selected display in the list.

5. Select Send.
   One of the following things happens:
• If the user has AES or delegated developer permissions and is new to the platform, the user is listed under the pending requests section and approval is required. Once the request is approved, both the requester and the user will get an email that the user has been added to the application.

![Welcome to the team!](image)

• If the user has AES or delegated developer permissions and is not new to the platform, the collaboration request is auto-approved. Both the requester and the user will get an email that the user has been added to the application.

**Changing collaborator permissions or creating custom descriptions**

You can change the permissions that are currently assigned to a collaborator or create a custom description.

**Before you begin**

Role required: admin or manage collaborator delegated development role
Procedure

1. From the AES dashboard, select the Manage Collaborators button. The Collaborate with others modal appears with the users and groups that are collaborators.

2. For the user or group you want to change, select a different collaborator descriptor from the drop-down list.

   Every application must have at least one owner and customized users and groups are not editable by non-admins.

3. Admin only: Custom collaboration descriptions can only be changed by administrators.
   a. To customize a user or group’s collaboration descriptor in the UI, select Custom Permissions from the drop-down list.
   b. Select or deselect delegated development permissions for the user or group. If the selection matches that of another standard Collaboration descriptor, the system will provide a prompt before proceeding. For details
on each delegated development permission, see Delegate development and deployment permissions to personnel delegated development permission documentation.

4. Select Save.

Removing a user or group from Collaboration

You can remove a user or group from the Collaboration feature.

Before you begin

Note: Every application must have at least one owner. If an application has only one owner, you cannot remove the owner until another owner is assigned first.

Role required: admin or manage collaborator delegated development role
Procedure

1. From the AES dashboard, select the Manage Collaborators button. The Collaborate with others modal appears with a list of the current collaborators.

2. To remove a user or group as a collaborator, select Remove from the drop-down list.

**Collaborate with others**

<table>
<thead>
<tr>
<th>Collaborators</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="mailto:AC@example.com">AC@example.com</a></td>
</tr>
<tr>
<td><a href="mailto:AT@example.com">AT@example.com</a></td>
</tr>
</tbody>
</table>

**Edit application properties**

Rename your app, rewrite the app description, or replace the app image.

**Procedure**

1. From the My Apps page, open your application.

2. From the app home, select the gear icon (⚙).

3. On the dialog box that appears, update the application properties.
   
   - To rename the application, update the Name field.
   
   - To rewrite the application description, update the Description field.
   
   - To replace an image, select Remove image and then add a new image.

4. Select Save.
Submit an application for approval

Start the process of getting your application published by submitting it for administrator review.

Before you begin
Role required: delegated developer with permissions

Procedure
1. From the My Apps page, open your application.
2. From the app home, select Submit.
3. On the dialog box that appears, review the submission details and then select Submit.

What to do next
The administrator reviews the submitted application and checks for potential issues. You can check the status of your application from the application home in App Engine Studio.

"Pending Approval" status

Office art loaner app Pending Approval
Updated: 2021-01-21 18:05:53

After a reviewer begins testing the application, the status changes to In Validation.

"In Validation" status

Office art loaner app In Validation
Updated: 2021-01-22 12:43:58

An administrator may provide you with test accounts for different roles in your application. Log in with each of these accounts to check that the application works as expected.

If the reviewer rejects the application, the status changes to Rejected. You may also receive an email that includes the reviewer’s comments. Use the feedback to improve your application, and then submit the application again.
"Rejected" status

Office art loaner app

Updated: 2021-01-22 12:43:58

If the application passes testing, the administrator publishes the application and the status changes to Published. The administrator on-boards the team that requested the application. For example, if you specified a certain security role for the application, the administrator assigns that role to the relevant users.

"Published" status

Office art loaner app

Updated: 2021-01-22 12:43:58

Related information

Delegated development in App Engine Studio

Publish an application from App Engine Studio when linked to Source Control

You can publish a custom application from App Engine Studio when linked to source control.

Before you begin

Role required: admin

About this task

When you publish an application from App Engine Studio that is linked to source control, there is a different outcome than when you publish via the sys_app or sys_store_app Publish related link.

Procedure

1. From the My Apps page, open your application.

2. From the app home, select Publish.
   App Engine Studio displays the publish app fields.
3. Enter the following field values.

**Provide a version number and release notes**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>View your current app version and provide a new version number for your application.</td>
</tr>
<tr>
<td>Release notes</td>
<td>Provide a brief description of the changes made to your application.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Publish options</td>
<td>Select if you want the application to be published to your application repository or the ServiceNow Store.</td>
</tr>
</tbody>
</table>

Note: All application developers on the instance share the credential used to link a Git repository to an application.

4. Click **Continue**.

5. The current state of the application is committed to source control, including any untracked or uncommitted changes. The value of the `glide.sourcecontrol.default_commit_mode` property is ignored. This occurs because when the application is published, all the untracked and uncommitted changes are also published. Therefore, the state of the application in the Git repository matches what is published. See the **Commit changes** topic for more information about the `glide.sourcecontrol.default_commit_mode` property.

6. A Source Control tag is created for the new version and the application is published. If needed, the sys_app record is updated with the new store correlation ID.

Remove an application from App Engine Studio

If you have the appropriate permissions, you can remove an application from App Engine Studio.

**Before you begin**
Role required: admin, delegated developer with permissions

**About this task**
Delegated developers can delete applications based on their permissions. To request delete permissions, contact your administrator.

**Procedure**
1. From the My Apps page, open your application.
2. In your application, select the gear icon (⚙).
3. Select **Delete application**.
4. On the dialog box, enter `delete` and then select **Delete**.
Template library

Use an application template to create an application with preconfigured data, experience, automation, and security.

Document Approval template

Build an application to manage the approval of documents in your organization.

A document approval app enables your users to quickly create & use document approval workflows. Document owners & submitters can upload documents needing approvals and check the progress on those approvals. Designated approvers can approve, reject, or return documents with comments.

Admins can use the document approval application to create different workflows for different document types.

The document approval app template is an ideal solution for making your internal workflows for processing documents quickly and efficiently. Document owners/submitters can upload a document needing approval to a portal using simple upload functionality that automatically sends submissions through a unique workflow. Once a document has been received, submitters can follow the progress of their documents while designated users approve, reject, or return their document for edits. Both submitters and approvers use comments to add notes and feedback to the submission, which is stored in the system for future reference. To give this app even more variety, admins can create unlimited workflows that are customized for each document type and user that allows for multiple stages and approvers depending on what the document category requires.

For more information on creating an application using a template, see Use an application template.

Data

The Document Approval app template includes the following tables:

<table>
<thead>
<tr>
<th>Tables</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Table label [name]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document Approver Table [x_ &lt;company-code&gt;_ &lt;document_approver&gt;]</td>
<td>Tracks document approvers, their comments, and state of a document.</td>
</tr>
</tbody>
</table>
Tables (continued)

<table>
<thead>
<tr>
<th>Table label [name]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document Approval Category Table [x_ &lt;company-code&gt;_ &lt;document_approval_category&gt;]</td>
<td>Tracks the category of document approval requests.</td>
</tr>
<tr>
<td>Document Approval Table [x_ &lt;company-code&gt;_ &lt;document_approval&gt;]</td>
<td>Tracks the document approval requests.</td>
</tr>
</tbody>
</table>

**Experience**
The Document Approval app template includes the following experiences:

<table>
<thead>
<tr>
<th>Experience</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document approval mobile experience</td>
<td>Employees and managers can use the mobile app check the status of an approval, approve/reject documents, and view submissions using your organization's iOS or Android app.</td>
</tr>
<tr>
<td>Document approval workspace</td>
<td>Employees can upload documents, check the status of an approval, edit documents, and view their submissions in the workspace. Approvers can view assigned approvals, view previous submissions, and approve/reject submissions.</td>
</tr>
<tr>
<td>Document approver form</td>
<td>Approvers can view assigned approvals and approve/reject submissions.</td>
</tr>
<tr>
<td>Doc approval portal</td>
<td>Approvers can view assigned approvals, view previous submissions, and approve/reject submissions using the doc approval portal.</td>
</tr>
</tbody>
</table>

**Logic and automation**
The Document Approval app template includes the following flows:
Flows

<table>
<thead>
<tr>
<th>Flow name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document Approval - Master Flow</td>
<td>This flow is executed when a new document approval record is submitted. It then retrieves the flow to be executed from the document category and executes it.</td>
</tr>
<tr>
<td>Single Stage Approval</td>
<td>This flow allows for the user’s manager to be the approver, with basic email notifications being generated on approval, return, or rejection.</td>
</tr>
</tbody>
</table>

Security

The Document Approval app template includes the following user roles:

Roles

<table>
<thead>
<tr>
<th>Role title [name]</th>
<th>Description</th>
<th>Contains roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approvers [x_&lt;company-name&gt;_&lt;document_approval_app&gt;.approver]</td>
<td>Approvers can create, read, delete all documents, and read document approval category data.</td>
<td>Create, Read (all), Edit (all)</td>
</tr>
<tr>
<td>Admins [x_&lt;company-name&gt;_&lt;document_approval_app&gt;.admin]</td>
<td>Admins can create, read, write, and delete in the tables document_approval and document_approval_category.</td>
<td>Create, Read (all), Edit (all), Delete (all)</td>
</tr>
<tr>
<td>Submitters [x_&lt;company-name&gt;_&lt;document_approval_app&gt;.submitter]</td>
<td>Submitters can create documents, read their own documents, edit their own documents when status is not submitted and delete their own documents. They can also read data from document approval category.</td>
<td>Create, Read (own), Edit (own when status is not submitted), Delete (own)</td>
</tr>
</tbody>
</table>
Emergency Alert template

Build an application to manage communications and track essential resources during an emergency.

An emergency alert app allows your organization to keep employees informed and connected during natural disasters and infectious diseases, such as COVID-19.

You can use the emergency alert template to create command center dashboards and a mobile app to monitor and notify employees about the latest news. Employees can self-report their status, request work-from-home or time off, and receive instructions during an emergency. Managers can assess the impact on their respective teams, approve/reject requests, request for status, and automatically receive instruction on actions to take during any emergency.

For more information on creating an application using a template, see Use an application template.

Data

The Emergency Alert app template includes the following tables:

<table>
<thead>
<tr>
<th>Table label [name]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self Report Table [x_&lt;company&gt;_my_emergency_alert_app&gt;_status_report]</td>
<td>Track reported emergencies created by employees or managers. For example, the emergency details, status, duration, and affected individuals.</td>
</tr>
<tr>
<td>Work Status Request Task Table [x_&lt;company&gt;_my_emergency_alert_app&gt;_work_status_request_task]</td>
<td>Track the work status request of an emergency. For example, the requestor, the assignee, and the state of the work status.</td>
</tr>
<tr>
<td>Emergency Alert Table [x_&lt;company&gt;_my_emergency_alert_app&gt;_emergency_alert]</td>
<td>Track emergency alerts for your company. For example, the emergency type, who created the alert, and who has read the alerts.</td>
</tr>
<tr>
<td>Status Report Request Table [x_&lt;company&gt;_my_emergency_alert_app&gt;_status_report_requests]</td>
<td>Track the severity of an emergency reported by employees. For example, the status, the color associated with danger levels, and conditions.</td>
</tr>
</tbody>
</table>
Experience
The Emergency Alert app template includes the following experiences:

<table>
<thead>
<tr>
<th>Experience</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emergency Alert App</td>
<td>Employees and managers can report emergencies, provide status updates, and view/manage time off requests using your organization's iOS or Android app.</td>
</tr>
<tr>
<td>Emergency Alert - Portal</td>
<td>Employees can view reported emergencies, the status of their team, and submit a self-report using the web portal.</td>
</tr>
<tr>
<td>Self Report Form</td>
<td>Employees and managers can use the self report form to report emergencies.</td>
</tr>
<tr>
<td>Emergency Alert - Workspace</td>
<td>Admins can use the workspace to set up and send notifications, view dashboards, and configure list and forms.</td>
</tr>
</tbody>
</table>

Logic and automation
The Emergency Alert app template includes the following flows:

<table>
<thead>
<tr>
<th>Flow name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Employee emergency report request response submitted</td>
<td>Sends a notification when an employee emergency report has been submitted.</td>
</tr>
<tr>
<td>Please report or update status</td>
<td>Sends a notification when a status change has been committed.</td>
</tr>
<tr>
<td>Status request for employees from manager</td>
<td>Sends a notification to employees when a status request is made by their manager.</td>
</tr>
<tr>
<td>Create App Self Reports</td>
<td>Sends a notification when a report request is created or updated.</td>
</tr>
</tbody>
</table>
Flows (continued)

<table>
<thead>
<tr>
<th>Flow name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request for WFH/Time off approved/denied</td>
<td>Sends a notification to an employee when a WFH or Time off request is approved or denied.</td>
</tr>
<tr>
<td>Create Work Status Task When Self Report is created</td>
<td>Starts the work status task when a self report emergency is submitted.</td>
</tr>
<tr>
<td>Manager emergency request submitted</td>
<td>Sends a notification when a manager emergency report has been submitted.</td>
</tr>
<tr>
<td>Employee emergency report changes to submitted</td>
<td>Sends a notification when an employee self report has been submitted.</td>
</tr>
<tr>
<td>Request for WFH/Time off submitted</td>
<td>Starts the work from home or time off request approval flow.</td>
</tr>
<tr>
<td>Employee emergency report submitted</td>
<td>Sends a notification when an employee submits changes to an emergency report.</td>
</tr>
<tr>
<td>Create Status Report Requests for Affected</td>
<td>Creates or updates status report requests.</td>
</tr>
<tr>
<td>Emergency report/update state for managers</td>
<td>Sends a notification to update an emergency report if the source is empty.</td>
</tr>
<tr>
<td>Emergency instructions for employees</td>
<td>Sends a notification if a status update is requested.</td>
</tr>
</tbody>
</table>

Security
The Emergency Alert app template includes the following roles:

Roles

<table>
<thead>
<tr>
<th>Role title [name]</th>
<th>Description</th>
<th>Contains roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Employee [x_&lt;company-name&gt;_ &lt;my emergency_alert_app&gt;.emergency_alert_employee]</td>
<td>Self-report emergency alerts</td>
<td>Create, Read (own), Edit (own)</td>
</tr>
</tbody>
</table>
## Roles (continued)

<table>
<thead>
<tr>
<th>Role title [name]</th>
<th>Description</th>
<th>Contains roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admins</td>
<td>Admins manage the app and configure the dashboard</td>
<td>Create, Read (all), Edit (all), Delete</td>
</tr>
<tr>
<td>[x_&lt;company-code&gt;_ &lt;my emergency_alert_app&gt;.emergency_alert_admin]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Managers</td>
<td>Managers can approve/reject requests, access dashboards, and send notifications</td>
<td>Create, Read (all), Edit (all)</td>
</tr>
<tr>
<td>[x_&lt;company-code&gt;_ &lt;my emergency_alert_app&gt;.manager]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Event Registration template

Build an application to manage scheduling, booking, and registration for company events.

An internal event registration app allows your event organizers to publish a list of events to employees and collect attendees in a centralized database. The app helps streamline the event discovery and registration process.

Event organizers can manage events, wait-lists, publish new events, and view registered users’ details. Employees and attendees can view a list of events, view event info, and register to attend in a portal. Event organizers can set the size of the event, and requesters will automatically be added to a wait-list.

For more information on creating an application using a template, see Use an application template.

## Data

The Event Registration app template includes the following tables:

### Tables

<table>
<thead>
<tr>
<th>Table label [name]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Table</td>
<td>Track event information such as event name, date of the event, and capacity.</td>
</tr>
<tr>
<td>[x_&lt;company-code&gt;_ &lt;event&gt;]</td>
<td></td>
</tr>
<tr>
<td>Attendee Table</td>
<td>Track event attendee information such as name, status, and date of submission.</td>
</tr>
<tr>
<td>[x_&lt;company-code&gt;_ &lt;attendee&gt;]</td>
<td></td>
</tr>
</tbody>
</table>
### Tables (continued)

<table>
<thead>
<tr>
<th>Table label [name]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Location Table [x_&lt;company-code&gt;_&lt;event_location&gt;]</td>
<td>Track event location details such as capacity and location.</td>
</tr>
</tbody>
</table>

### Experience

The Event Registration app template includes the following experiences:

#### Experiences

<table>
<thead>
<tr>
<th>Experience</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile app for organizers</td>
<td>Organizers can view/edit events and view attendees using your organization's iOS or Android app.</td>
</tr>
<tr>
<td>Event Location Workspace</td>
<td>Organizers and admins can use the workspace to configure events, create event applications, and view lists.</td>
</tr>
<tr>
<td>Attendee Workspace</td>
<td>Attendees can view the event name, change their antecedence status, and take notes about the event.</td>
</tr>
<tr>
<td>Event Location Workspace</td>
<td>Organizers can add event name, location, status, and capacity.</td>
</tr>
<tr>
<td>Event Location Form</td>
<td>Organizers can add event name, location, status, and capacity.</td>
</tr>
<tr>
<td>Attendee Form</td>
<td>Organizers can view attendee names, numbers, status, and date submitted.</td>
</tr>
<tr>
<td>Event Reg Workspace</td>
<td>Admins and organizers can view and manage upcoming events, attendance numbers, and unpublished events.</td>
</tr>
<tr>
<td>Event Reg Portal</td>
<td>Attendees can use the web portal to search, register, and view events.</td>
</tr>
<tr>
<td>Event Form</td>
<td>Admins and organizers can use the event form to create events.</td>
</tr>
</tbody>
</table>
Logic and automation
The Event Registration app template includes the following flows:

<table>
<thead>
<tr>
<th>Flows</th>
<th>Flow name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Event Registration canceled events</td>
<td>Updates attendee records when an event is canceled.</td>
</tr>
<tr>
<td></td>
<td>Event Registration expired events</td>
<td>Set event state to expired and attendees to event ended after the event ends.</td>
</tr>
<tr>
<td></td>
<td>Send Wait-list Confirmation Email</td>
<td>Sends an email to attendees when they are added to the wait-list of an event.</td>
</tr>
<tr>
<td></td>
<td>Event RSVP Confirmation Email</td>
<td>Sends a email to attendees when they are confirmed to attend an event.</td>
</tr>
<tr>
<td></td>
<td>Send Event Details Changed Email</td>
<td>Sends an email to all confirmed and wait-listed attendees when details of an event have changed or the event was canceled.</td>
</tr>
<tr>
<td></td>
<td>Update Event Registration attendee records</td>
<td>Updates attendee records when an event is canceled or is past the end date.</td>
</tr>
<tr>
<td></td>
<td>Send Event Reminder Email</td>
<td>Sends a reminder email to Event Attendees.</td>
</tr>
</tbody>
</table>

Security
The Event Registration app template includes the following user roles:

<table>
<thead>
<tr>
<th>Roles</th>
<th>Role title [name]</th>
<th>Description</th>
<th>Contains roles</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Attendee [x_&lt;company-name&gt;_&lt;_my events app&gt;_attendee]</td>
<td>Attendees can search, sign up, and RSVP to events.</td>
<td>Create, Read (all), Update (own)</td>
</tr>
<tr>
<td></td>
<td>Organizer</td>
<td>Organizers create and manage events.</td>
<td>Create, Read (all), Update (all)</td>
</tr>
</tbody>
</table>
Roles (continued)

<table>
<thead>
<tr>
<th>Role title [name]</th>
<th>Description</th>
<th>Contains roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>[x_&lt;company-name&gt;_&lt;_my events app&gt;_organizer]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Admin [x_&lt;company-name&gt;_&lt;_my events app&gt;_admin]</td>
<td>Admins configure event locations and applications.</td>
<td>Create, Read (all), Update (all), Delete</td>
</tr>
<tr>
<td>Public [x_&lt;company-name&gt;_&lt;_my events app&gt;_admin]</td>
<td>No login is required to access features or functions with the public role.</td>
<td>No access by default.</td>
</tr>
</tbody>
</table>

Inventory Tracker template

Build an application to track the status of inventory and assets.

An Inventory Tracker app enables your business and teams to easily track what they have, where it is, and who has it. Anytime, anywhere, and on any device. If your company needs to track the stock of items, this app can be used as a stock management tool or inventory tracker at home or in a warehouse.

This app covers three inventory use cases: consumables, loaners, and discontinued. Employees can request items via mobile app or web portal from any warehouse. Inventory managers can set reorder levels and will be automatically notified if an item is low on stock. Discontinued items with outstanding stock can still be requested but will not prompt inventory managers for reorder. Inventory managers can create items manually or by uploading a spreadsheet.

For more information on creating an application using a template, see Use an application template.

Data

The Inventory Tracker app template includes the following tables:
### Tables

<table>
<thead>
<tr>
<th>Table label [name]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inventory Control Table</td>
<td>Track the stock of items used by your business. For example, item name, manufacturer, image, category, and location.</td>
</tr>
<tr>
<td>Inventory Request Table</td>
<td>Track inventory requests submitted by employees. For example, the name and department of the requestor, stock location, count, requested date, and return date.</td>
</tr>
<tr>
<td>Stock Location Table</td>
<td>Track the location of stocked items. For example, name, address, and inventory manager.</td>
</tr>
</tbody>
</table>

### Experience

The Inventory Tracker app template includes the following experiences:

<table>
<thead>
<tr>
<th>Experience</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inventory Tracker App</td>
<td>Employees can view inventory list, see stock quantities, request items, and check the status of their requests using your organization's iOS or Android app.</td>
</tr>
<tr>
<td>Inventory Control Portal</td>
<td>Admins and managers can use the inventory control portal to manage inventory items, view stock quantity, and stock locations.</td>
</tr>
<tr>
<td>Inventory Tracker Portal App Config</td>
<td>Admins and managers can configure the employee request form.</td>
</tr>
<tr>
<td>Inventory request</td>
<td>Employees can use the inventory request form to submit and manage inventory requests.</td>
</tr>
<tr>
<td>Inventory Control Workspace</td>
<td>Admins and managers can use the inventory control workspace to</td>
</tr>
</tbody>
</table>
### Experiences (continued)

<table>
<thead>
<tr>
<th>Experience</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inventory Tracker Workspace App Config</td>
<td>Admins and managers can configure the inventory tracker workspace to view and manage inventory.</td>
</tr>
<tr>
<td>Inventory Control (default view)</td>
<td>Admins and managers can use the inventory control form to manage inventory items, view stock quantity, and stock locations.</td>
</tr>
</tbody>
</table>

### Logic and automation

The Inventory Tracker app template includes the following flows:

#### Flows

<table>
<thead>
<tr>
<th>Flow name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inventory Tracker Request Completed - Admin/Manager</td>
<td>Sends a notification when an inventory request has been completed.</td>
</tr>
<tr>
<td>Inventory Tracker Inventory Discontinued - Admin</td>
<td>Sends a notification when an inventory item is discontinued.</td>
</tr>
<tr>
<td>Inventory Tracker Inventory track Created - Admin</td>
<td>Sends a notification when an inventory track is created.</td>
</tr>
<tr>
<td>Inventory Tracker Request Overdue - Employee/Manager/Admin</td>
<td>Sends a notification to employees, managers, and admins if a request is overdue.</td>
</tr>
<tr>
<td>Inventory Tracker Inventory track Updated - Admin</td>
<td>Sends a notification to admins in an inventory track is updated.</td>
</tr>
<tr>
<td>Inventory Tracker Item Out of stock - Admin/Manager</td>
<td>Sends a notification to admins and managers when an inventory item is out of stock.</td>
</tr>
<tr>
<td>Inventory Tracker Checked Out - Employee</td>
<td>Sends a notification to an employee when an item is checked out.</td>
</tr>
<tr>
<td>Flow name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Inventory Tracker Stock Location Created - Admin</td>
<td>Sends a notification to admins when a stock location is created.</td>
</tr>
<tr>
<td>Inventory Tracker Out of Stock - employee</td>
<td>Sends a notification to employees if a inventory item is out of stock.</td>
</tr>
<tr>
<td>Inventory tracker Return Date Changes - Admin/Manager</td>
<td>Sends a notification to admins and managers when a return date changes.</td>
</tr>
<tr>
<td>Inventory Tracker Comments Added - Admin/Manager</td>
<td>Sends a notification to admins and managers when comments are added to inventory requests.</td>
</tr>
<tr>
<td>Inventory tracker Request Denied - Employee</td>
<td>Sends a notification to employees when their inventory request is denied.</td>
</tr>
<tr>
<td>Inventory Tracker Request Submitted - Manager</td>
<td>Sends a notification to managers when an inventory request is submitted.</td>
</tr>
<tr>
<td>Inventory tracker Request Cancelled - Admin/Manager</td>
<td>Sends a notification to admins and managers when an employee cancels an inventory request.</td>
</tr>
<tr>
<td>Inventory Tracker Item Returned - Employee/Manager</td>
<td>Sends a notification to employees and managers when an inventory item is returned.</td>
</tr>
<tr>
<td>Inventory tracker Request Created - Admin</td>
<td>Sends a notification to admins when an inventory request is created.</td>
</tr>
<tr>
<td>Inventory Tracker Request Submitted - Employee</td>
<td>Sends a notification to employees when their inventory request is submitted.</td>
</tr>
<tr>
<td>Inventory Tracker Request Denied - Admin</td>
<td>Sends a notification to admins when an inventory request is denied.</td>
</tr>
<tr>
<td>Inventory Tracker Stock Location Updated - Admin</td>
<td>Sends a notification to admins when a stock location is updated.</td>
</tr>
<tr>
<td>Inventory Tracker Request Updated - Manager</td>
<td>Sends a notification to managers when an inventory request is updated.</td>
</tr>
</tbody>
</table>
Flows (continued)

<table>
<thead>
<tr>
<th>Flow name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inventory Tracker Request Approved - Employee/Admin/Manager</td>
<td>Sends a notification to employees, admins, and managers when an inventory request is approved.</td>
</tr>
</tbody>
</table>

Security

The Inventory Tracker app template includes the following user roles:

Roles

<table>
<thead>
<tr>
<th>Role title [name]</th>
<th>Description</th>
<th>Contains roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Employee</td>
<td>Employees can view inventory lists, check stock, request items, and check the status of their requests.</td>
<td>Create, Read (own), Update (own)</td>
</tr>
<tr>
<td>Inventory Manager</td>
<td>Inventory managers can create inventory items, define the type of item, monitor inventory levels, and update vendor information.</td>
<td>Read (all), Update (all)</td>
</tr>
<tr>
<td>Admin</td>
<td>Admins can set up multiple inventory managers and stock locations.</td>
<td>Create, Read (all), Update (all), Delete</td>
</tr>
<tr>
<td>Public</td>
<td>Public users require no login to access features or functions with the public role.</td>
<td>Contains no roles.</td>
</tr>
</tbody>
</table>

Performance Review template

Build an application to automate scheduling, notifications, and feedback for performance reviews.

A performance review app automates the process of scheduling annual or quarterly reviews, reviewing employee performance, and providing manager
feedback. The app displays report information by default and lets managers view, update, and request employee performance reviews.

Employees can complete their self-evaluation, monitor the status, and confirm evaluations via a web portal. The app will automatically notify employees and managers to submit their evaluations.

Managers can request an update to the self-evaluations, submit manager-evaluations, confirm evaluations, and monitor the status of their team's submissions via a portal.

The HR/admin role can select an annual, bi-annual, or quarterly review process schedule and edit both the self-evaluation and manager-evaluation templates. HR/admins can monitor the status of all evaluations by department, organization, or manager.

You can improve this template by incorporating the following features:

- Add continuous option as a review process
- Use Playbook to manage review schedules
- Include HR review/escalation process for both self-reviews and manager reviews
- Add peer reviews process
- Add goal setting or OKRs
- Add integration with HR software
- Add skill set assessment metrics
- Add PIP process/individual development plans
- Include compensation info/APR

For more information on creating an application using a template, see Use an application template.

Data
The Performance Review template includes the following tables:

<table>
<thead>
<tr>
<th>Table label [name]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance review table</td>
<td>Track the performance evaluations created by employees and managers. For example, the employee name, state of the review,</td>
</tr>
</tbody>
</table>
### Experience

The Performance Review template includes the following experiences:

<table>
<thead>
<tr>
<th>Experience</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile app</td>
<td>Employees can access their performance reviews using your organization's iOS or Android app.</td>
</tr>
<tr>
<td>Edit Settings form</td>
<td>Admins can use the Edit Settings form to configure performance reviews. For example, the frequency of a review, which employees are affected, and when the review is due.</td>
</tr>
<tr>
<td>Performance Review Test Workspace</td>
<td>HR/Admin users can use the workspace to configure review schedules, update evaluation templates, and monitor the status of all evaluations.</td>
</tr>
<tr>
<td>Manager Review form</td>
<td>Managers can review and make updates employee performance reviews using the manager review form.</td>
</tr>
<tr>
<td>Self Review form</td>
<td>Employees and managers can submit a self-assessment of their performance using the self review form.</td>
</tr>
<tr>
<td>Workspace form</td>
<td>Employees and managers can complete performance review using the workspace form.</td>
</tr>
<tr>
<td>Employee Portal</td>
<td>Employees can use the portal to submit, update, review, and confirm evaluations.</td>
</tr>
</tbody>
</table>
**Experiences (continued)**

<table>
<thead>
<tr>
<th>Experience</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default view form</td>
<td>Managers and Admins can use the default form view as a template for performance reviews.</td>
</tr>
</tbody>
</table>

**Logic and automation**

The Performance Review app template includes the following flows:

<table>
<thead>
<tr>
<th>Flows</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Flow name</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>Send PR Employee Update Self-Eval email</td>
<td>This flow triggers the PR Employee Update Self-Eval email.</td>
</tr>
<tr>
<td>Send PR Employee Complete Evaluation email</td>
<td>This flow triggers the PR Employee Complete Evaluation email.</td>
</tr>
<tr>
<td>Send PR Employee Manager-Eval submitted</td>
<td>This flow triggers the PR Employee Manager-Evaluation submitted email.</td>
</tr>
<tr>
<td>Send deadline email</td>
<td>Send notifications to employees to submit their reviews by a specific date.</td>
</tr>
<tr>
<td>Main Flow</td>
<td>Main flow which triggers daily. This flow controls when reviews are triggered, email reminders, and end of review period is reached.</td>
</tr>
<tr>
<td>Send reminder emails</td>
<td>Subflow to trigger reminder emails on the date indicated by sys_properties.</td>
</tr>
<tr>
<td>Create records</td>
<td>Subflow to trigger review records on date indicated by sys_properties.</td>
</tr>
<tr>
<td>Set review inactive</td>
<td>Subflow to inactivate reviews on the end date of the review period, specified in sys_properties.</td>
</tr>
<tr>
<td>Send Manager/ADMIN Self Evaluation Submitted email</td>
<td>This flow triggers the PR Admin Self-Evaluation Submitted w/o manager or the PR Manager Self Evaluation Submitted email.</td>
</tr>
</tbody>
</table>
Security
The Performance Review app template includes the following user roles:

<table>
<thead>
<tr>
<th>Role title [name]</th>
<th>Description</th>
<th>Contains roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Employee [x_&lt;company-name&gt;_ &lt;my performance_review_app&gt;.employee]</td>
<td>Employees can complete/update self-evaluations and confirm evaluations.</td>
<td>Create, Read (own), Update (own, Delete (own)</td>
</tr>
<tr>
<td>Managers [x_&lt;company-name&gt;_ &lt;my performance_review_app&gt;.manager]</td>
<td>Managers can request updates to a self-evaluation, submit manager evaluations, and monitor issues related to evaluation submissions.</td>
<td>Create, Read (my team), Update (my team)</td>
</tr>
<tr>
<td>HR/Admins [x_&lt;company-name&gt;_ &lt;my performance_review_app&gt;.admin]</td>
<td>HR/Admins can schedule reviews (annual, bi-annual, custom, or manual), edit self and manager evaluation templates, and monitor the status of all reviews.</td>
<td>Read (all), Update (all), Delete (all)</td>
</tr>
</tbody>
</table>

Team Contacts template
Build an application to manage contact information for the people in your organization.

A Team Contacts app uses a mobile experience to view employee contact information, tag favorites, and add privacy notes. The app displays report information by default and lets an employee search for employees, view employee profiles, and create a list of favorite employee contacts.

For more information on creating an application using a template, see Use an application template.

Data
The Team Contacts app template includes the following tables:
Tables

<table>
<thead>
<tr>
<th>Table label [name]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relationship Contact Table [x_&lt;company-code&gt;_&lt;team_contact&gt;_relationship_contact]</td>
<td>Tracks employee contact information. For example, contact names, date created, favorites, and updates.</td>
</tr>
<tr>
<td>Reminder Table [x_&lt;company-code&gt;_&lt;team_contact&gt;_relationship_contact_reminders]</td>
<td>Tracks reminders created by your team. For example, when to send a reminder and the subject of a reminder.</td>
</tr>
<tr>
<td>Relationship Notes Table [x_&lt;company-code&gt;_&lt;team_contact&gt;_relationship_notes]</td>
<td>Tracks the notes that employees create for each other. For example, the employee name, reason for contact, HTML notes, and option to mark the contact as a favorite.</td>
</tr>
</tbody>
</table>

Experience
The Team Contacts app template includes the following experiences:

<table>
<thead>
<tr>
<th>Experience</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile</td>
<td>Employees can access an employee directory, and relationship notes using your organization’s iOS or Android app.</td>
</tr>
<tr>
<td>Portal</td>
<td>Employees and managers can view employee profiles, mark favorite contacts, and add relationship notes using a website. The team contacts portal includes a sign-in page, a home page, and a page to add relationship notes.</td>
</tr>
<tr>
<td>Reminder Form</td>
<td>Employees can set reminders using the team contacts form.</td>
</tr>
</tbody>
</table>

Logic and automation
The Team Contacts app template includes the following flows:
Flows

<table>
<thead>
<tr>
<th>Flow name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notify Contact Reminder</td>
<td>Sends a notification about a reminder that an employee sets up in a relationship note. The notification contains details from the relationship note.</td>
</tr>
</tbody>
</table>

Security

The TeamContacts app template includes the following user roles:

<table>
<thead>
<tr>
<th>Role title [name]</th>
<th>Description</th>
<th>Contains roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manager [x_&lt;company-name&gt;_ &lt;app-name&gt;.manager]</td>
<td>Managers can view profile info, add relationship notes, and see their direct reports.</td>
<td>Create, Read (own), Update (own), Delete (own)</td>
</tr>
<tr>
<td>Admin [x_&lt;company-name&gt;_ &lt;app-name&gt;.admin]</td>
<td>Managers can configure the team contacts app and grant access to the app.</td>
<td>Read (all), Update (all), Delete</td>
</tr>
<tr>
<td>Employee [x_&lt;company-name&gt;_ &lt;app-name&gt;.employee]</td>
<td>Employees can view information on their team, search for employees, view employee profiles, and mark favorite contacts.</td>
<td>Create, Read (own), Update (own), Delete (own)</td>
</tr>
</tbody>
</table>

Time Off Requests template

Build an application to schedule employee vacations and time-off.

Streamline the process for employees to request time off through a single mobile or web-based form. Managers are notified via email when a request is submitted. They can use a mobile app or a web-based portal to view historical and pending requests by month or year for an employee or their entire team. This allows the manager to make informed decisions, improve planning for upcoming work, or identify where shifts must be covered. Employees are automatically notified when their requests are approved or rejected and why. If plans change and updates to an approved request are required, the employee
can edit and resubmitted their requests for approval through a mobile or a web-based portal where they can also see their historical requests.

For more information on creating an application using a template, see Use an application template.

Data
The Time Off Request app template includes the following tables:

<table>
<thead>
<tr>
<th>Tables</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Table label [name]</strong></td>
</tr>
<tr>
<td>Time Off Requests Table</td>
</tr>
</tbody>
</table>

Experience
The Time Off Request app template includes the following experiences:

<table>
<thead>
<tr>
<th>Experiences</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Experience</strong></td>
</tr>
<tr>
<td>Mobile app</td>
</tr>
<tr>
<td>Time off request form</td>
</tr>
<tr>
<td>Time off request workspace</td>
</tr>
</tbody>
</table>
Experiences (continued)

<table>
<thead>
<tr>
<th>Experience</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time off request portal</td>
<td>Employees can use the portal to view, edit, review, or cancel their time off requests.</td>
</tr>
<tr>
<td>Time off request record form</td>
<td>Managers can use the record form to view, submit, and approve/deny time off requests.</td>
</tr>
<tr>
<td>Time off request default form</td>
<td>Managers and Admins can use the default form view as a template for time off requests.</td>
</tr>
</tbody>
</table>

Logic and automation
The Time Off Request app template includes the following flows:

<table>
<thead>
<tr>
<th>Flows</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send notifications</td>
<td>Sends a notification to an employee or manager when a time off request state changes. The notification contains details from the time off request.</td>
</tr>
<tr>
<td>Time Off Request</td>
<td>Employees can draft, request, approve, reject, and cancel time off requests.</td>
</tr>
<tr>
<td>Send comments</td>
<td>Sends a notification to an employee or manager when a comment has been added to a time off request.</td>
</tr>
<tr>
<td>Daily Time Off inactivating records</td>
<td>Time off requests will update the remaining time off until the remaining days is set to 0.</td>
</tr>
</tbody>
</table>

Security
The Time Off Request app template includes the following user roles:
## Roles

<table>
<thead>
<tr>
<th>Role title [name]</th>
<th>Description</th>
<th>Contains roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Employee [x_&lt;company-name&gt;_ &lt;my_time_off_request_app&gt;.employee]</td>
<td>Employees can submit, view, edit, and cancel time off requests</td>
<td>Create, Read (own), Edit (own), Delete (own)</td>
</tr>
<tr>
<td>Manager [x_&lt;company-name&gt;_ &lt;my_time_off_request_app&gt;.manager]</td>
<td>Managers can submit, view, approve, and deny time off requests</td>
<td>Create, Read (all), Edit (all)</td>
</tr>
<tr>
<td>Admin [x_&lt;company-name&gt;_ &lt;my_time_off_request_app&gt;.admin]</td>
<td>Admins can manage the time off app and view all records</td>
<td>Create, Read (all), Edit (all), Delete (all)</td>
</tr>
</tbody>
</table>

### Getting help in App Engine Studio

Find help for building your application from product documentation, learning tutorials, or community forums.

The App Engine Studio Resources page contains helpful links to assist you as you build your application. For example, for help with editing a portal page, you can access the relevant UI Builder documentation on working with components.

#### Resources page

You can refer to any of the following resources:

**ServiceNow product documentation**

Read product documentation to learn about App Engine Studio concepts. You can also follow step-by-step instructions to get started on specific tasks.

**ServiceNow Developer Site**

Follow tutorials on the Developer Site to build applications for example use cases.

**Now Community**

Browse community forums to find answers from other developers using App Engine Studio. If you can’t find an answer, you can post a question in the Now Platform forum.
Source Control integration in App Engine Studio

Enable application developers to integrate with a Git Source Control repository. Save and manage multiple versions of an application from a non-production instance.

Linking an application to Source Control enables all application developers on a non-production instance to:

- Import applications from a Git repository.
- Pull and apply remote changes from a Git repository.
- Commit all local changes on the instance to a Git repository.
- Create tags to permanently link to a given version of an application.
- Create branches to maintain multiple versions of an application simultaneously.

Integration requirements

To link an application to source control:

- The user must have the admin role.
- The non-production instance must have network access to the Git repository.
- Each application must be within its own Git repository.
- The repository user credentials must grant read and write access.

⚠️ Note: All application developers on the instance share a single set of credentials per repository.

Options available from App Engine Studio

After linking an application to Source Control, application developers can use App Engine Studio to manage the repository. From App Engine Studio, developers can:

- Edit the application repository credentials.
- Commit all local changes on the instance.
- Apply remote changes from the repository.
- Create a branch.
- Switch branches.
- Import an application from a remote repository.

Source Control integration does not support managing applications on a production instance. Instead, you can manage applications on a production
instance using the application repository, an update set, or App Engine Studio. For more information about managing applications on a production instance, see Application sharing.

Options available from a Git repository
The ServiceNow platform offers limited support for modifying linked application files outside of an instance. From Git, developers can:

- Move application files to a different Git directory structure.
- Edit application files outside of App Engine Studio.

The system generates a properties text file called **sn_source_control.properties** at the root level of the repository. To move application files to a different Git directory structure, application developers can set the **path** parameter to specify the subfolder path containing their application files. For example, if you moved your application to the **src/app** subfolder, set the **path** to `path=src/app`.

The system generates a **checksum.txt** file in the Git repository to determine if any application files have been changed outside of App Engine Studio. When the checksum value from the file matches the current checksum value, the integration skips the validation and sanitization process. When the checksum values do not match, the integration validates and sanitizes the application files as part of the Source Control operation. The sanitization process:

- Creates upgrade log entries for each sanitization action taken.
- Removes unsupported folders and files from the repository.
- Aborts all Source Control operations when a system application file fails XML schema validation. For example, if a database dictionary record fails XML schema validation, the system aborts all operations.
- Skips the current Source Control operation when a non-system application file fails XML schema validation.

The Source Control integration sanitizes only content within the application path listed in the **sn_source_control.properties** file. Repository content outside the application path is ignored.

MID Server support
Use an existing MID Server to connect to a Source Control repository, or an application through a MID Server enables access to repositories behind a firewall.
Source control operations in App Engine Studio

The source control integration primarily supports operations from App Engine Studio, but can also support some operations directly from the GIT repository.

### Available source control operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
<th>Available from</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import from Source Control</td>
<td>Imports an application from the repository to the local instance.</td>
<td>App Engine Studio</td>
</tr>
<tr>
<td>Link to Source Control</td>
<td>Allows developers to manage application changes from a GIT repository.</td>
<td>App Engine Studio</td>
</tr>
<tr>
<td>Edit Repository Configuration</td>
<td>Updates the GIT repository user credentials.</td>
<td>App Engine Studio</td>
</tr>
<tr>
<td>Apply Remote Changes</td>
<td>Updates the local version of the application to match the repository version.</td>
<td>App Engine Studio</td>
</tr>
<tr>
<td>Commit Changes</td>
<td>Updates the repository version of the application to match the local version.</td>
<td>App Engine Studio</td>
</tr>
<tr>
<td>Stash Local Changes</td>
<td>Removes and saves local changes for later work.</td>
<td>App Engine Studio</td>
</tr>
<tr>
<td>Switch Branch</td>
<td>Updates the local version of the application to match the repository branch version.</td>
<td>App Engine Studio</td>
</tr>
<tr>
<td>Create Branch</td>
<td>Creates a branch in the repository to save a different version of the application.</td>
<td>• App Engine Studio&lt;br&gt;• GIT repository</td>
</tr>
<tr>
<td>Create Tag</td>
<td>Creates a tag in the repository to link to a particular application version.</td>
<td>• App Engine Studio&lt;br&gt;• GIT repository</td>
</tr>
<tr>
<td>Manage Stashes</td>
<td>Allows developers to apply or delete stashed changes.</td>
<td>App Engine Studio</td>
</tr>
<tr>
<td>Create repository</td>
<td>Creates a repository to store application changes.</td>
<td>GIT repository</td>
</tr>
</tbody>
</table>
Available source control operations (continued)

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
<th>Available from</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create credentials</td>
<td>Creates credentials to the repository.</td>
<td>GIT repository</td>
</tr>
<tr>
<td>Grant access to repository</td>
<td>Provides read and write access to the repository tied to a specific set of credentials.</td>
<td>GIT repository</td>
</tr>
</tbody>
</table>

Import application or application-customization from source control

Import an application or application-customization from a source control repository to continue developing it on this instance.

Before you begin

- Role required: admin
- Verify that the non-production instance has network access to the Git repository.
- Verify that the repository contains a valid application.
- Ensure that users add the email address to their respective Users Table (ServiceNow sys_user) record that they use in their commits to the Git repository.
- Learn more about application-customizations Manage customizations to applications.

About this task

The source control integration does not support importing an application on a production instance. Instead install applications on a production instance from the application repository, an update set, or the App Engine Studio.

Procedure

   App Engine Studio will open in a new tab.
2. Click Import app.
   App Engine Studio displays the Import from source control fields.
3. Enter the following field values.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network protocol</td>
<td>Https or ssh credential type that enables secure channel data exchange.</td>
</tr>
<tr>
<td>URL</td>
<td>The URL to the Git repository where the application files reside.</td>
</tr>
<tr>
<td>Branch</td>
<td>The repository branch to work on within the application.</td>
</tr>
</tbody>
</table>

**Note:** If the Git repo URL for SSH provided by your Git server does not work, check with your Git server owner or provider for the correct URL. There may be additional specifications such as scheme protocol prefixes, port numbers, and so on, required for your Git repo URL to function.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MID Server Name</td>
<td>Select an existing MID Server to link to a Git repository stored behind your corporate firewall. &lt;br&gt; <strong>Note:</strong> Use a separate MID Server to prevent conflicts with Discovery activities.</td>
</tr>
<tr>
<td>Default email</td>
<td>The committer email address is defined by the sys_user record if available. But if a committer's sys_user record email field is empty, the system generates an alternate email (<a href="mailto:username@instancename.service-now.com">username@instancename.service-now.com</a>). You can also enter a default email address and change it later. To use that default email address in all cases, select the check box.</td>
</tr>
<tr>
<td>Credential</td>
<td>Select the credential for your Git repository. (See Getting started with Credentials.) &lt;br&gt; <strong>Note:</strong> If you select the ssh network protocol, enter a valid credential of the SSH Private Key type. If you select the https protocol, enter a valid credential of the Basic Auth Credentials type.</td>
</tr>
</tbody>
</table>

**Note:** All application developers on the instance share the credential used to link a Git repository to an application.

4. **Click Import app.** <br>The system compares the checksum in the checksum.txt file to current checksum. When the checksum values match, the integration skips validation and imports the application. When the checksum values do not match, the integration first validates and sanitizes the application files before importing them.

5. **Click Select Application.** <br>App Engine Studio displays the application as a new choice in the Switch Applications window.

**What to do next**
- Review the upgrade logs for any sanitization applied to application files during the import.
- Select the imported application to edit it.
Related information

MID Server

Getting started with credentials

Link an application or application-customization to source control

Linking an application or application-customization to source control allows application developers to manage changes from a Git repository.

Before you begin

- Role required: admin
- Learn more about Manage customizations to applications.
- Create a dedicated Git repository for the application. For increased security, enable multi-factor authentication for the Git repository.
- Generate an access token that the source control integration can use instead of a password and multi-factor authentication passkey. Search for personal access token on GitHub or GitLab.
- Restrict permissions on the access token to allow read and write access to the Git repository.
- Verify that the non-production instance has network access to the Git repository.
- Ensure that users add the email address to their respective Users Table (ServiceNow sys_user) records that they use in their commits to the Git repository.
- Learn more:

About this task

The source control integration does not support linking to an application or customization on a production instance. Instead, install applications on a production instance from the application repository, an update set, or the App Engine Studio.

Procedure

1. Open the application you want to link to source control in App Engine Studio.
2. Select Source Control > Link to source control.
   App Engine Studio displays the Link to Source Control dialog box.
3. Enter the connection details for the Git repository.

### Source control connection details

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network protocol</td>
<td>Https or SSH credential type that enables secure channel data exchange.</td>
</tr>
<tr>
<td>URL</td>
<td>The URL to the Git repository where you want to save application files. For SSH protocol, use command to generate private key ssh-keygen -t rsa -m PEM -b 4096 -C “email@address”.</td>
</tr>
<tr>
<td>Note</td>
<td>If the Git repo URL for SSH provided by your Git server does not work, check with your Git server owner or provider for the correct URL. There may be additional specifications such as scheme protocol prefixes, port numbers, and so on, required for your Git repo URL to function.</td>
</tr>
<tr>
<td>Branch</td>
<td>The repository branch to work on within the application.</td>
</tr>
<tr>
<td>MID Server Name</td>
<td>The name of the existing MID Server to link through.</td>
</tr>
</tbody>
</table>
Field | Description
---|---
| **Note:** Use a separate MID Server to prevent conflicts with Discovery activities.  
Be sure that the MID server user can create files to the sys_attachment table and that the table can accept files of the “bundle” type.  
Linking or an application through a MID Server enables access to repositories behind a firewall.

| Default email | The committer email address is defined by the sys_user record if available. But if a committer's sys_user record email field is empty, the system generates an alternate email (username@instancename.service-now.com). You can also enter a default email address and change it later. To use that default email address in all cases, select the check box.

| Credential | The credential to be used with the selected protocol. See Getting started with Credentials to learn more about creating credentials.  
**Note:** If you select the SSH network protocol, enter a valid credential of the SSH private key type. If you select the https protocol, enter a valid credential of the Basic Auth credentials type.

| Commit Comment | An optional description of the repository or application.

**Note:** All application developers on the instance share a single set of repository credentials.

4. Click **Link to source control**.  
The system validates the connection and user credentials and displays a success message.  
All application developers on the instance can use the linked Git repository to manage changes.

**Related information**

MID Server
Getting started with credentials
Edit a Git repository configuration

You can edit a Git repository to change the network protocol selection, credentials or other field entries.

Before you begin
Role required: admin

Procedure

1. In your App Engine Studio app, select Source control > Edit repository configuration.

2. Choose https or ssh Network protocol and enter the URL address of your repository.
3. Change your credential or MID Server name if you wish.

*i Note:* If you have no MID server name, you can select a new one from the drop-down list. If you choose a new MID server, in the Source Control menu before making any further Source Control operations to avoid errors.

4. Default email field: The committer email address is defined by the sys_user record if available. But if a committer's sys_user record email field is empty, the system generates an alternate email (username@instancename.service-now.com). You can also enter a default email address and change it later. To use that default email address in all cases, select the check box.

5. Click Save.
Pull changes from repository

Application developers can pull changes from a linked Git repository to apply remote changes to the local instance.

Before you begin

- Role required: admin
- Link an application or application-customization to source control

Procedure

1. In your App Engine Studio app, select Source control > Pull from repository.
2. Review your uncommitted files, then select Stash local changes or Discard local changes.
3. Click Pull from repository.

The following operations occur:
• The system fetches the most recent changes from the remote repository.
• The system applies the remote changes to the instance.
• The system identifies any change conflicts requiring resolution.
If there are conflicts, the system displays the Resolve Conflicts window.

Delta loading is enabled by default in sys properties so your data isn't removed. You can disable this feature if you want data automatically deleted.

What to do next
Resolve any change conflicts.

Commit changes
Application developers can commit their changes on the instance to the linked Git repository. You can either select a few changes to commit, or commit all changes on the instance at once.

Before you begin
• Role required: admin
• Link an application or application-customization to source control

Procedure
1. In your App Engine Studio app, select Source control > Commit changes. The system displays the Select files to commit to source control window. The file changes from all the updates sets display. By default, the file changes from the current update set display.
2. Select the file changes you wish to commit.

3. To include untracked changes, select the **Include changes not tracked via the Customer Update [sys_update_xml] table** check box.
   - The default for this check box is set via the **glide.sourcecontrol.default_commit_mode** property.
     - Property can be set to **include_untracked** or **exclude_untracked**.
     - The **include_untracked** mode commits the updates to the application that do not generate sys_update_xml records, as well as any user-selected updates.
     - The **exclude_untracked** mode commits only updates selected by the user in the **Select files to commit to source control** dialog.
   - The base system setting for the property is **exclude_untracked**.
   - To hide the check box and use the value of the **glide.sourcecontrol.default_commit_mode** property, create the **sn_devstudio.vcs.allow_commit_mode_selection** property and set it to false.
   - Checking this check box may incur a performance penalty.

**Note:**
Commits always occur in **include_untracked** mode in the following cases:
   - Linking to Source Control for the first time. (To learn more, see Link an application or application-customization to source control.)
   - Publishing an application that's linked to Source Control from App Engine Studio. (To learn more, see .)
   - Selective commit mode is disabled.

4. Click **Continue**.

5. In **Commit comment**, enter a comment for the changes.

6. Click **Commit Files**.
   The following operations occur:
   - The system identifies all local changes.
   - The system commits all local changes to the remote repository.

**Note:** For list of known files that don’t have customer update records and are untracked, see Customer updates table
Stash local changes
Application developers can remove and save changes locally to apply them later.

Before you begin
• Role required: admin
• Link an application to source control
• Change one or more application files

About this task
Stashing changes removes them from the current application and saves them for a developer to later apply or delete.

Procedure
1. In your App Engine Studio app, select Source control > Stash local changes. The system displays a list of locally changed files.
2. Enter your description.
3. Click Stash Changes. The system saves the current changes and displays a success message.

<table>
<thead>
<tr>
<th>File name</th>
<th>File type</th>
<th>State</th>
<th>Updated date</th>
<th>Updated by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interactions</td>
<td>Dictionary Entry [sys_dictionary]</td>
<td>modified</td>
<td>2021-08-09 19:02:49</td>
<td>admin</td>
</tr>
<tr>
<td>Interactions</td>
<td>Table [sys_db_object]</td>
<td>modified</td>
<td>2021-08-09 19:02:49</td>
<td>admin</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.

263
What to do next
• Close dialog
• Manage stashes

To learn more see Getting started with credentials

Create tag
Create a tag in the repository to link to a particular application version.

Before you begin
• Role required: admin or sn_group_creator.app_creator
• Link an application or application-customization to source control

Procedure
1. In your App Engine Studio app, select Source control > Create tag.
   App Engine Studio opens the Create Branch window.

2. Enter the Tag Name.
3. Click Create Tag.
   App Engine Studio creates the tag.
4. Click Close.

What to do next
Commit changes to the new branch.
Switch branch
Application developers can switch to a different repository branch to work on another version of the application.

Before you begin
• Role required: admin
• GIT repository with one or more available branches.

Procedure
1. In your App Engine Studio app, select Source control > Switch branch.

   ![Source control menu with Switch branch option]

   The system displays the Switch Branch window.

2. Optional: If there any local changes on the instance, you can save or discard them.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save local changes</td>
<td>Saves local changes before switching to an alternate branch. You can later merge or discard the saved changes.</td>
</tr>
<tr>
<td>Discard local changes</td>
<td>Permanently deletes all local changes before switching to an alternate branch.</td>
</tr>
</tbody>
</table>
Note: Use caution when discarding local changes. Since all application developers share repository credentials, there is no way to discard just one set of user changes. Note you cannot later restore discarded changes.

3. Select the branch you want to switch to.

4. Click **Switch Branch**.
   App Engine Studio updates the local application to match the branch version from repository.

Create branch
Application developers can create a branch to work on a new version of an existing application.

Before you begin
- Role required: admin or sn_group_creator.app_creator
- Link an application or application-customization to source control

Procedure
1. In your App Engine Studio app, select **Source control > Create branch**.
   App Engine Studio opens the Create Branch window.
   
   ![Create a branch]
   Creating a branch will result in a new branch being created in the remote repository that is configured for this application. This application, including any uncommitted changes, will switch to the new branch.

   Branch name:

   Create branch

2. Enter the **Branch Name**.

3. **Optional:** To create a branch from a tag, click the **Create from Tag** dropdown list and select an existing tag.

4. Click **Create Branch**.
   App Engine Studio creates the branch.
5. Click Close.

**What to do next**
Commit changes to the new branch.

**Set default branch**
Set a default branch when you want to use a branch other than main for new changes or for your main development repository.

**Before you begin**
- Role required: admin
- Link an application or application-customization to source control

**Procedure**
1. Follow the steps to Add a system property.
2. Add the `glide.source_control.default_branch_name` property, and specify the default branch name of the Git source control repository to work from (pull requests, code commits, etc.). Application developers’ work is managed from and saved into the default branch if not otherwise specified. If not changed, this value defaults to `sn_instances/<instance_name>`.

**Manage stashes**
Application developers can apply or delete stashed changes from App Engine Studio.
Before you begin
- Role required: admin
- Link an application to source control.
- Stash one or more application file changes.

Procedure
1. In your App Engine Studio app, select **Source control > Manage stashes**. The system displays a list of locally stashed changes.
2. Click the action next to the stash you want to manage.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply</td>
<td>Commits the stashed changes to the application and checks for conflicts.</td>
</tr>
<tr>
<td>Delete</td>
<td>Removes the stashed changes.</td>
</tr>
</tbody>
</table>

Resolve conflicts
Application developers can choose which application file version to use when applying remote or stashed changes.

Before you begin
- Role required: admin
- Link an application to source control
- Apply a stashed change

About this task
Conflicts occur when there are multiple change versions of the same application file: one set of changes in the remote or stashed version and another set of changes in the local version. App Engine Studio requires developers to resolve conflicts before applying remote or stashed changes.

Procedure
1. In your App Engine Studio app, pull from repository or stash local changes. If the system identifies a conflict, it displays the Resolve Conflicts dialog.
2. Select how to resolve the conflict.
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select an action</td>
<td>Apply or discard all stashed changes. Go to step 3.</td>
</tr>
<tr>
<td>Manually merge changes</td>
<td>Individually select which changes to apply. Go to step 6.</td>
</tr>
</tbody>
</table>

3. If you want to apply or discard all stashed changes, select an **Action**.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Take Stashed Changes</td>
<td>Applies the application file version from the stashed changes.</td>
</tr>
<tr>
<td>Discard Stashed Changes</td>
<td>Applies the application file version from the most recent pull from the repository.</td>
</tr>
</tbody>
</table>

4. Click **Apply Stashed Changes**. The system applies the selected changes.

5. Click **Close Dialog**.

6. If you want to merge the conflicting changes, click **Manually Apply**. The system displays a list of version differences by field.

7. Select the field values you want the merged application file to have.

8. Click **Save Merge**. The system applies the selected changes.

**View commit history**

Application developers can view the commit history of applications linked to a source control repository.

**Before you begin**

- Role required: admin
- An existing link to a Git repository

**Procedure**

1. Navigate to **Source Control > View History**. The system displays the History window.

2. Select the commit sort order type.
3. Select the sort order direction.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>Sort by commit date.</td>
</tr>
<tr>
<td>Committer</td>
<td>Sort by user name.</td>
</tr>
</tbody>
</table>

Descending

Sort dates from the most recent to oldest date. Sort user names reverse-alphabetically from Z to A.

Ascending

Sort dates from the oldest to most recent date. Sort user names alphabetically from A to Z.

The system sorts commits by the selected sort order.

4. Select a commit.

The system displays the commit details for the selected commit.

5. Review the commit details.

<table>
<thead>
<tr>
<th>Commit Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>Committer</td>
</tr>
<tr>
<td>Date</td>
</tr>
<tr>
<td>SHA-1</td>
</tr>
<tr>
<td>Message</td>
</tr>
<tr>
<td>Files</td>
</tr>
</tbody>
</table>

6. Close the History window.

Move application files in a GIT repository

Move application files linked to source control to any folder of the repository. Allow application developers to store supporting content such as automated tests in the same repository as the applications they support.
Before you begin

- Link an application or application-customization to source control
- Role required: Source control credentials with write access

About this task

Linking an application to source control generates a properties text file called `sn_source_control.properties` at the root level of the repository. The properties file specifies the folder containing the application files. The integration tracks changes to these application files by generating a `checksum.txt` file. When the checksum matches, the integration skips the validation and sanitization process. When the checksum does not match, the integration validates and sanitizes the application files as part of the source control operation. The integration ignores all repository content outside the application path.

Note: You can set system properties `glide.source_control.checksum_required` to enable optional checksum validations and sanitizations and `glide.source_control.checksum_quick_install` to bypass sanitization steps on checksum matches. See Available system properties for more information.

Procedure

1. Login to source control repository linked to the application.
2. Create the destination folder where you will move the application files.
   
   Example
   
   For example, create the folders `src/app`.
3. Move the folder containing your application files to the destination folder.
   
   Example
   
   For example, move the folder `demo_my_app` to `src/app`.
4. Navigate to the root level of the repository.
5. Open the `sn_source_control.properties` text file in a text editor.
6. For the `path=` property, enter the folder path where you moved the application files.
   
   Example
   
   For example, enter `path=src/app`.
7. Save the properties file.
What to do next
Login to your instance and perform Source control operations in App Engine Studio from App Engine Studio.

Reviewing App Engine Studio applications
Review the applications the developers build in App Engine Studio so that you can deploy with confidence.

Deployment requests
After a developer submits an application in App Engine Studio, a deployment request is created. A deployment request is a record to track the review of submitted applications. From the deployment request form, a reviewer can deploy the application to your test environment, reject an application, or publish an application.

For more information on reviewing a deployment request, see Deployment Request form.

Testing an application
Before you publish a submitted application, test it in a non-production instance. To begin testing, open the deployment request and deploy the application to
your test environment. For more information, see Move an application to your test environment.

The goal of testing the application is to ensure that it won’t break the production instance. You may consider using the ServiceNow Automated Test Framework to run these tests.

If the application doesn’t pass testing, then you reject the deployment request. For more information, see Reject an application.

Publishing an application

If an application passes testing, make it available to employees in your organization by publishing it in your production instance. To publish an application, open the deployment request and deploy the application to your production environment.

The administrator on-boards the team that requested the application. For example, if the developer specified a certain security role for the application, the administrator assigns that role to the relevant users.

For more information on publishing an application, see Deploy an application.

Related information

- Delegated development in App Engine Studio
- Automated Test Framework (ATF)

Move an application to your test environment

Test an application safely by moving it from your development environment to your test environment.

Before you begin

Role required: admin

Procedure

1. Log in to your production instance.
2. Navigate to **App Engine > App Engine Studio > App Deployment Request**.
3. Open the deployment request that is associated with the application.
4. In the **Action** field, select **Deploy to Test Env**.
5. Save the form.
6. From the Related Links, select **Set State to In Validation**.
Results
The application is available in the non-production instance that you’ve defined as your test environment. To learn which instance is your test environment, navigate to App Engine Studio > Configuration > Pipeline and view the active pipeline for your App Engine Studio implementation.

The developer can see that the application is being tested by viewing the application status in App Engine Studio.

"In Validation" status

![Office art loaner app in Validation](image)

Updated: 2021-01-22 12:43:58

What to do next
Optionally, provide the developer with accounts to test their application using various roles in their application.

Reject an application
Let a developer know that their application is not ready to publish.

Before you begin
Role required: admin

Procedure
1. Log in to your production instance.
3. Open the deployment request that is associated with the application.
4. From the Related Links, select Set State to Rejected.
5. On the dialog box that appears, enter comments for the developer and then select OK.
6. On the form, select **Update**.

**Results**
The developer can see that the application is rejected by viewing the application status in App Engine Studio.

"Rejected" status

The deployment request state is automatically set to **Rejected**.
Your review comments are sent via email to the address included in the **Requestor Email Address** field.

**Deploy an application**
After successfully testing an application, make it available to the requesting team by deploying it to your production environment.

**Before you begin**
Role required: admin, delegated developer with permissions

**Procedure**
1. Log in to your production instance.
2. Navigate to **App Engine > App Engine Studio > App Deployment Request**.
3. Open the deployment request that is associated with the application.
4. In the **Action** field, select **Deploy to Prod Env**.
5. Save the form.
6. From the Related Links, select Set State to Published.

Results
The application is available in your production instance.

What to do next
On-board the team that requested the application. For example, if the developer specified a certain security role for the application, assign that role to the relevant users.

App Engine Studio reference
Reference topics provide additional information about the lists and forms that you use to configure and administer App Engine Studio.

App Engine Studio glossary
Learn about terms and concepts that are unique to App Engine Studio.

Application template
Provides predefined data, experience, logic and automation, and security to support a certain use case. For example, the Travel Request template provides application content for submitting and approving employee travel requests.

For more information on the available templates, see Template library.

Logic and automation
Business automate application processes. You define logic and automation using flows. A flow includes a sequence of actions and a trigger. You can use a flow template or create a flow from scratch.

Data
Information that is stored in your application. For example, employee phone numbers or office locations. You configure application data using tables.

Deployment request
Ticket to track the review of submitted applications. From the deployment request form, a reviewer can deploy the application to different environments, accept or reject an application, and send feedback to a developer.
Environment

Instance that you use for the developing, testing, or launching an application. To set up App Engine Studio, you must define environments in each of the instances that you're using as environments. For more information, Define environments.

Experience

Graphical interface that your users interact with. For example, you can create a portal where users find information, submit requests, or complete business tasks. For more information on the available application experiences, see Experiences that you can create in App Engine Studio.

Pipeline

Configuration for deploying an application to different environments.

Security

Roles and access controls to limit who can use your application.

Components installed with App Engine Studio

Several types of components are installed with activation of the App Engine Studio plugin, including tables and user roles.

Note: The Application Files table lists the components that are installed with this application. For instructions on how to access this table, see Find components installed with an application.

Roles installed

<table>
<thead>
<tr>
<th>Role title [name]</th>
<th>Description</th>
<th>Contains roles</th>
</tr>
</thead>
</table>
| App Engine Studio user [sn_app_eng_studio.user] | Builds applications in App Engine Studio. This role is assigned automatically to users in the App Engine Studio Users group. For more information, see Grant access | • catalog_builder_editor  
• app_template_runner  
• sn_g_app_creator.app_creator  
• sn_table_builder.user |
### Tables installed

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>App Details</strong></td>
<td>Details about the operations that a developer used to create an application in App Engine Studio. This table is updated automatically as developers build applications in App Engine Studio.</td>
</tr>
<tr>
<td>[sn_app_eng_studio_app_details]</td>
<td></td>
</tr>
<tr>
<td><strong>Applications in Projects</strong></td>
<td>Applications that a developer creates in App Engine Studio. This table is updated automatically as developers build applications in App Engine Studio.</td>
</tr>
<tr>
<td>[sn_app_eng_studio_project_application_m2m]</td>
<td></td>
</tr>
<tr>
<td><strong>Environment</strong></td>
<td>Instances that you use for the developing, testing, or launching an application. You update this table as you define environments for App Engine Studio. For more information, see Define environments.</td>
</tr>
<tr>
<td>[sn_app_eng_studio_environment]</td>
<td></td>
</tr>
<tr>
<td><strong>Deployment Request</strong></td>
<td>Requests to review and publish an application that a developer created in App Engine Studio. From the deployment request form, a reviewer can deploy the application to different environments, accept or reject an application, and send feedback to a developer. For more information, see</td>
</tr>
<tr>
<td>Table</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Pipeline [sn_app_eng_studio_pipeline]</td>
<td>Configurations for deploying applications to different environments. There can be only one active pipeline at a time. You update this table as you create a pipeline for the deployment of applications from App Engine Studio. For more information, see Create a pipeline.</td>
</tr>
<tr>
<td>Project [sn_app_eng_studio_project]</td>
<td>Details about App Engine Studio development sessions. A project is created automatically when a developer creates an application in App Engine Studio. This table is updated automatically as developers build applications in App Engine Studio.</td>
</tr>
<tr>
<td>Resources Content [sn_app_eng_studio_resources_content]</td>
<td>Help resources that a developer can access in App Engine Studio. This table includes configurations to support the default user experience for App Engine Studio.</td>
</tr>
<tr>
<td>Resources Content Topic [sn_app_eng_studio_resources_content_topic]</td>
<td>Categorizations of help resources in App Engine Studio. This table includes configurations to support the default user experience for App Engine Studio.</td>
</tr>
<tr>
<td>Taxonomy [sn_app_eng_studio_taxonomy]</td>
<td>Application files that a developer creates in App Engine Studio. This table is updated automatically as</td>
</tr>
<tr>
<td>Table</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Taxonomy Category</td>
<td>Categorizations for application files in App Engine Studio. By default, application files are categorized as data, experience, logic and automation, or security. This table includes configurations to support the default user experience for App Engine Studio.</td>
</tr>
<tr>
<td>[sn_app_eng_studio_taxonomy_category]</td>
<td></td>
</tr>
<tr>
<td>Taxonomy Details</td>
<td>Details about application files that a developer creates in App Engine Studio. This table includes configurations to support the default user experience for App Engine Studio.</td>
</tr>
<tr>
<td>[sn_app_eng_studio_taxonomy_details]</td>
<td></td>
</tr>
</tbody>
</table>

**App Engine Studio properties**

You can access system properties for App Engine Studio by navigating to **App Engine Studio > Configuration > System Properties**.

**sn_app_eng_studio.aes_admin_contact**

Defines the contact email for the App Engine Studio administrator. The system sends deployment request notifications to each email address listed in this property. For more information on setting this property, see [Set up the administrator contact email](#).

- **Type**: string
- **Default value**: none

**sn_app_eng_studio.delete_user_sync_queue_in_x_days**

Defines the number of days before a record is deleted from the User Sync Queue. The value reflects the number of days after the record has been created. For more information on user synchronization, see [Accessing App Engine Studio for the first time](#).
**sn_app_eng_studio.illustration_supported_content_types**
Defines the supported content types for Taxonomy Category illustrations. This property is read-only.
- Type: string
- Default value: image/svg+xml

**sn_app_eng_studio.mobile_studio_access**
Controls whether a developer can add a mobile experience to their custom application in App Engine Studio.
- Type: true | false
- Default value: true

**sn_app_eng_studio.user_sync_email_notification**
Enables the sending of an email notification to users who you add to the App Engine Studio Users group. For more information, see Grant access to App Engine Studio.
- Type: true | false
- Default value: true

**sn_app_eng_studio.user_sync_enabled**
Enables or disables the scheduled job that processes changes in the User Sync Queue. For more information on user synchronization, see Accessing App Engine Studio for the first time.
- Type: true | false
- Default value: true

**sn_app_eng_studio.user_sync_queue_enabled**
Controls whether App Engine Studio user information is queued up to synchronize with the development environment. This property doesn't control the actual synchronization, only the queuing of the changes. Changes can be queued while synchronization is disabled. If synchronization is re-enabled later, then all the changes are synchronized, including changes that were queued up
when synchronization was disabled. For more information on user synchronization, see Accessing App Engine Studio for the first time.

- Type: true | false
- Default value: true

Domain separation and App Engine Studio

Domain separation is unsupported for App Engine Studio. Domain separation enables you to separate data, processes, and administrative tasks into logical groupings called domains. You can control several aspects of this separation, including which users can see and access data.

Support level: No support

- The domain field may exist on data tables but there is no business logic to manage the data.
- This level is not considered domain-separated.

For more information on support levels, see Application support for domain separation.

Related information

Domain separation for service providers

Deployment Request form

The Deployment Request form displays detailed information about the review status of an application that a developer submits for approval.

To view the Deployment Request form, navigate to App Engine > App Engine Studio > Configuration > App Deployment Request and open a record.

Deployment Request form

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number to track the deployment request.</td>
</tr>
<tr>
<td>Name</td>
<td>Name of the deployment request.</td>
</tr>
<tr>
<td>Pipeline</td>
<td>Pipeline to define which environments to deploy to.</td>
</tr>
<tr>
<td>State</td>
<td>State of the deployment request. There are several states:</td>
</tr>
<tr>
<td></td>
<td><strong>Pending Approval</strong></td>
</tr>
<tr>
<td></td>
<td>The submitted application is ready for review.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Deployment Request form (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>In Validation</strong></td>
<td>The submitted application is being reviewed in the test environment. You enable this state by selecting <strong>Set State to In Validation</strong> from the Related Links.</td>
</tr>
<tr>
<td><strong>Published</strong></td>
<td>The submitted application has been deployed to the production instance. You enable this state by selecting <strong>Set State to Published</strong> from the Related Links.</td>
</tr>
<tr>
<td><strong>Rejected</strong></td>
<td>The submitted application is not ready to publish. You enable this state by selecting <strong>Set State to Rejected</strong> from the Related Links. The system then prompts you to include the reason why the application is rejected. These comments are then sent via email to the address included in the <strong>Requestor Email Address</strong> field.</td>
</tr>
</tbody>
</table>

Each state is visible to the developer in App Engine Studio. For example, after you set the deployment request state to **Published**, the application status in App Engine Studio changes to **Published**.

<table>
<thead>
<tr>
<th>Action</th>
<th>Action to take for the deployment request. There are several options:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>-- None --</strong></td>
<td>No action has been taken on the deployment request.</td>
</tr>
<tr>
<td><strong>Deploy to Test Env</strong></td>
<td>Option to deploy the submitted application to your test environment.</td>
</tr>
<tr>
<td><strong>Deploy to Prod Env</strong></td>
<td>Option to deploy the submitted application to your production environment.</td>
</tr>
</tbody>
</table>
Deployment Request form (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>App name</td>
<td>Name of the submitted application.</td>
</tr>
<tr>
<td>App Sys ID</td>
<td>Sys ID of the submitted application.</td>
</tr>
<tr>
<td>App version</td>
<td>Version of the submitted application.</td>
</tr>
<tr>
<td>Requestor Email Address</td>
<td>Email address of the user who requested deployment of the application. If a reviewer rejects an application, the comments are sent via email to the address included in this field.</td>
</tr>
</tbody>
</table>

Related information

- Reviewing App Engine Studio applications
- Move an application to your test environment
- Reject an application
- Deploy an application

Guided Application Creator

Guided Application Creator is an intuitive development interface for building applications on the Now Platform®. It provides a step-by-step process to guide you through your initial application construction.

Watch this four-minute video to learn about getting started with Guided Application Creator.

Explains how to use Guided Application Creator to create a new application for workspace, mobile, and classic with an imported spreadsheet.

Note: Although the video describes creating a workspace in Guided Application Creator, this feature is no longer supported. Instead, you can create a workspace using App Engine Studio. For more information, see Building applications in App Engine Studio.

Using Guided Application Creator

Setting up an application in Guided Application Creator involves:

- Creating an application record
- Defining user roles
- Designating data tables
- Designing the application for different user experiences
You can also use Guided Application Creator to create an application record and then configure it in Studio later. For more information, see Create an application record in Guided Application Creator.

**User experiences**
When you set up an application in Guided Application Creator, you can make your application available in the following user experiences.

<table>
<thead>
<tr>
<th>User experience</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile</td>
<td>ServiceNow Agent mobile app. Select this user experience to support users who work from a mobile device.</td>
</tr>
<tr>
<td>Classic</td>
<td>Classic Now Platform experience. Select this user experience to let your users work on your application via lists and forms.</td>
</tr>
</tbody>
</table>

**Guided Application Creator and the application creation process**
Setting up an application in Guided Application Creator does not capture the entire process of creating an application. It is just one step of the process. Before you set up an application in Guided Application Creator, plan what kind of application you need for your business requirements before you build it.

After you set up an application in Guided Application Creator, you can further develop the application using Studio, Flow Designer, and Team Development. After the application is fully built, you can test the application and then share the application with your users.

For more information on the application creation process, see Creating applications.

**Activation information**
Guided Application Creator is enabled via the Guided Application Creator (com.glide.sn-guided-app-creator) plugin, which is active by default in the Now Platform.

If you are using Microsoft Edge (non-Chromium) or a version of Internet Explorer that is IE11 or older, you can work only in the legacy application creator. To work in Guided Application Creator, use a different browser. For more detail on Internet Explorer 11 support, see KB0683275.

**Setting up an application in Guided Application Creator**
Set up an application in Guided Application Creator to store information and manage business processes.
Watch this nine-minute video to learn about creating a no-code application using Guided Application Creator. Shows how to create a no-code application on the ServiceNow platform quickly and easily through Guided Application Creator.

**Note:** Although the video describes creating a workspace in Guided Application Creator, this feature is no longer supported. Instead, you can create a workspace using App Engine Studio. For more information, see Building applications in App Engine Studio.

**Requirements**

Plan your application before you build it. Define the business requirements of your organization, and then identify what information you want the application to track. For more information on the application development process, see Creating applications.

If you are using Microsoft Edge (non-Chromium) or a version of Internet Explorer that is IE11 or older, you can work only in the legacy application creator. To work in Guided Application Creator, use a different browser. For more detail on Internet Explorer 11 support, see KB0683275.

You must have the sn_g_app_creator.app_creator or admin role to access Guided Application Creator.

**What to do**

1. **Create an application record in Guided Application Creator.**
   
   Create an application record in Guided Application Creator to identify a custom application.

2. **Define roles in Guided Application Creator.**
   
   Create or select roles in Guided Application Creator for the members of your organization who use your custom application.

3. **Select user experiences in Guided Application Creator.**
   
   Let users access your application via the ServiceNow Agent mobile app or lists and forms.

4. **Designate data tables in Guided Application Creator.**
   
   Select an existing table or create a custom table in Guided Application Creator to store data for your custom business application.

5. **Customize user experiences in Guided Application Creator.**
   
   Customize how the application appears in each user experience that you select.
Tip: Complete the entire process of setting up an application in Guided Application Creator. If you exit, you have to continue setting up your application in Studio and across various forms. You can't pick up where you left off in Guided Application Creator.

Next steps
Although your application is ready to use, you may need to customize your application to fit the needs of your organization.

In ServiceNow Studio, configure:
- System notifications
- Form layouts
- UI policy
- Reports

You can also learn how to automate your processes and create business logic for your application in Flow Designer.

Create an application record in Guided Application Creator
Create an application record in Guided Application Creator to identify a custom application.

Before you begin
By default, developers can't create applications in the global scope in Guided Application Creator. You can limit global application development to certain developers by assigning them an additional role. For more information, see Allow global application development in Guided Application Creator.

Role required: sn_g_app_creator.app_creator or admin

Procedure
1. Navigate to System Applications > Studio, and then select Create Application. If you are launching Guided Application Creator for the first time, select Let's get started on the welcome screen.
2. On the screen, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of your application.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of your application.</td>
</tr>
</tbody>
</table>
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope</td>
<td>Scope of your application. The application scope is set automatically when you name your application. If available, you can also select to create your application in the global scope. For more information on application scopes, see Application scope.</td>
</tr>
</tbody>
</table>

3. **Optional:** Give your application a logo image.
   You can drag and drop an image on the logo field, or you can select Drag and drop or browse to upload logo.

4. Select **Create** to assign roles for your application.
   To continue building your application in Guided Application Creator, follow the steps in Define roles in Guided Application Creator.
   You can optionally exit Guided Application Creator on the Let's create some roles for this app screen to save the application and add more functions to it later. To exit, select X to close the screen and then select Yes, close.

   ❗ **Tip:** Complete the entire process of setting up an application in Guided Application Creator. If you exit, you have to continue setting up your application in Studio and across various forms. You can't pick up where you left off in Guided Application Creator.

**Results**
Your application record is saved in the Custom Application [sys_app] table. You are added as a delegated developer for the application. To view your application later, navigate to System Applications > My Company Applications.

**Define roles in Guided Application Creator**
Create or select roles in Guided Application Creator for the members of your organization who use your custom application.
Before you begin
Complete the steps in Create an application record in Guided Application Creator.
Role required: sn_g_app_creator.app_creator or admin

About this task

Procedure
1. Define roles for your application.
   You can select existing roles or create roles.

<table>
<thead>
<tr>
<th>Option</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select an existing role</td>
<td>a. In the Roles field, enter the role name.</td>
</tr>
<tr>
<td></td>
<td>b. On the list, select the role name.</td>
</tr>
<tr>
<td>Create a role</td>
<td>a. Select + Create new role.</td>
</tr>
<tr>
<td></td>
<td>b. Enter a role name.</td>
</tr>
<tr>
<td></td>
<td>c. Select Create.</td>
</tr>
<tr>
<td>Option</td>
<td>Procedure</td>
</tr>
<tr>
<td>--------</td>
<td>-----------</td>
</tr>
</tbody>
</table>
| Select an existing role and then create a role | a. In the **Roles** field, enter the existing role name.  
b. On the list, select the existing role name.  
c. Select + **Create new role**.  
d. Enter a role name.  
e. Select **Create**. |

2. **Select Continue**.

You are taken to the screen in which you select user experiences for your application.

**Note:** If you created any roles, you must select **Continue** to save the roles to your instance. If you exit Guided Application Creator before selecting **Continue**, the roles that you created are not saved.

**What to do next**

Continue building your application by following the steps in Select user experiences in Guided Application Creator.

**Select user experiences in Guided Application Creator**

Let users access your application via the ServiceNow Agent mobile app or lists and forms.

**Before you begin**

Complete:

1. **Create an application record in Guided Application Creator**
2. **Define roles in Guided Application Creator**

Role required: **sn_g_app_creator.app_creator or admin**
Procedure

1. Select user experiences in which to make your application available.

<table>
<thead>
<tr>
<th>User experience</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile</td>
<td>ServiceNow Agent mobile app. Select this user experience to support users who work from a mobile device.</td>
</tr>
<tr>
<td>Classic</td>
<td>Classic Now Platform experience. Select this user experience to let your users work on your application via lists and forms.</td>
</tr>
</tbody>
</table>

There is no option to create a workspace in Guided Application Creator. Instead, you can create a workspace using App Engine Studio. For more information, see Building applications in App Engine Studio.

2. Select **Continue** to designate a data table for your application.

What to do next

Continue building your application by following the steps in Designate data tables in Guided Application Creator. If you exit Guided Application Creator, the user experiences that you selected are not saved to your application.
**Designate data tables in Guided Application Creator**

Select an existing table or create a custom table in Guided Application Creator to store data for your custom business application.

**Before you begin**

Complete:

1. Create an application record in Guided Application Creator
2. Define roles in Guided Application Creator
3. Select user experiences in Guided Application Creator

If you plan to create a custom data table, review the Data table guidelines for Guided Application Creator to ensure that your system performs as expected after you create a table.

Role required: sn_g_app_creator.app_creator or admin

**About this task**

---

**Procedure**

Designate data tables for your application.

You can select existing tables or create custom tables.
<table>
<thead>
<tr>
<th>Option</th>
<th>Procedure</th>
</tr>
</thead>
</table>
| Select an existing table without creating a custom table | a. In the **Tables** field, enter the table name.  
b. On the list, select the table name.  
c. Select **Done with tables**. |
| Create a custom table without selecting an existing table | a. Select **Create new table**.  
b. Select a table creation method.  
c. Follow the steps for the table creation method that you selected.  
  - Upload a spreadsheet in Guided Application Creator  
  - Extend a table in Guided Application Creator  
  - Create a table in Guided Application Creator |
| Select an existing table and then create a custom table | a. In the **Tables** field, enter the name of the existing table to designate for your application.  
b. On the list, select the table name.  
c. Select **Create new table**.  
d. Select a table creation method.  
e. Follow the steps for the table creation method that you selected.  
  - Upload a spreadsheet in Guided Application Creator  
  - Extend a table in Guided Application Creator  
  - Create a table in Guided Application Creator |

**What to do next**

Continue building your application by following the steps in **Customize user experiences in Guided Application Creator**. If you exit Guided Application Creator, the data tables that you designated are not saved to your application.
Upload a spreadsheet in Guided Application Creator

Turn your spreadsheet into a custom table in Guided Application Creator to store data for your custom application.

Before you begin

Complete:

1. Create an application record in Guided Application Creator
2. Define roles in Guided Application Creator
3. Select user experiences in Guided Application Creator
4. Designate data tables in Guided Application Creator

Review the spreadsheet guidelines to ensure that your data uploads as expected.

Role required: sn_g_app_creator.app_creator or admin

Procedure

1. To select a table creation method, on the screen, select Upload spreadsheet and then select Continue.

   ![Upload options]

   Ok. Let's create a table for this app
   If you want to create more than one, we'll do them one at a time.

   - Upload spreadsheet
   - Extend a table
   - Create table from scratch

2. Upload your spreadsheet.

   You can drag and drop the spreadsheet file or browse your system to select the spreadsheet file.

3. In Enter a row for the table header, enter the number of the header row on your spreadsheet.

   For example, if your spreadsheet has two columns in which the headers are in the second row (such as in the following image), you would enter 2.
4. Select the **Import spreadsheet data** option.

5. Select **Continue**.

   Your spreadsheet content is parsed into fields for your custom table.

6. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field Label</td>
<td>Unique label for the field.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Name of the field in the database.</td>
</tr>
<tr>
<td>Field Type</td>
<td>Type of field. For more information on the different field types, see <a href="#">Field types</a>. By default, there are only 18 field types to choose from. You can add a property to include more field types in Guided Application Creator. For more information, see <a href="#">Add field types in Guided Application Creator</a>.</td>
</tr>
<tr>
<td>Display</td>
<td>When used as a reference field, the field is used as the display value for the table. Only one field can be defined as the display value for a table.</td>
</tr>
<tr>
<td>Mandatory</td>
<td>Requires that the field must contain a value before the record can be saved.</td>
</tr>
</tbody>
</table>

7. Select **Continue** to define properties and permissions for your custom table.

8. On the form, fill in the fields.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table label</td>
<td>Unique label for the table (such as <strong>Laptops</strong> or <strong>Thin clients</strong>). The label appears on list and form views for the table. See Field Labels in <strong>Data dictionary tables</strong>.</td>
</tr>
<tr>
<td>Table name</td>
<td>Name of the table in the database. The table name is automatically populated based on the table label and the application scope that you defined earlier.</td>
</tr>
<tr>
<td>Make extensible</td>
<td>Option to enable other tables to extend this table. For more information on table extension, see <strong>Table extension and classes</strong>.</td>
</tr>
<tr>
<td>Auto-number</td>
<td>Option to add a number field to the table and automatically increment the ID numbers as they get added.</td>
</tr>
<tr>
<td>Manage access</td>
<td>User permissions for your application. For each role that you selected earlier, you can select to give your different levels of access.</td>
</tr>
</tbody>
</table>

**Create**

Enables users to insert new records (rows) into a table.

**Read**

Enables users to display records from a table.

**Write**

Enables users to update records in a table.

**Delete**

Enables users to remove records from a table or drop a table.

9. Select **Continue**.

10. On the confirmation screen, select **Continue**.

11. To add more tables to your application, follow the steps in **Designate data tables in Guided Application Creator**.

12. To finish designating tables, select **Done with tables**.
What to do next
Continue building your application by following the steps in Customize user experiences in Guided Application Creator. If you exit Guided Application Creator, the tables that you configured are not saved to the system.

Extend a table in Guided Application Creator
Extend a table in Guided Application Creator to create a custom table that copies an existing table. You can add more fields to your child table.

Before you begin
Complete:
1. Create an application record in Guided Application Creator
2. Define roles in Guided Application Creator
3. Select user experiences in Guided Application Creator
4. Designate data tables in Guided Application Creator

Plan which table to extend. Review the extension model so that you can track which database tables are created after you extend your table.

Role required: sn_g_app_creator.app_creator or admin

Procedure
1. To select a table creation method, on the screen, select **Extend a table**.

   ![Ok. Let's create a table for this app](image)
   If you want to create more than one, we'll do them one at a time.

   - **Upload spreadsheet**
   - **Extend a table**
   - **Create table from scratch**

2. In the **Table** field, select a table to extend.
3. Select **Continue**.
   The fields from the parent table are copied over to your custom table. You can't change the copied fields.
4. To add a field to your child table, select **Add a new field** and then fill in the fields.
## Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field Label</td>
<td>Unique label for the field.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Name of the field in the database.</td>
</tr>
<tr>
<td>Field Type</td>
<td>Type of field. For more information on the different field types, see <a href="#">Field types</a>. By default, there are only 18 field types to choose from. You can add a property to include more field types in Guided Application Creator. For more information, see <a href="#">Add field types in Guided Application Creator</a>.</td>
</tr>
<tr>
<td>Display</td>
<td>When used as a reference field, the field is used as the display value for the table. Only one field can be defined as the display value for a table.</td>
</tr>
<tr>
<td>Mandatory</td>
<td>Requires that the field must contain a value before the record can be saved.</td>
</tr>
</tbody>
</table>

### 5. Select **Continue** to define properties and permissions for your custom table.

### 6. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table label</td>
<td>Unique label for the table (such as Laptops or Thin clients). The label appears on list and form views for the table. See Field Labels in <a href="#">Data dictionary tables</a>.</td>
</tr>
<tr>
<td>Table name</td>
<td>Name of the table in the database. The table name is automatically populated based on the table label and the application scope that you defined earlier.</td>
</tr>
<tr>
<td>Make extensible</td>
<td>Option to enable other tables to extend this table. For more information on table extension, see <a href="#">Table extension and classes</a>.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Field | Description
--- | ---
Auto-number | Option to add a number field to the table and automatically increment the ID numbers as they get added.
Manage access | User permissions for your application. For each role that you selected earlier, you can give different levels of access.

**Create**
- Enables users to insert new records (rows) into a table.

**Read**
- Enables users to display records from a table.

**Write**
- Enables users to update records in a table.

**Delete**
- Enables users to remove records from a table or drop a table.

7. Select **Continue**.

8. On the confirmation screen, select **Continue**.

9. To add more tables to your application, follow the steps in Designate data tables in Guided Application Creator.

10. To finish designating tables, select **Done with tables**.

### What to do next
Continue building your application by following the steps in Customize user experiences in Guided Application Creator. If you exit Guided Application Creator, the tables that you configured are not saved to the system.

### Create a table in Guided Application Creator
Create a table in Guided Application Creator to customize your application to fit your business needs.

### Before you begin
Complete:

1. Create an application record in Guided Application Creator
2. Define roles in Guided Application Creator
3. Select user experiences in Guided Application Creator
4. Designate data tables in Guided Application Creator

Role required: sn_g_app_creator.app_creator or admin

Procedure

1. To select a table creation method, on the screen, select **Create table from scratch** and then select **Continue**.

![Ok. Let's create a table for this app](image)

If you want to create more than one, we'll do them one at a time.

- Upload spreadsheet
- Extend a table
- Create table from scratch

2. Add fields to your custom table.

   a. Select **Add a new field**.

   b. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field Label</td>
<td>Unique label for the field.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Name of the field in the database.</td>
</tr>
<tr>
<td>Field Type</td>
<td>Type of field. For more information on the different field types, see Field types.</td>
</tr>
<tr>
<td></td>
<td>By default, there are only 18 field types to choose from. You can add a property to include more field types in Guided Application Creator. For more information, see Add field types in Guided Application Creator.</td>
</tr>
</tbody>
</table>
3. Select **Continue** to define properties and permissions for your custom table.

4. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display</td>
<td>When used as a reference field, the field is used as the display value for the table. Only one field can be defined as the display value for a table.</td>
</tr>
<tr>
<td>Mandatory</td>
<td>Option to require that the field must contain a value before the record can be saved.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table label</td>
<td>Unique label for the table (such as <strong>Laptops</strong> or <strong>Thin clients</strong>). The label appears on list and form views for the table. See Field Labels in Data dictionary tables.</td>
</tr>
<tr>
<td>Table name</td>
<td>Name of the table in the database. The table name is automatically populated based on the table label and the application scope that you defined earlier.</td>
</tr>
<tr>
<td>Make extensible</td>
<td>Option to enable other tables to extend this table. For more information on table extension, see Table extension and classes.</td>
</tr>
<tr>
<td>Auto-number</td>
<td>Option to add a number field to the table and automatically increment the ID numbers as they get added.</td>
</tr>
<tr>
<td>Manage access</td>
<td>User permissions for your application. For each role that you selected earlier, you can give different levels of access.</td>
</tr>
</tbody>
</table>

- **Create**
  - Enables users to insert new records (rows) into a table.

- **Read**
  - Enables users to display records from a table.

- **Write**
  - Enables users to update records in a table.

- **Delete**
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Enables users to remove records from a table or drop a table.</td>
</tr>
</tbody>
</table>

5. Select **Continue**.

6. On the confirmation screen, select **Continue**.

7. To add more tables to your application, follow the steps in **Designate data tables in Guided Application Creator**.

8. To finish designating tables, select **Done with tables**.

**What to do next**

Continue building your application by following the steps in **Customize user experiences in Guided Application Creator**. If you exit Guided Application Creator, the tables that you configured are not saved to the system.

**Customize user experiences in Guided Application Creator**

Customize how the application appears in each user experience that you select.

**Before you begin**

1. Create an application record in Guided Application Creator
2. Define roles in Guided Application Creator
3. Select user experiences in Guided Application Creator
4. Designate data tables in Guided Application Creator

Role required: sn_g_app_creator.app_creator or admin

**About this task**
Customize your application for the Mobile Agent app mobile app

If you selected the Mobile Agent app mobile app as a user experience for your application, you can customize how the application appears in the mobile app.

**Procedure**

1. Next to **Mobile**, select **Start** to configure application details.
2. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>App icon</td>
<td>Icon that appears on the application home page in the mobile app.</td>
</tr>
<tr>
<td>Available offline</td>
<td>Option to enable users to update application records while in offline mode. This option is available only if the SG Offline Support (com.glide.sg.offline) plugin is activated. For information on offline mode, see <a href="#">Updating records without an internet connection</a>.</td>
</tr>
<tr>
<td>Name</td>
<td>Name of your application as it appears on the application home page. By default, this value is the application name that you created earlier.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of your application. By default, this value is the application description that you created earlier.</td>
</tr>
<tr>
<td>Tables</td>
<td>Tables that appear as applets in the mobile app. Applets are subsections under your application that categorize the information and business processes in your application.</td>
</tr>
</tbody>
</table>
3. Select **Create**.

**What to do next**
You’ve finished customizing your application for the Mobile Agent app mobile app. If you selected another user experience, customize that user experience next. If you have no more user experiences left to customize, select **Done with apps**.

**Customize your application for the classic Now Platform experience**
If you selected the classic Now Platform experience for your application, you can customize how the application appears in the application navigator.

**Procedure**
1. Next to **Classic**, select **Start** to configure application details.
2. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of your application as it appears on the application navigator. By default, this value is the application name that you created earlier.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of your application. By default, this value is the application description that you created earlier.</td>
</tr>
<tr>
<td>Tables</td>
<td>Tables in your application. Application modules are created for each table. By default, these values are the data tables that you designated earlier.</td>
</tr>
<tr>
<td>Roles</td>
<td>User roles that can access your application. By default, these values are the roles that you defined earlier.</td>
</tr>
</tbody>
</table>

3. Select **Create**.

**What to do next**
You’ve finished customizing your application for the classic Now Platform experience. If you selected another user experience, customize that user experience next.
experience next. If you have no more user experiences left to customize, select **Done with apps.**

**Data table guidelines for Guided Application Creator**

When you create an application in Guided Application Creator, you can optionally create a data table for your application. To ensure that you are within the limits of your subscription and that your application performs as expected, consult the data table guidelines before you create a data table.

### Ok. Let's create a table for this app

If you want to create more than one, we'll do them one at a time.

- **Upload spreadsheet**
- **Extend a table**
- **Create table from scratch**

⚠️ **Note:**

Now Platform application subscriptions include custom table entitlements. You can create custom tables for any purpose, up to the entitlement limit in the subscription. To learn more about how your usage administrator allocates the custom tables that you create to subscriptions, see **Managing your custom applications and tables** in Subscription Management.

To allocate your custom tables to subscriptions and stay within the entitlement limit, follow the steps in **Map your custom tables**.

**Spreadsheet guidelines**

One way to create a new data table in Guided Application Creator is to upload an external spreadsheet. Prepare your spreadsheet beforehand so that you can transfer your data seamlessly and build custom applications faster.

Ensure your spreadsheet is:

- Formatted with horizontal columns and a header label for each
- Saved as an XLSX file type

For example, format your spreadsheet as follows.
In Guided Application Creator, you would designate the second row as the header row so that **Client name** and **Phone number** are uploaded as table fields.

**Table extension guidelines**

One option for creating a data table in Guided Application Creator is to copy an existing data table and then add more fields to the newly created table. Before you extend a data table, review the extension model for the table so that you can track which database tables are created after extension.

Extending a base table incorporates all the fields of the original table and creates system fields for the new table. If they are in the same scope or if they can be configured from other scopes, you can extend tables that are marked as extensible.

For more information on extension models, see [Table extension and classes](#).

**Table creation guidelines**

You can optionally create a new table in Guided Application Creator, but it’s possible to create more custom tables than your subscription permits. To ensure that you remain within the limits of your subscription, review the custom table entitlements for your organization.

**Allow global application development in Guided Application Creator**

By default, developers can’t create applications in the global scope in Guided Application Creator. You can limit global application development to certain developers by assigning them an additional role.
Before you begin

**Important:** Applications that you create in the global scope bypass scope protections and may cause licensing issues. For information on what to expect when you create an application in the global scope, see Application scope.

Role required: admin

About this task

You designate someone to use Guided Application Creator by assigning them the sn_g_app_creator.app_creator role. You can allow this user to create applications in the global scope by also assigning the sn_g_app_creator.global role.

Alternatively, to allow all users with the sn_g_app_creator.app_creator role to create applications in the global scope, add the `sn_g_app_creator.allow_global` system property and set it to `true`. For more information on adding a property, see Add a system property.

**Note:** Users with the admin role can create applications in the global scope, regardless of whether or not they have the sn_g_app_creator.global role.

Procedure

1. Navigate to User Administration > Users and open the user record of the developer.
2. On the Roles related list, select Edit....
3. On the Edit Members page, move the sn_g_app_creator.global role from Collection to Roles List.
Ensure that the sn_g_app_creator.app_creator role is also selected. The developer needs both the sn_g_app_creator.app_creator and sn_g_app_creator.global roles to create applications in the global scope in Guided Application Creator.

4. Select Save.

5. On the user record, select Update.

Results
The option to create an application in the global scope is available to developers with the sn_g_app_creator.global role.
Add field types in Guided Application Creator

When you add fields to a custom table in Guided Application Creator, there are only 18 field types available by default. You can add a property to include more field types.

Before you begin
Role required: admin

About this task
For a list of field types that are available in the Now Platform, see Field types.

Procedure
1. Add a system property with the following settings:
   - Name: sn_g_app_creator.field_types
   - Type: string
2. In the Value field, enter a comma-separated list of field type names. For example, to add the Image and Document ID field types, enter user_image,document_id.
3. Select Submit.
**ServiceNow Studio**

ServiceNow Studio provides an Integrated Development Environment (IDE)-like interface for application developers to work on custom applications in one centralized location. It offers a simple way to create, review, and update application files from a tabbed environment.

The system opens Studio whenever you edit a custom application.

**Capabilities**

- Create an application and application artifacts.
- Perform code search.
- Integrate with source control.
- Create your company's customizations to store applications that belong to other organizations.
- Use Virtual Agent Designer to create and manage topics, which are blueprints for conversations between a virtual agent and a user. You can design topics that help your users resolve common work issues or guide them through self-service tasks.

With Studio, application developers can:
• See exactly what files comprise their application in the **Application Explorer**.
• Add new files to their application using a single **Create Application File** interface.
• Navigate to files using familiar search-by-name or by-type behavior with the **Go To** dialog.
• Find code both within and outside an application using the **Code Search** tool.
• Operate on multiple files at once using the tabbed interface.
• Operate on multiple applications at once using multiple studio windows.
• Publish the application to company instances or the ServiceNow Store.
• View information about the current application from the **Status Bar**.

### Parts of the Studio UI

<table>
<thead>
<tr>
<th>UI element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td>Displays menus and controls.</td>
</tr>
<tr>
<td><strong>File Menu</strong></td>
<td>Contains a list of application-specific options.</td>
</tr>
<tr>
<td></td>
<td>• Create File</td>
</tr>
<tr>
<td></td>
<td>• Import From Source Control</td>
</tr>
<tr>
<td></td>
<td>• Create Application</td>
</tr>
<tr>
<td></td>
<td>• Publish</td>
</tr>
<tr>
<td></td>
<td>• Settings</td>
</tr>
<tr>
<td></td>
<td>• Switch</td>
</tr>
<tr>
<td></td>
<td>• Manage Developers</td>
</tr>
<tr>
<td></td>
<td>• Launch Script Debugger</td>
</tr>
<tr>
<td><strong>Source Control Menu</strong></td>
<td>Contains a list of source control options.</td>
</tr>
<tr>
<td></td>
<td>• Link To Source Control</td>
</tr>
<tr>
<td></td>
<td>• Edit Repository Configuration</td>
</tr>
<tr>
<td></td>
<td>• Apply Remote Changes</td>
</tr>
<tr>
<td></td>
<td>• Commit Changes</td>
</tr>
<tr>
<td></td>
<td>• Stash Local Changes</td>
</tr>
<tr>
<td></td>
<td>• Switch Branch</td>
</tr>
<tr>
<td></td>
<td>• Create Branch</td>
</tr>
</tbody>
</table>
### Parts of the Studio UI (continued)

<table>
<thead>
<tr>
<th>UI element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Window Menu</strong></td>
<td>Contains a list of tab management options.</td>
</tr>
<tr>
<td></td>
<td>• Close Current Tab</td>
</tr>
<tr>
<td></td>
<td>• Close All Tabs</td>
</tr>
<tr>
<td></td>
<td>• Close Other Tabs</td>
</tr>
<tr>
<td></td>
<td>• Close Unmodified Tabs</td>
</tr>
<tr>
<td><strong>Search Menu</strong></td>
<td>Contains a list of search options.</td>
</tr>
<tr>
<td></td>
<td>• Go To</td>
</tr>
<tr>
<td></td>
<td>• Code Search</td>
</tr>
<tr>
<td><strong>User name</strong></td>
<td>The header displays the name of current user.</td>
</tr>
<tr>
<td><strong>Create Application File</strong></td>
<td>Allows developers to add an application file to an application.</td>
</tr>
<tr>
<td><strong>Go To</strong></td>
<td>Search for application files by name or type.</td>
</tr>
<tr>
<td><strong>Code Search</strong></td>
<td>Search within application files for a text string. Search options include:</td>
</tr>
<tr>
<td></td>
<td>• Restrict search to a particular table</td>
</tr>
<tr>
<td></td>
<td>• Include all applications</td>
</tr>
<tr>
<td><strong>Application Explorer</strong></td>
<td>Displays a list of application files by type.  Resize the Application Explorer to see more about application files or to provide more space for the content frame.</td>
</tr>
<tr>
<td><strong>Collapse All</strong></td>
<td>Collapses all nodes in the application explorer.</td>
</tr>
<tr>
<td><strong>Expand All</strong></td>
<td>Expand all nodes in the application explorer.</td>
</tr>
<tr>
<td><strong>Data Model &gt; Tables</strong></td>
<td>A list of application tables.     Click a table name to display and edit it in the content frame.</td>
</tr>
<tr>
<td><strong>Access Control</strong></td>
<td>A list of application access elements such as:</td>
</tr>
</tbody>
</table>
## Parts of the Studio UI (continued)

<table>
<thead>
<tr>
<th>UI element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Roles</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Access Controls</strong></td>
<td></td>
</tr>
<tr>
<td>Click a record name to display and edit it in the content frame.</td>
<td></td>
</tr>
</tbody>
</table>

### Navigation

A list of application navigational elements such as:
- **Application Menus**
- **Modules**
- **Application Menus (Mobile)**
- **Modules (Mobile)**

Click a record name to display and edit it in the content frame.

<table>
<thead>
<tr>
<th>Content frame</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Welcome to Studio</td>
<td>A list of keyboard shortcuts.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tabs</th>
<th>Each tab contains a specific application file record identified by the record name and file type. Click a tab to display and edit the record. A tab with a blue circle icon indicates that the record contains unsaved changes.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Status Bar</th>
<th>Displays information about the application and the source control integration.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application name</td>
<td>The status bar displays the name of the current application.</td>
</tr>
<tr>
<td>Application version</td>
<td>The status bar displays the current application version.</td>
</tr>
<tr>
<td>Total files</td>
<td>The status bar displays the total number of application files.</td>
</tr>
<tr>
<td>Unsaved files</td>
<td>The status bar displays the current number of application files with unsaved changes.</td>
</tr>
<tr>
<td>Source control</td>
<td>The status bar displays an icon indicating the current status of the source control integration.</td>
</tr>
</tbody>
</table>
Parts of the Studio UI (continued)

<table>
<thead>
<tr>
<th>UI element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>integration status</td>
<td></td>
</tr>
</tbody>
</table>

**Related information**

**Contextual development environment**

**Access ServiceNow Studio**

Application developers access ServiceNow Studio to create, import, or open applications.

**Before you begin**

Role required: admin

**Procedure**

1. Select the path available to your role.

<table>
<thead>
<tr>
<th>User role</th>
<th>Action to open Studio</th>
</tr>
</thead>
<tbody>
<tr>
<td>admin</td>
<td>Navigate to <strong>System Applications &gt; Studio</strong>.</td>
</tr>
</tbody>
</table>

The Studio opens in a new browser tab and displays the Select Application modal dialog.

- To create an application, click **Create Application**.
The system displays a list of application creation options.

- To import an existing application from source control, click **Import From Source Control** list.

Studio opens **Import From Source Control** page.

**Note:** In a few scenarios for scoped applications installed via plugins, the Name field in the Store Applications (sys_store_app) table doesn't match the plugin name. To make sure you are choosing the correct application, search the Select Store Application list picker with the Name of the application mentioned in the sys_store_app. For example: The plugin **Agile - Scaled Agile Framework - Essential SAFe** creates the sys_store_app record with the name "SAFe". Therefore you would search the Select Store Application list picker with the word "SAFe".

- To view an existing application, click the application name from the **Applications** list. The list also includes any store applications or scoped applications installed via plugins that are customized.

2. To open an installed store application so you can create your company's application-customization for it, click **Select Store application** and select the store application from the list picker display.
Studio opens the selected application.

**Note:** The linked icon 🔄 indicates that there are uncommitted local changes in the application. The property is ignored for your company applications in development. The **Can Edit Application in Studio** property for a store application or plugin determines if the application can be edited in Studio. If set to false, you see a warning that you cannot edit the application in Studio. But you can open the application in Studio to use Source Control features or to publish its customizations to the application repository. The default is true for new applications. The application’s owner can change the property value before releasing a new version.

---

### ServiceNow Studio keyboard shortcuts

ServiceNow Studio supports various keyboard shortcuts to manage and edit application files.

#### Studio keyboard shortcuts

<table>
<thead>
<tr>
<th>Keyboard combination</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Windows:</strong> Control+Shift +O</td>
<td><strong>Go To.</strong> Open any file in your application.</td>
</tr>
</tbody>
</table>
### Studio keyboard shortcuts (continued)

<table>
<thead>
<tr>
<th>Keyboard combination</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mac</strong>: Command + Shift + O</td>
<td>Create New. Create an application file.</td>
</tr>
<tr>
<td><strong>Windows</strong>: Control + Shift + C</td>
<td>Code Search. Search for any file or value.</td>
</tr>
<tr>
<td><strong>Mac</strong>: Command + Shift + C</td>
<td>Close Tab. Close the current tab. If the tab contains unsaved changes, the system prompts the user to save them.</td>
</tr>
<tr>
<td><strong>Windows</strong>: Control + Shift + F</td>
<td></td>
</tr>
<tr>
<td><strong>Mac</strong>: Command + Shift + F</td>
<td></td>
</tr>
<tr>
<td><strong>Windows</strong>: Control + Shift + X</td>
<td></td>
</tr>
<tr>
<td><strong>Mac</strong>: Command + Shift + X</td>
<td></td>
</tr>
</tbody>
</table>

### Related information

**ServiceNow Studio**

### Add an application file to an application

Studio allows application developers to add new application files by type.

**Before you begin**

Role required: admin  
This procedure requires creating a scoped application.

**About this task**

You can add application files to update the features of a custom application.
Procedure

1. Navigate to **System Applications > My Company Applications**.

2. From the **Develop** tab, click the **Edit** button next to the application you want to modify.
   The system opens the application in the Studio.

3. From the content frame, click **Create Application File**.
   You can also use a Studio keyboard shortcut.

### Studio keyboard shortcut

<table>
<thead>
<tr>
<th>Keyboard combination</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Windows</strong>: Control+Shift+C</td>
<td><strong>Create New</strong>. Create an application file.</td>
</tr>
<tr>
<td><strong>Mac</strong>: Command+Shift+C</td>
<td></td>
</tr>
</tbody>
</table>

Studio opens the Create New Application File pop-up window.

![Create New Application File](image)

**Note:** Not all application file types will display in Studio. Some types of application files need to be created outside Studio, such as dashboards, even though they extend sys_metadata table.

4. Find the application file type you want to create.
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search by application file name</td>
<td>In the Filter entry field, enter the name of the application file.</td>
</tr>
<tr>
<td>Search by category</td>
<td>From the left pane, select a category name.</td>
</tr>
</tbody>
</table>

5. From the results pane, select the application file type you want to create.

6. Click Create.

   The system displays a blank form for the application file type in a new studio tab.

**What to do next**

Enter the necessary fields for the particular application file type you selected.

**Related information**

Application files

**Add an experience to UI Builder from Studio**

Create an experience from Studio for users to interact with your Now® Experience UI Builder application, such as a workspace experience.

**Before you begin**

Role required: System Administrator

**Procedure**

1. Navigate to System Applications > Studio.

2. Open an existing app, or create an app. For more information, see Legacy application creator.

3. Click Create Application File.

4. Select Now Experience > UX Application to add a UI Builder experience, and then click Create.
a. Add a title for the new experience. For example, New UI Experience for new application.

b. Choose an app shell for the experience, such as Agent Workspace App Shell. For more information on app shells, see App Shells for UI Builder.

c. Type a URL path for the experience, such as newhome.

d. Ensure the Active check box is selected, then click Submit.

5. Click Create Application File again.

6. Select Now Experience > UX App Configuration to configure the new UI Builder experience, and then click Create.
Configure UX experience
Create Application File

- Add a name for the configuration. For example, `New UI Experience for new application - app config`.
- Add a landing path. This is the same name as the URL path of your experience. For example, `newhome`.
- Ensure the Active check box is selected, then click Submit.

The new experience for the new application, and the application config files show in the Application Explorer.

7. Select the UX Application tab that has your new experience. You need to tie the experience file together with the application configuration file you created.
8. Click the search icon next to the Admin panel. Search for and select your experience. Click OK > Update.

Now, when you go back to the new experience that you created, you can open it in UI Builder from Studio. You can also delegate a developer to have UI
Builder application access. For more information, see Delegated development and deployment.

**Open experience in UI Builder**

- **New UI Experience for new application**
- **Test**
- **newhome**
- **Open in UI Builder**

**Related information**

- Delegated development and deployment
  - Delegate development and deployment permissions to personnel

**Publish an application from ServiceNow Studio when linked to Source Control**

You can publish a custom application from ServiceNow Studio when linked to Source Control.

**Before you begin**

Role required: admin

**About this task**

When you publish an application from ServiceNow Studio that is linked to Source Control, there is a different outcome than when you publish via the sys_app or sys_store_app Publish related link.
Procedure

1. When you select Publish, the sys_app or sys_app_customization record for the application is updated with the new version.

2. The current state of the application is committed to Source Control, including any untracked or uncommitted changes. The value of the glide.sourcecontrol.default_commit_mode property is ignored. This occurs because when the application is published, all the untracked and uncommitted changes are also published. Therefore, the state of the application in the Git repository matches what is published. See the Commit changes topic for more information about the glide.sourcecontrol.default_commit_mode property.

3. A Source Control tag is created for the new version and the application is published. If needed, the sys_app record is updated with the new store correlation ID.

   Note: If your application is linked to Source Control and you publish a new version outside of Studio, a source control commit and tag are not created.

Search for an application file by name or type

Application developers can use Studio to search for application files.

Before you begin

Role required: admin
This procedure requires creating a scoped application.

About this task

You can search for application files to add, remove, or update the features of a custom application.

Procedure

1. Navigate to System Applications > Applications.

2. From the Develop tab, click the Edit button next to the application you want to modify.
   The system opens the application in the Studio.

3. From the header, click Go To.
   You can also use a Studio keyboard shortcut.
**Studio keyboard shortcut**

<table>
<thead>
<tr>
<th>Keyboard combination</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows: Control+Shift+O</td>
<td><strong>Go To.</strong> Open any file in your application.</td>
</tr>
<tr>
<td>Mac: Command+Shift+O</td>
<td></td>
</tr>
</tbody>
</table>

Studio opens the Go To window.

4. Enter a search string.

![Go To window](image)

Studio displays a list of matching application files as you type.

5. From the list of search results, click a record name.
   Studio opens the application file record in a new tab in the content frame.

**Search within application files**

Studio allows application developers to search within application files for matching record values.

**Before you begin**
- Role required: admin
- This procedure requires creating a scoped application.

**About this task**
- You can search within application files to add, remove, or update application file values.

**Procedure**
1. Navigate to **System Applications > Applications**.
2. From the **Develop** tab, click the **Edit** button next to the application you want to modify.
   The system opens the application in the Studio.
3. From the header, click **Code Search**.
   You can also use a Studio keyboard shortcut.
Studio keyboard shortcut

<table>
<thead>
<tr>
<th>Keyboard combination</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows: Control+Shift+F</td>
<td>Code Search. Search for any file or value.</td>
</tr>
<tr>
<td>Mac: Command+Shift+F</td>
<td></td>
</tr>
</tbody>
</table>

Studio opens the Search pop-up window.

4. In **Search term**, enter a search string.

5. **Optional:** Select any additional search criteria.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select a table to search</td>
<td>Search for matches only within the selected file type.</td>
</tr>
<tr>
<td>Search in all applications</td>
<td>Search for matches throughout the instance, not just within the current application.</td>
</tr>
</tbody>
</table>

⚠️ **Note:** Searches across all applications can take a long time.

6. Click **Search**.

   The Studio conducts a case-insensitive search of the application files you selected. While the search is running, Studio displays a search progress indicator. You can click the cancel icon to stop the search. When the search is complete, the system opens a new tab in the content frame to display the search results by application file type. Each application file type displays the number of matching search results.
7. From the search results tab, expand an application file type and click a record name. Studio opens the application file record in a new tab in the content frame.

**Update a custom application record**

You can update a custom application record to add new features or change application functionality.

**Before you begin**

Role required: admin or a delegated developer role granting full access

**About this task**

You can only update applications in development on your local instance. You cannot edit applications downloaded from your company application repository or the ServiceNow Store.

**Procedure**

1. Navigate to **System Applications > Applications > Develop**.
2. Click the application name or the **Edit** button for the application you want to update. The system displays the application and application files in Studio.
3. Click **File > Settings**.
   Studio opens a tab containing the Custom Application record for the current application.

4. Fill in the fields, as appropriate.

<table>
<thead>
<tr>
<th>Custom Application form</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Field</strong></td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Version</td>
</tr>
<tr>
<td><strong>Note:</strong> To publish the application in the ServiceNow Store, the version must conform to the application certification standards.</td>
</tr>
</tbody>
</table>

**Application Scoping**

<table>
<thead>
<tr>
<th><strong>Field</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope</td>
<td>[Read Only] Displays the unique application scope set during the creation process. You can change this value only by deleting and recreating the application with a new value. For more information about the protections offered, see Application scope.</td>
</tr>
<tr>
<td>Application administration</td>
<td>Select whether to protect sensitive application data by restricting how users acquire application-specific roles. See Application administration.</td>
</tr>
</tbody>
</table>

**Design and Runtime**

<table>
<thead>
<tr>
<th><strong>Field</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>JavaScript Mode</td>
<td>Select the JavaScript standard the application supports. Select <strong>ES5 Standards Mode</strong> to support features in ECMAScript 5th edition. Select <strong>Compatibility Mode</strong> to support earlier ECMAScript editions.</td>
</tr>
<tr>
<td>Runtime Access Tracking</td>
<td>Select how the application handles script access requests to resources in other applications. Select <strong>None</strong> to authorize all access requests to cross-scope resources without logging them. Select <strong>Tracking</strong> to log and authorize all access requests to cross-scope resources. Select <strong>Enforcing</strong> to log...</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Field</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>Field</td>
<td>access requests to cross-scope resources but require an administrator to authorize each request.</td>
</tr>
<tr>
<td>Restrict Table Choices</td>
<td>Clear the option to allow the application to see tables from other application scopes. Select the option to restrict design choices to only tables in the same application.</td>
</tr>
<tr>
<td>Subscription Management</td>
<td></td>
</tr>
<tr>
<td>Licensable</td>
<td>Specifies whether the application is tracked by the Subscription Management application.</td>
</tr>
<tr>
<td>Subscription requirement</td>
<td>Not applicable for ServiceNow customers who build custom applications for their own use. Used only by partners who sell and monitor the usage of resellable applications on the ServiceNow Store.</td>
</tr>
<tr>
<td></td>
<td>Specifies whether the application requires a separate subscription <strong>(Required)</strong> or is monitored only.</td>
</tr>
<tr>
<td>Subscription model</td>
<td>Not applicable for ServiceNow customers who build custom applications for their own use. Used only by partners who sell and monitor the usage of resellable applications on the ServiceNow Store.</td>
</tr>
<tr>
<td></td>
<td>Specifies how the Subscription Management application tracks usage. See Types of subscriptions.</td>
</tr>
<tr>
<td>Primary Menu</td>
<td></td>
</tr>
<tr>
<td>Menu</td>
<td>Select the application menu where you want to display modules. For more information about menus and modules, see Create an application menu.</td>
</tr>
<tr>
<td>End user access</td>
<td></td>
</tr>
<tr>
<td>User role</td>
<td>Select the user role required to access the application menu. For more information about user roles, see Create a role.</td>
</tr>
<tr>
<td>Short description</td>
<td>Enter a description of the application purpose or usage.</td>
</tr>
<tr>
<td>Logo</td>
<td>Select the image the system displays in the applications list and ServiceNow Store.</td>
</tr>
<tr>
<td>Application Files</td>
<td>View configuration records associated with this application in platform feature tables.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Dependencies</td>
<td>View or add tables or applications on which this application depends. The system automatically adds records to this list when you extend tables or when another application creates application files for this application. Add script-based dependencies. See Dependencies for custom applications.</td>
</tr>
<tr>
<td>Cross scope privileges</td>
<td>View or create cross-scope privilege records to determine which script operations and targets the system allows to run. See Cross-scope privilege record.</td>
</tr>
<tr>
<td>Design Access</td>
<td>View or specify which other applications have design access to tables or records in this application. See Application design access record.</td>
</tr>
</tbody>
</table>

5. Click **Update**.

**Switch between applications**

Application developers can switch between applications without leaving the Studio environment.

**Before you begin**

Before starting this procedure, you must create at least one custom application with its own application scope.

Role required: admin

**About this task**

The contextual development environment restricts some changes when the application files belong to another application. Switching to the application that owns the application files ensures that you change the proper application.

**Procedure**

1. Navigate to **App Options Menu > Switch Applications**.
   The system displays the list of applications.

2. Click the application you want to switch to.
   The system reloads Studio to display the selected application.

**Global application file management**

Once you create a globally scoped application in the ServiceNow Studio, you can add existing globally scoped files to it, remove files from it, or move application files between global applications.
Add a file from the global scope to a global application

Add existing globally scoped files into a selected global application. You can search for files in another globally scoped application by the update set name, table, or file name.

Before you begin
The Global App File Management plugin must be activated.
Role required: admin or a delegated_developer role that grants full access to the application.

About this task
When you select an application in the application picker, application files are automatically assigned to it. You cannot move an application file into or out of a scoped application. See Application scope.

Procedure
1. Navigate to System Applications > Applications > In Development.
2. To the right of the application name, click Edit in Studio.
3. In Studio, click File, then select Add Existing Files.
4. Use any of the following options to find existing globally scoped application files that you can add into the current globally scoped application:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search by update set</td>
<td>Search and filter application files that are associated with an update set.</td>
</tr>
<tr>
<td>Search by table</td>
<td>Search and filter application files that are associated with a specific table. For example, if you select Incident, the search returns application files that are associated with that table.</td>
</tr>
<tr>
<td>Search by name</td>
<td>Perform a custom name search for application files that contain a specified update name (sys_update_name) or an actual file name (sys_-name).</td>
</tr>
</tbody>
</table>

Note: You can view up to 250 files at a time. If you want to view more files, perform another search with more specific search criteria.

5. Click Continue to list application files that are associated with the selected update set, table, or name. You can filter the results as needed.
The system displays the filtered results but omits any files that are already associated with the current global application. The system displays the file type, application, when the file was last updated, and if it is a customized file.

6. Select the files that you want to add, and then click **Add**.
   The Confirm Changes dialog appears, stating that the files being moved are now considered customizable files and may appear on a skip list during an upgrade.

7. Click **Continue** to add the selected files.

**Results**
The system moves the selected files to the selected application from the global scope, and flags them as customizable files. For example, if you select the sys_ui_policy file, the associated sys_ui_policy_actions file is also added to the selected application, and flagged as customizable. You see error messages when errors occur in the add process, or a confirmation when your files are successfully added to the globally scoped application.

**Remove a file from a global application**
Remove selected files from the selected global application and return them to the previous global application that owned the file. If there is no previous owner, file is returned back to global scope. The removed files are still considered customized files.

**Before you begin**
This option is only available if the Global Application File Management plugin is active.
Role required: admin or a delegated_developer role that grants full access.

**Procedure**

1. Navigate to **System Applications > My Company Applications > In Development**.
2. To the right of the application name, select **Edit in Studio**.
3. In Studio, select **File > Remove Files**.
4. The system displays a list of the files in this global application. Select the file that you want to remove from this application.
5. Click **Remove**.
Move an application file between global applications

Move selected application files and their descendant files to a selected global application. When you select a global application, the selected application files are automatically assigned to it. You cannot move an application file into or out of a scoped application.

Before you begin
This option is only available if the Global App File Management plugin is active. Role required: admin or a delegated_developer role that grants full access.

Procedure
1. Navigate to the application file in a list or form view. For example, navigate to System UI > UI Policies to select UI policy files to move.
2. Select the check box beside each file you wish to move, and then select Move to Application in the Actions on selected rows choice list.
3. Select the global application.

![Move to application](image)

4. Click Move.

The selected application files and its descendant files are moved to the selected global application.

Review claimed or skipped global files
This topic explains what to do when one or more of your files has been skipped or claimed by another application.

During the development of a global application, warnings about skipped or claimed files may display on various progress screens. Skipped or claimed files can occur when the files in one application are claimed by other application(s).
These warnings display during the installation of an application from the Application or Git repositories or when you publish your application. When you see that your application files have been either skipped or claimed by another application, contact the team that owns the other application to determine the best course of action for both teams.

What are your options?
Only one version can be loaded for production so your options are:

• Decide with the other team from which application the file should be removed so that only one file is loaded for production.

• Use the Merge tool: If changes to files in both apps are needed in production, use the merge tool to pull the files into a new common application that serves both parties effectively.

To learn more see the Resolve conflicts topic.

Automatic recovery of draft records
Studio can maintain a version of any open existing record with unsaved changes. Users can recover unsaved changes when their user session ends unexpectedly due to network latency, session timeout, or service interruption.

Automatic recovery only applies to:

• Records open in Studio. The system does not save all draft records.

• Changes made in desktop (non-mobile) browsers. The system does not save draft records from mobile browsers.

• Changes made to existing records. The system does not save draft changes to new records.

• Records containing unsaved changes that are the most recent update to the record. The system discards draft changes when another user has updated the same record.

• Records for tables that extend the Application File [sys_metadata] table.

After the user re-establishes a session, Studio displays a message for each record with recovered changes.
For each recovered record, users can:
  • Continue editing and save the record.
  • Clear the changes from the recovery cache.

The system automatically clears changes from the recovery cache when a user:
  • Saves the record. The system removes the saved record from the recovery cache.
  • Confirms navigating away from a record without saving changes. The system removes the abandoned record from the recovery cache.
  • Reaches the recovery cache limit of 5 MB of changes. The system removes the record with the oldest update date-time stamp.

By default, automatic recovery is enabled for all Application File [sys_metadata] tables while working from Studio.

Administrators can configure automatic recovery properties to:
  • Disable or re-enable automatic recovery.
  • Specify a list of field types to exclude from automatic recovery.

Users can enable or disable automatic recovery as a user preference.

Related information
  User preferences

Auto recovery properties

Administrators can configure how Studio handles the recovery of draft records by navigating to Auto Recovery > Properties.

<table>
<thead>
<tr>
<th>Auto recovery properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Property</strong></td>
</tr>
<tr>
<td>Enable Auto Recovery</td>
</tr>
<tr>
<td>glide.ui.auto.recovery</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Auto recovery properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comma separated list of tables not supported for auto recovery</td>
<td>By default, automatic recovery is enabled for all Application File [sys_metadata] tables while working from Studio. Set this property to exclude specific tables from automatic recovery.</td>
</tr>
<tr>
<td>glide.ui.auto.recovery.unsupported.tables</td>
<td></td>
</tr>
<tr>
<td>Comma separated list of field types not supported for auto recovery</td>
<td>By default, automatic recovery supports all field types. Set this property to exclude certain field types from automatic recovery.</td>
</tr>
<tr>
<td>glide.ui.auto.recovery.unsupported.field.types</td>
<td></td>
</tr>
<tr>
<td>Comma separated list of field types to exclude from auto recovery</td>
<td>By default, automatic recovery is enabled for all field types. Set this property to exclude certain field types from automatic recovery.</td>
</tr>
<tr>
<td>glide.ui.auto.recovery.exclude.field.types</td>
<td></td>
</tr>
</tbody>
</table>

Auto recovery dictionary attribute

Administrators can configure how the Studio handles the recovery of draft records with a dictionary attribute.

Auto recovery properties

<table>
<thead>
<tr>
<th>Dictionary attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exclude Auto Recovery</td>
<td>Disables automatic recovery of draft records for this table.</td>
</tr>
<tr>
<td>exclude_auto_recovery</td>
<td></td>
</tr>
</tbody>
</table>

Related information

Dictionary attributes
Source Control integration

Enable application developers to integrate with a Git Source Control repository. Save and manage multiple versions of an application from a non-production instance.

Linking an application to Source Control enables all application developers on a non-production instance to:

- Import applications from a Git repository.
- Pull and apply remote changes from a Git repository.
- Commit all local changes on the instance to a Git repository.
- Create tags to permanently link to a given version of an application.
- Create branches to maintain multiple versions of an application simultaneously.

Integration requirements

To link an application to source control:

- The user must have the admin role.
- The non-production instance must have network access to the Git repository.
- Each application must be within its own Git repository.
- The repository user credentials must grant read and write access.

⚠️ Note: All application developers on the instance share a single set of credentials per repository.

Options available from ServiceNow Studio

After linking an application to Source Control, application developers can use ServiceNow Studio to manage the repository. From Studio, developers can:

- Edit the application repository credentials.
- Commit all local changes on the instance.
- Apply remote changes from the repository.
- Create a branch.
- Switch branches.
- Import an application from a remote repository.

Source Control integration does not support managing applications on a production instance. Instead, you can manage applications on a production instance using the application repository, an update set, or the ServiceNow
Options available from a Git repository
The ServiceNow platform offers limited support for modifying linked application files outside of an instance. From Git, developers can:

- Move application files to a different Git directory structure.
- Edit application files outside of ServiceNow Studio.

The system generates a properties text file called `sn_source_control.properties` at the root level of the repository. To move application files to a different Git directory structure, application developers can set the `path` parameter to specify the subfolder path containing their application files. For example, if you moved your application to the `src/app` subfolder, set the `path` to `path=src/app`.

The system generates a `checksum.txt` file in the Git repository to determine if any application files have been changed outside of Studio. When the checksum value from the file matches the current checksum value, the integration skips the validation and sanitization process. When the checksum values do not match, the integration validates and sanitizes the application files as part of the Source Control operation. The sanitization process:

- Creates upgrade log entries for each sanitization action taken.
- Removes unsupported folders and files from the repository.
- aborts all Source Control operations when a system application file fails XML schema validation. For example, if a database dictionary record fails XML schema validation, the system aborts all operations.
- Skips the current Source Control operation when a non-system application file fails XML schema validation.

The Source Control integration sanitizes only content within the application path listed in the `sn_source_control.properties` file. Repository content outside the application path is ignored.

MID Server support
Use an existing MID Server to connect to a Source Control repository. Linking or importing an application through a MID Server enables access to repositories behind a firewall.

Production deployment tips
When you develop customizations to applications on the ServiceNow® platform, you deploy them via the application repository to a production instance. This
topic examines and provides cautions for the tradeoffs between installing an application from the application repository versus Git repository with source control.

**Overview**

Technically, you can still “deploy” an application from a Git repository to a production instance using source control. This can have unintended consequences.

**Glossary of terms**

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metadata or application files</td>
<td>The sys_metadata records that define configuration in ServiceNow and are packaged in an application. These records alter the behavior of the instance but do not contain data such as incident or CMDB records. (See Note below)</td>
</tr>
<tr>
<td>Scoped applications</td>
<td>ServiceNow applications that restrict allowing only updates and operations within the boundary of the scope. This mechanism is used for most new development.</td>
</tr>
<tr>
<td>Global applications</td>
<td>Global applications are developed in the legacy global scope. Work is often done in this scope to customize existing ServiceNow applications such as IT Service Management (ITSM).</td>
</tr>
<tr>
<td>Application repository</td>
<td>Applications are typically published here for deployment in production instances. Although the application repository has separate entitlement rules, it operates similarly to the ServiceNow Store.</td>
</tr>
<tr>
<td>ServiceNow Store</td>
<td>Repository for third-party (vendor) applications as well as ServiceNow published applications. Most customers do not publish to the Store, but often install applications from it.</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Update sets</strong></td>
<td>Standard method of packaging customizations for deployment in each successive instance. They contain the incremental collection of insertions, updates, and deletions.</td>
</tr>
<tr>
<td><strong>Delta loading</strong></td>
<td>The most efficient method of loading because it changes only from source control rather than earlier uninstall/reinstall methods.</td>
</tr>
<tr>
<td><strong>Schema</strong></td>
<td>Definition of tables and columns in the tables.</td>
</tr>
<tr>
<td><strong>Rollback</strong></td>
<td>Administrators can roll back the last installation of a selected application. A rollback removes all code, table, and file updates from the initial installation.</td>
</tr>
</tbody>
</table>

**Note:** The `sys_metadata` table is the parent table of all application files in the ServiceNow platform using the table inheritance model. You can view summary information for metadata by visiting the parent table or tables that extend directly or indirectly as indicated by the Extends table(super_class) field on the Table(sys_db_object record). You can also see the whole schema by visiting the Table(sys_db_object) form for the `sys_metadata` table and selecting the Show Schema Map related link at the bottom of the form. The schema is large and so takes some time to render.
Installation location

When you install source control, it facilitates the ongoing development of a custom application. Therefore, the application is managed as an “In development” application in the Custom Application [sys_app] table rather than as an “Installed” application in the Store Application [sys_store_app] table. Both tables are extensions of sys_scope so they both provide the same
protections and restrictions as the scope. So when you search for the installation of a source control deployed application, refer to the System Application [sys_app] table and the **in development** section of the Application Manager page.

You cannot have a sys_app record on the instance while deploying that same application from the ServiceNow Store or application repository. The two deployment models are mutually exclusive. If at any point the deployment model changes, the sys_app record must be converted to a sys_store_app record first. You can contact ServiceNow Support for help with performing that operation.

**Delta loading**

Prior to the ServiceNow Paris release, application installation from source control always removed and reinstalled the entire application when triggered, including the Apply Remote Changes function. With **Delta loading**, now only the changes update, simplifying the process considerably.

The Delta loading process loads the changes from source control incrementally. When you apply remote changes, you don’t drop existing tables or columns unless they were removed from the repository. This preserves the data for tables and fields that continued to be present.

Note: The `glide.source_control.allow_delta_loading_in_scopedapp` property allows you to disable Delta loading in Paris; however, this will revert to the more destructive behavior of removing and reinstalling the application. Global applications in Paris always use Delta loading.

Below is a table of the different expected outcomes in an instance using Delta loading.

<table>
<thead>
<tr>
<th>Application type</th>
<th>Install source</th>
<th>Schema present in package</th>
<th>Schema contains data</th>
<th>Claim by another app (Global)</th>
<th>Expected outcome for data and schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scoped</td>
<td>Application repository or Store</td>
<td>Yes</td>
<td>Yes/No</td>
<td>N/A</td>
<td>Preserved</td>
</tr>
<tr>
<td>Scoped</td>
<td>Application repository or Store</td>
<td>No</td>
<td>Yes/No</td>
<td>N/A</td>
<td>Preserved</td>
</tr>
<tr>
<td>Application type</td>
<td>Install source</td>
<td>Schema present in package</td>
<td>Schema contains data</td>
<td>Claim by another app (Global)</td>
<td>Expected outcome for data and schema</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------</td>
<td>----------------------------</td>
<td>----------------------</td>
<td>-------------------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>Scoped</td>
<td>Source control</td>
<td>Yes</td>
<td>Yes/No</td>
<td>N/A</td>
<td>Preserved</td>
</tr>
<tr>
<td>Scoped</td>
<td>Source control</td>
<td>No</td>
<td>Yes/No</td>
<td>N/A</td>
<td>Removed</td>
</tr>
<tr>
<td>Global</td>
<td>Source control, Application repository, or Store</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes/No</td>
<td>Preserved</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Preserved (1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Preserved (2)</td>
</tr>
<tr>
<td>Source control</td>
<td></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Removed (3)</td>
</tr>
<tr>
<td>Application repository</td>
<td></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Preserved</td>
</tr>
</tbody>
</table>

**Note:** 1: While the database schema and the data are preserved, they will be moved into the default global application.

**Note:** 2: While the database schema and the data are preserved, they will be moved into the global application which previously held claim to these files if installed. Otherwise, they will move into default global application.

**Note:** 3: Applicable only on custom columns with `u_` prefix. ServiceNow platform-authored columns are not dropped.

When switching branches in source control for a scoped application, use extreme caution in a production environment. If the target branch is missing schema elements found in the current branch, the related schema is dropped, destroying any data it contains. (Global applications do not drop schema when data is present.)

Just as with update sets, only a subset of the incremental changes needs to be applied with Delta loading. Unlike update sets, the application package represents the complete application. Files that are absent from the new package are deleted. This can alter functionality and delete data. Update sets
and applications upgraded from the application repository or ServiceNow Store must have an explicit **DELETE** payload to remove a file or drop a schema.

If application files are being generated dynamically in any fashion, the next install/apply remote changes the operation of an application deletes those records. They are considered absent from the incoming application package. If you stash local changes, the application files may be recoverable by a stash commit, but if data is lost as a result of the changes, the data is not recovered.

**Note:** Removing or suppressing sys_update_xml records prevents them from being removed by Delta loading; however, this action can have other severe or undesirable results.

Direct edits to the repository, especially to remove files, can have significant ramifications, including loss of data and cascading deletes. Perform this action with care.

**Author elective and customer updates**

When you install an application from the application repository or ServiceNow® store, you can set a series of properties to define the behavior of delete and choice processing. These kinds of choices are called an "author-elective" feature.

**Overview**

You can find details on these properties in the Skipped records that occur during application installation topic. With these properties, you can opt in and out of deletes and choice updates depending on whether you are using your own or a third-party application.

When you install from source control, however, these records do not skip, except when a global application file is claimed by a different global application. Other than `com.glide.apps.include_my_deletes` and `com.glide.apps.include_global_deletes` that disable the processing of the author_elective_update folder altogether, those properties are not effective for source installed apps.

**Note:** An “absent” file detected in Delta loading for source control is vastly different than a Delete payload housed inside the author_elective_update folder. Author_elective_update properties do not prevent Delta loading in source control from deleting the file.

Similarly, update sets protect customizations that you make in an instance against incoming changes that force a preview decision. Before you commit an update set, a preview must be run to attempt to detect collisions. You must address all preview problems before committing the changes. Source control
may ask you to stash a local change, but the outcome of the installation is to load what is present in the source even if a change had been made locally.

Loading what is present in the source is challenging when properties must have different versions based on the target of the installation. For example, it’s difficult to resolve when a property containing an integration URL differs based on instance production role. The `is_private` flag is effective with a source control installation and does not overwrite the property if set, mitigating this concern.

### Roll back, back out, and uninstall

The application installation from the application repository is recorded for rollback, which means as an administrator, you can roll back the last installation of a selected application. When you roll back an application, you remove all code, table, and file updates from the initial installation.

Source control installations are not recorded for rollback, which means that the rollback feature to undo an application repository, store installation, or an in-family upgrade is not available for source control installations. They also cannot be uninstalled in the same sense that ServiceNow® Store applications can be (that is, with the option to retain tables and columns). They can be deleted by deleting the `sys_app` record. However, for scoped applications, this destroys the underlying schema and its data.

You cannot delete global applications until all application files are moved to another global application. Deleting a `sys_app` record for an application also creates `sys_update_xml` records with `DELETE` payloads. These payloads can generate skips if the same application is then installed via the application repository or ServiceNow Store.

To avoid these skips, the `sys_update_xml` records must be manually deleted before the application is installed from the application repository. Installing from source control also does not have the Back out option that update sets contain to remove a subset of new changes. That means that recoverability in this model is limited to installing a corrected version (older or newer) of the application on top of the current version. This action restores only metadata or configuration. Data lost as a result of a poor installation must be recovered from a database restore.

### Upgrade history

Upgrade history records and history line dispositions are generated as part of the source control installation providing a record of the install.

The source control installation is more aggressive and generally doesn’t skip unless there is a global claim or `is_private` protection on a property. Global applications also protect columns containing data, or those from the ServiceNow base installation, from being removed.
Packaging system-generated changes

When an application is published to an application repository, its current state on the instance is packaged, including any system-generated changes. When you install this version in higher instances, these changes are included as part of an application package.

Excluding changes

The ServiceNow® Rome release introduced a new feature in source control to exclude system-generated changes to the application files. In other words, these changes don’t generate `sys_update_xml` during the commit. This allows only user changes, which generate `sys_update_xml` to be committed to the Git repository. To learn more about the feature see the Commit changes topic.

If you exclude the system changes when you commit to the Git repository, then installing applications from source control will be missing the changes as well.

Development considerations

When you are developing code, consider some of these suggestions for the most efficient performance.

Production instances shouldn’t publish to the Git repository

To protect the integrity of production, you shouldn’t push changes to publish to the Git repository from production even though technically you can open and commit changes in the application in ServiceNow® Studio. The “Can Edit Application in Studio” option can be disabled in production on the `sys_app` record. But it resets at the next source code operation that triggers an update (that is, when you apply remote changes or switch branches).

Dependencies are not installed when installing an application from the Git repository

Source-controlled applications do not automatically install or upgrade if they are listed as the dependency of your application. Instead, the customer must install or upgrade the source-controlled application on its own. The customer is also responsible for installing and upgrading the applications in the correct order.

Tight control on source control privileges in production

The source code operations are generally available to those with development privileges in the instance. Since production environments shouldn’t have ongoing development, be sure to tightly control admin and delegated development privileges to avoid data loss and other serious consequences.
Available source control operations

The source control integration primarily supports operations from Studio, but can also support some operations directly from the GIT repository.

### Available source control operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
<th>Available from</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import from Source Control</td>
<td>Imports an application from the repository to the local instance.</td>
<td>Studio</td>
</tr>
<tr>
<td>Link to Source Control</td>
<td>Allows developers to manage application changes from a GIT repository.</td>
<td>Studio</td>
</tr>
<tr>
<td>Edit Repository Configuration</td>
<td>Updates the GIT repository user credentials.</td>
<td>Studio</td>
</tr>
<tr>
<td>Apply Remote Changes</td>
<td>Updates the local version of the application to match the repository version.</td>
<td>Studio</td>
</tr>
<tr>
<td>Commit Changes</td>
<td>Updates the repository version of the application to match the local version.</td>
<td>Studio</td>
</tr>
<tr>
<td>Stash Local Changes</td>
<td>Removes and saves local changes for later work.</td>
<td>Studio</td>
</tr>
<tr>
<td>Switch Branch</td>
<td>Updates the local version of the application to match the repository branch version.</td>
<td>Studio</td>
</tr>
</tbody>
</table>
| Create Branch           | Creates a branch in the repository to save a different version of the application. | • Studio  
                          |                                                                             | • GIT repository          |
| Create Tag              | Creates a tag in the repository to link to a particular application version. | • Studio  
                          |                                                                             | • GIT repository          |
| Manage Stashes          | Allows developers to apply or delete stashed changes.                        | Studio                  |
| Create repository       | Creates a repository to store application changes                            | GIT repository          |
| Create credentials      | Creates credentials to the repository.                                       | GIT repository          |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**Available source control operations (continued)**

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
<th>Available from</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grant access to repository</td>
<td>Provides read and write access to the repository tied to a specific set of credentials.</td>
<td>GIT repository</td>
</tr>
</tbody>
</table>

**Import application or application-customization from source control**

Import an application or application-customization from a source control repository to continue developing it on this instance.

**Before you begin**

- Role required: admin
- Verify that the non-production instance has network access to the Git repository.
- Verify that the repository contains a valid application.
- Ensure that users add the email address to their respective Users Table (ServiceNow sys_user) record that they use in their commits to the Git repository.
- Learn more about application-customizations Managing application-customizations.

**About this task**

The source control integration does not support importing an application on a production instance. Instead install applications on a production instance from the application repository, an update set, or the ServiceNow Store.

**Procedure**

1. Navigate to **System Applications > Studio**.
   The system displays the Welcome to Studio page.
2. Click **Open Studio > Go**.
   The system opens Studio and the Switch Applications window.
3. Click **Import from Source Control**.
   Studio displays the Import from Source Control fields.
4. Enter the following field values.

### Import from source control fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network protocol</td>
<td>Https or ssh credential type that enables secure channel data exchange.</td>
</tr>
<tr>
<td>URL</td>
<td>The URL to the Git repository where the application files reside.</td>
</tr>
</tbody>
</table>

**Note:** If the Git repo URL for SSH provided by your Git server does not work, check with your Git server owner or provider for the correct URL. There may be additional specifications such as scheme protocol prefixes, port numbers, and so on, required for your Git repo URL to function.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Note:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Credential</td>
<td>Select the credential for your Git repository. (See Getting started with Credentials.)</td>
<td>If you select the ssh network protocol, enter a valid credential of the SSH Private Key type. If you select the https protocol, enter a valid credential of the Basic Auth Credentials type.</td>
</tr>
<tr>
<td>Branch</td>
<td>The repository branch to work on within the application.</td>
<td>The default branch is named after your instance. If you do not choose a name, the branch defaults to master.</td>
</tr>
<tr>
<td>MID Server Name</td>
<td>Select an existing MID Server to link to a Git repository stored behind your corporate firewall.</td>
<td>Use a separate MID Server to prevent conflicts with Discovery activities.</td>
</tr>
<tr>
<td>Default email</td>
<td>The committer email address is defined by the sys_user record if available. But if a committer's sys_user record email field is empty, the system generates an alternate email (<a href="mailto:username@instancename.service-now.com">username@instancename.service-now.com</a>). You can also enter a default email address and change it later. To use that default email address in all cases, select the check box.</td>
<td>All application developers on the instance share the credential used to link a Git repository to an application.</td>
</tr>
</tbody>
</table>

**Note:** All application developers on the instance share the credential used to link a Git repository to an application.

5. Click **Import**.

The system compares the checksum in the `checksum.txt` file to current checksum. When the checksum values match, the integration skips validation and imports the application. When the checksum values do not match, the integration first validates and sanitizes the application files before importing them.

6. Click **Select Application**.

Studio displays the application as a new choice in the Switch Applications window.
What to do next

- Review the upgrade logs for any sanitization applied to application files during the import.
- Select the imported application to edit it.

Related information

MID Server
Getting started with credentials

Link an application or application-customization to source control

Linking an application or application-customization to source control allows application developers to manage changes from a Git repository.

Before you begin

- Role required: admin
- Learn more about Managing application-customizations.
- Create a dedicated Git repository for the application. For increased security, enable multi-factor authentication for the Git repository.
- Generate an access token that the source control integration can use instead of a password and multi-factor authentication passkey. Search for personal access token on GitHub or GitLab.
- Restrict permissions on the access token to allow read and write access to the Git repository.
- Verify that the non-production instance has network access to the Git repository.
- Ensure that users add the email address to their respective Users Table (ServiceNow sys_user) records that they use in their commits to the Git repository.
- Learn more: Migrate completed update set history to Source Control

About this task

The source control integration does not support linking to an application or customization on a production instance. Instead, install applications on a production instance from the application repository, an update set, or the ServiceNow Store.
Procedure

1. Open the application you want to link to source control in Studio.

2. Navigate to **Source Control > Link to Source Control**.
   Studio displays the Link to Source Control dialog box.

3. Enter the connection details for the Git repository.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network protocol</td>
<td>Https or SSH credential type that enables secure channel data exchange.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>URL</td>
<td>The URL to the Git repository where you want to save application files. For SSH protocol, use command to generate private key ssh-keygen -t rsa -m PEM -b 4096 -C “email@address”.</td>
</tr>
<tr>
<td>Credential</td>
<td>The credential to be used with the selected protocol. See Getting started with Credentials to learn more about creating credentials.</td>
</tr>
<tr>
<td>Branch</td>
<td>The repository branch to work on within the application.</td>
</tr>
<tr>
<td>MID Server name</td>
<td>The name of the existing MID Server to link through.</td>
</tr>
<tr>
<td>Default email</td>
<td>The committer email address is defined by the sys_user record if available. But if a committer’s sys_user record email field is empty, the system generates an alternate email (<a href="mailto:username@instancename.service-now.com">username@instancename.service-now.com</a>). You can also enter a default email address and change it later. To use that default email address in all cases, select the check box.</td>
</tr>
<tr>
<td>Commit Comment</td>
<td>An optional description of the repository or application.</td>
</tr>
</tbody>
</table>
Note: All application developers on the instance share a single set of repository credentials.

4. Click Submit.
   The system validates the connection and user credentials and displays a success message.
   All application developers on the instance can use the linked Git repository to manage changes.

Related information
   Getting started with credentials

Using MID Server with source control
The ServiceNow® MID Server enables communication and the movement of data between a ServiceNow instance and external applications, data sources, and services.

How bundle files work with MID Server
The .bundle file helps source control function with a MID Server. A bundle file is the way Git packages a local repository in a single file. This makes sharing or moving the repository simpler and more streamlined. The file is then sent to the MID Server, which passes it on to the remote repository.

The outgoing.bundle (commit operations) and incoming.bundle (apply remote changes) are attached to the MID Server attachment table [ecc_agent_attachment] for any request that goes to the MID Server. The outgoing.bundle is created on the instance while the incoming.bundle is created on the MID Server.

After an operation completes successfully, the bundle file is “promoted” into a golden.bundle that is attached to the Repository configuration table [sys_repo_config]. There is one golden bundle per repository and it’s used to initialize the repository on a node that has not performed any Source Control operations yet.

The Auto Flush tool [sys_auto_flush] is a “table cleaner” that removes any ecc_agent_attachment record older than 30 days by default. This action removes the corresponding attachment as well, as they are not necessary after the cleanup operation.

Note: If you want to customize the cleanup operation for bundle files, it is best to create and configure an additional auto-flush tool rather than changing any of the default settings.
The bundle files are kept on the MID Server and then saved to the Import directory on the MID Server.

On the MID Server, the bundle file is saved in the Import folder. This folder is under the user directory defined by the system property (user.dir), which users can configure. The bundle file is removed as part of the system flushing at the end of every operation.

**Working with the MID Server**

- Avoid conflicts with Discovery and create files for the system attachment [sys_attachment] table: **MID Server**
- Learn about system properties restrictions: **Configure attachment system properties**

**Migrate completed update set history to Source Control**

When linking to Source Control, this feature allows application developers the choice of migrating the information in completed update sets to Source Control history.

**Before migrating**

Make sure that you have fulfilled these criteria before attempting to migrate your update sets:

- Role required: admin
- Read the Link an application or application-customization to source control topic
- Complete any update sets for your application that you want to export as Source Control history.
- Export the completed update set if you want to preserve it.

When you link an application to Source Control, the update sets and customer update records are deleted. After you link to Source Control, if the application has any completed update sets, you will be asked to make a choice in the dialog box below.

- If you select “Yes, do retain update set history as commits”, the update set history is preserved as Source Control commits.
- If you select “No, do not retain update set history as commits,” they are not preserved as commits.

Regardless of which option you select, if you select **Continue**, the Link to Source Control operation starts, and all completed update sets and all Customer...
Update records are deleted. If you need to complete any additional update sets or choose not to continue, select **Cancel**.

For every completed update set with updates to the application that you are linking to Source Control, commits are generated automatically by the system based on the `sys_update_xml` records in the update sets. The commits are ordered by the `sys_recorded_at` timestamp. For Global applications: Any `sys_update_xml` records that belong to the application and are part of a completed Global update set are captured as historical commits.

When the Link to Source Control operation is complete, the most recent commit is the current state of your application in its entirety. You can view historical commits in your Git repository or by clicking the Source Control menu option and selecting **View History**. Updates are separated into multiple commits:

- If there are updates for a file that are out of order between different update sets.
- If an update set contains multiple update records for a single file.

The commits for an update set are split into multiple commits ([Historical Commit 1], [Historical Commit 2]...) to represent each update. This is done so that each file has an ordered history of updates.

⚠️ **Warning:** Any commit prefixed by [Historical Commit] is generated solely to display its history. Do not attempt to check out these commits in the development process as they do not necessarily represent a stable snapshot of the application.

The `author_elective_update` folder is not created until the initial commit. That means that in the initial commit you might see files such as `sys_choice` files being renamed and moved from the update folder to the `author_elective_update` folder. Any files that are deleted from update sets in historical commits are
deleted, and not moved to the `author_elective_update` folder as they would be for actual commits. During the initial commit, DELETE payloads are also created for any DELETE `sys_update_xml` records that were deleted as part of completed update sets.

Example commit message:

<table>
<thead>
<tr>
<th>[Historical Commit 1]</th>
<th>&lt;Name of update set that this commit belongs to&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description:</td>
<td>&lt;Description of update set that this commit belongs to&gt;</td>
</tr>
<tr>
<td>Update Set was completed on</td>
<td>&lt;date&gt;</td>
</tr>
<tr>
<td>Update Set was completed by</td>
<td>&lt;sys_user user_name &gt; &lt;sys_user email&gt;</td>
</tr>
</tbody>
</table>

Additional values from `sys_update_set` record (see **Customization** section below)

Batch update set information (See the **Batch update sets** section below) }

**Batch update sets**

If an update set is part of a batch update set, that information is appended to the commit message in the following format, with the highest number being the Batch Base:

```json
{
  "1": {
    "parent": "<name of parent update set>",
    "description": "<description of parent update set>"
  },
  "2": {
    "parent": "<name of parent 1’s parent update set> ",
    "description": "<description of parent 1’s parent update set> "
  }
}
```

**Customization**

You can add additional fields to include in the commit message by adding a `glide.source_control.historical_commit_fields` property. The value is a comma-separated list of fields the user wants to include from `sys_update_set` XML fields. Spaces and invalid or misspelled field names are ignored. This property is used for all applications that are linked to Source Control from the instance if the committer chooses to retain update set history.
Note: If the value of a field references another table or sys_id, only the value of the field is added. For example: sys_id for a user instead of the name of the user.

XML example:

```xml
<sys_update_set>
  <application display_value="test_app2">a3885a8735be720107c7899b8b601c7d8</application>
  <sys_created_by display_value="System Administrator">6816f79cc08016401c5a33be04bde41</sys_created_by>
  <sys_created_on>2021-04-22 22:14:38</sys_created_on>
  <sys_mod_count>1</sys_mod_count>
  <sys_updated_by display_value="System Administrator">6816f79cc08016401c5a33be04bde41</sys_updated_by>
  <sys_updated_on>2021-04-22 22:14:38</sys_updated_on>
  <is_default>false</is_default>
</sys_update_set>
```

Value of the property:

```
<glide.source_control.historical_commit_fields>
  glide.source_control.historical_commit fields
<is_default,sys_created_by,
  sys_id>
```

Result in commit message:
Edit a Git repository configuration

You can edit a Git repository to change the network protocol selection, credentials or other field entries.

Before you begin
Role required: admin

Procedure
1. In Studio, select Edit Repository Configuration.

2. Choose https or ssh Network protocol and enter the URL address of your repository.
3. Change your credential or MID Server name if you wish.

!!! Note: If you have no MID server name, you can select a new one from the drop-down list. If you choose a new MID server, **Apply remote changes** in the Source Control menu before making any further Source Control operations to avoid errors.

4. Default email field: The committer email address is defined by the sys_user record if available. But if a committer's sys_user record email field is empty, the system generates an alternate email (username@instancename.service-now.com). You can also enter a default email address and change it later. To use that default email address in all cases, select the check box.

5. Click **Save**.

**Apply remote changes**

Application developers can pull changes from a linked GIT repository to apply remote changes to the local instance.

**Before you begin**

- Role required: admin
- Link an application or application-customization to source control
Procedure

Navigate to **Source Control > Apply Remote Changes**.

The following operations occur:

- The system fetches the most recent changes from the remote repository.
- The system applies the remote changes to the instance.
- The system identifies any change conflicts requiring resolution.

If there are conflicts, the system displays the **Resolve Conflicts** window.

Delta loading is enabled by default in sys properties so your data isn't removed. You can disable this feature if you want data automatically deleted.

**What to do next**

Resolve any change conflicts.

**Commit changes**

Application developers can commit their changes on the instance to the linked Git repository. You can either select a few changes to commit, or commit all changes on the instance at once.

**Before you begin**

- Role required: admin
- Link an application or application-customization to source control

**Procedure**

1. Navigate to **Source Control > Commit Changes**.

   The system displays the **Select files to commit to source control** window. The file changes from all the updates sets display. By default, the file changes from the current update set display.
2. Select the file changes you wish to commit.

3. To include untracked changes, select the **Include changes not tracked via the Customer Update [sys_update_xml] table** check box.

   - The default for this check box is set via the `glide.sourcecontrol.default_commit_mode` property.
     - Property can be set to **include_untracked** or **exclude_untracked**.
     - The **include_untracked** mode commits the updates to the application that do not generate sys_update_xml records, as well as any user-selected updates.
     - The **exclude_untracked** mode commits only updates selected by the user in the **Select files to commit to source control** dialog.
   - The base system setting for the property is **exclude_untracked**.
   - Prior to the ServiceNow Rome release, only the **include_untracked** mode is used.
   - To hide the check box and use the value of the `glide.sourcecontrol.default_commit_mode` property, create the `sn_devstudio.vcs.allow_commit_mode_selection` property and set it to false.
   - Checking this check box may incur a performance penalty.
Note:
Commits always occur in `include_untracked` mode in the following cases:

- Linking to Source Control for the first time. (To learn more, see [Link an application or application-customization to source control.](#))
- Publishing an application that's linked to Source Control from ServiceNow Studio. (To learn more, see [Publish an application from ServiceNow Studio when linked to Source Control.](#))
- Selective commit mode is disabled.

4. Click **Continue**.

5. In **Commit comment**, enter a comment for the changes.

6. Click **Commit Files**.
   The following operations occur:
   - The system identifies all local changes.
   - The system commits all local changes to the remote repository.

   Note: For list of known files that don’t have customer update records and are untracked, see **Customer updates table**

---

**Stash local changes**

Application developers can remove and save changes locally to apply them later.

**Before you begin**

- Role required: admin
- Link an application to source control
- Change one or more application files

**About this task**

Stashing changes removes them from the current application and saves them for a developer to later apply or delete.

**Procedure**

1. From Studio, navigate to **Source Control > Stash Local Changes**.
   The system displays a list of locally changed files.

2. Enter your description.
3. Click **Stash Local Changes.**  
The system saves the current changes and displays a success message.

![Stash Local Changes window](image)

### What to do next
- Close dialog
- Manage stashes

To learn more see [Getting started with credentials](#)

### Switch branch

Application developers can switch to a different repository branch to work on another version of the application.

#### Before you begin
- Role required: admin
- GIT repository with one or more available branches.

#### Procedure

1. Navigate to **Source Control > Switch Branch.**  
The system displays the Switch Branch window.

2. **Optional:** If there any local changes on the instance, you can save or discard them.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save local changes</td>
<td>Saves local changes before switching to an alternate branch. You can later merge or discard the saved changes.</td>
</tr>
<tr>
<td>Discard local changes</td>
<td>Permanently deletes all local changes before switching to an alternate branch.</td>
</tr>
</tbody>
</table>
Note: Use caution when discarding local changes. Since all application developers share repository credentials, there is no way to discard just one set of user changes. Note you cannot later restore discarded changes.

3. Select the branch you want to switch to.

4. Click **Switch Branch**. Studio updates the local application to match the branch version from repository.

Create branch

Application developers can create a branch to work on a new version of an existing application.

Before you begin

- Role required: admin or sn_group_creator.app_creator
- Link an application or application-customization to source control

Procedure

1. Navigate to **Source Control > Create Branch**. Studio opens the Create Branch window.

   ![Create Branch Window](image)

2. Enter the **Branch Name**.

3. Optional: To create a branch from a tag, click the **Create from Tag** drop-down list and select an existing tag.

4. Click **Create Branch**. Studio creates the branch.
5. Click **Close**.

**What to do next**
Commit changes to the new branch.

**Set default branch**
Set a default branch when you want to use a branch other than main for new changes or for your main development repository.

**Before you begin**
- Role required: admin
- Link an application or application-customization to source control

**Procedure**

1. Follow the steps to **Add a system property**.

2. Add the `glide.source_control.default_branch_name` property, and specify the default branch name of the GIT source control repository to work from (pull requests, code commits, etc.). Application developers' work is managed from and saved into the default branch if not otherwise specified. If not changed, this value defaults to `sn_instances/<instance_name>`.

**Manage stashes**
Application developers can apply or delete stashed changes from Studio.

**Before you begin**
- Role required: admin
- Link an application to source control.
- Stash one or more application file changes.
Procedure

1. From Studio, navigate to Source Control > Manage Stashes.
   The system displays a list of locally stashed changes.
2. Click the action next to the stash you want to manage.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply</td>
<td>Commits the stashed changes to the application and checks for conflicts.</td>
</tr>
<tr>
<td>Delete</td>
<td>Removes the stashed changes.</td>
</tr>
</tbody>
</table>

Resolve conflicts

Application developers can choose which application file version to use when applying remote or stashed changes.

Before you begin

- Role required: admin
- Link an application to source control
- Apply a stashed change

About this task

Conflicts occur when there are multiple change versions of the same application file: one set of changes in the remote or stashed version and another set of changes in the local version. Studio requires developers resolve conflicts before applying remote or stashed changes.

Procedure

1. From Studio, apply remote or stash changes.
   If the system identifies a conflict, it displays the Resolve Conflicts dialog.
2. Select how to resolve the conflict.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select an action</td>
<td>Apply or discard all stashed changes. Go to Step 3.</td>
</tr>
<tr>
<td>Manually merge changes</td>
<td>Individually select which changes to apply. Go to Step 6.</td>
</tr>
</tbody>
</table>

3. If you want to apply or discard all stashed changes, select an Action.
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Take Stashed Changes</td>
<td>Applies the application file version from the stashed changes.</td>
</tr>
<tr>
<td>Discard Stashed Changes</td>
<td>Applies the application file version from the most recent pull from the repository.</td>
</tr>
</tbody>
</table>

4. Click **Apply Stashed Changes**. The system applies the selected changes.

5. Click **Close Dialog**.

6. If you want to merge the conflicting changes, click **Manually Apply**. The system displays a list of version differences by field.

7. Select the field values you want the merged application file to have.

8. Click **Save Merge**. The system applies the selected changes.

**View commit history**

Application developers can view the commit history of applications linked to a source control repository.

**Before you begin**

- Role required: admin
- An existing link to a GIT repository

**Procedure**

1. Navigate to **Source Control > View History**. The system displays the History window.
2. Select the commit sort order type.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>Sort by commit date.</td>
</tr>
<tr>
<td>Committer</td>
<td>Sort by user name.</td>
</tr>
</tbody>
</table>

3. Select the sort order direction.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Descending</td>
<td>Sort dates from the most recent to oldest date. Sort user names reverse-alphabetically from Z to A.</td>
</tr>
<tr>
<td>Ascending</td>
<td>Sort dates from the oldest to most recent date. Sort user names alphabetically from A to Z.</td>
</tr>
</tbody>
</table>

The system sorts commits by the selected sort order.

4. Select a commit.
   The system displays the commit details for the selected commit.

5. Review the commit details.
Commit Details

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Committer</td>
<td>The user who committed the change.</td>
</tr>
<tr>
<td>Date</td>
<td>The date-time stamp of the commit.</td>
</tr>
<tr>
<td>SHA-1</td>
<td>The secure hash value identifying this commit in the repository.</td>
</tr>
<tr>
<td>Message</td>
<td>The commit message associated with this commit.</td>
</tr>
<tr>
<td>Files</td>
<td>The list of application files changed in this commit.</td>
</tr>
</tbody>
</table>

6. Close the History window.

Move application files in a GIT repository

Move application files linked to source control to any folder of the repository. Allow application developers to store supporting content such as automated tests in the same repository as the applications they support.

Before you begin

- Link an application or application-customization to source control
- Role required: Source control credentials with write access

About this task

Linking an application to source control generates a properties text file called `sn_source_control.properties` at the root level of the repository. The properties file specifies the folder containing the application files. The integration tracks changes to these application files by generating a `checksum.txt` file. When the checksum matches, the integration skips the validation and sanitization process. When the checksum does not match, the integration validates and sanitizes the application files as part of the source control operation. The integration ignores all repository content outside the application path.

⚠️ Note: You can set system properties `glide.source_control.checksum_required` to enable optional checksum validations and sanitizations and `glide.source_control.checksum_quick_install` to bypass sanitization steps on checksum matches. See Available system properties for more information.

Procedure

1. Login to source control repository linked to the application.
2. Create the destination folder where you will move the application files.
Example
For example, create the folders src/app.

3. Move the folder containing your application files to the destination folder.

Example
For example, move the folder demo_my_app to src/app.

4. Navigate to the root level of the repository.

5. Open the sn_source_control.properties text file in a text editor.

6. For the path= property, enter the folder path where you moved the application files.

Example
For example, enter path=src/app.

7. Save the properties file.

What to do next
Login to your instance and perform source control operations from Studio.

Collision avoidance
Avoid modifying an application file across different update sets to ensure seamless experience during the commit process.

When the logged in user opens an application file that was modified in an update set different from the user’s current update set for the corresponding application:

• Collision is detected in the application file and a warning message is displayed.
• Logged in user to prompted to choose an update set.
• Read-only protection policy is applied to the application file.
User can make changes to the application file only after selecting the required update set.

Note: This feature is applicable to only those applications that are linked to GIT.

Enable or disable the collision avoidance feature using the glide.ui.vcs.collision_avoidance property in the System Property [sys_properties] table. By default, the feature is enabled. See Available system properties for more information.

When the feature is enabled, users can’t work in the default update set of the application. If a user is assigned the default update set, a unique update set is created when the user logs in to the application for the first time after the collision avoidance feature is enabled. This new update set is specific to the user in current application.

Name of the update set specific to the logged in user is, User ID or user name based on the value specified for the glide.ui.vcs.updateidentifier property in the System Property [sys_properties] table. See Available system properties for more information. By default, the update set name is set to user name of the logged in user. However, users can rename the update set.

Delta loading

"Delta loading" is an optimized way to load an application from a Git repository. When you switch branches and apply remote changes, the Delta loading feature makes sure only changed updates load. Rather than having to perform a full uninstall and reinstall of the application, only the changes update. This speeds and eases the process for developers and does away with the need to use all branches during development. Data stored in tables is retained during these operations, lowering the need to load demo data back into the application after a reinstall or branch change operation.
Benefits of Delta loading

Time savings

Typically, the longest part of an application uninstall or reinstall is dropping and creating tables. Delta loading prevents that from occurring. Depending on the size of the application, not having to drop or create tables can significantly improve an operation's completion time.

Test and demo data retained

Tables that contain test and demo data are no longer deleted, so data is not lost. Developers can save time if they don't need to reimport demo data each time.

Cross-scope dependent references preserved

Items that are cross-scoped and share references are no longer lost when you apply changes. This used to occur when an item in scope B was associated to a parent item in scope A. Applying remote changes on scope A would delete the parent item and reinstall it, breaking the reference between both items, but not restoring that relationship. Delta loading prevents the uninstall, so the reference is not lost.

Recommended practice

Delta loading is enabled by default on all instances starting with the ServiceNow Paris release. The feature is designed to help in application development and receives continued support and upgrades.

Service Creator

Service creator enables a department to offer custom services through the service catalog, such as the HR department offering tuition reimbursement for further education.

Each published service has an associated record producer catalog item. Users designated as managers and editors create and design these catalog items. End users can request services by ordering the catalog item.

All services belong to a published service category, which has an associated application and modules. When a user orders the catalog item for a service, the ServiceNow system creates a new task record within the application for that service category. Users designated as service fulfillers for the department complete these tasks to fulfill the service request.
Service creator process

The service creator process involves requesting and publishing a service category, designating editors and service fulfillers, creating and publishing services, and submitting and fulfilling service requests.

Request and publish a service category

A user, typically the department manager, can request a service category for the department. This user provides high-level information regarding the service category, such as the name, the department, and the manager for the service category.

A catalog administrator can approve the request which publishes the service category, creates a ServiceNow application for managing service requests associated with the category, and creates system components for the application.

Designate editors and service fulfillers

After a service category is published, the associated manager designates editors and service fulfillers. Editors can create and modify services within that service category. Service fulfillers can complete tasks that are generated by service requests.

The manager, editors, and service fulfillers must be members of the department the service category belongs to.

Create and publish services

The manager and editors create services within a service category. The service design interface provides a work area for creating and modifying services.

When the service is complete, the manager publishes the service to the service catalog.

Submit and fulfill service requests

End users can request published services by submitting a service catalog request. This request creates a new task record within the service category application. Service fulfillers then complete the task to fulfill the service request.

Related information

Service Creator

Activate Service Creator

If the Service Creator plugin is not already activated, an administrator can activate it to access the application.
Procedure

1. Navigate to System Applications > All Available Applications > All.

2. Find the plugin using the filter criteria and search bar.
   
   You can search for the plugin by its name or ID. If you cannot find a plugin, you might have to request it from ServiceNow personnel. For more information, see Request a plugin.

3. Click Install, and then in the Activate Plugin dialog box, click Activate.

   Note: When domain separation and delegated admin are enabled in an instance, the administrative user must be in the global domain. Otherwise, the following error appears: Application installation is unavailable because another operation is running: Plugin Activation for <plugin name>.

Installed with Service Creator

Several types of components are installed with Service Creator.

Demo data is available with Service Creator. The demo data provides the Departmental Services service catalog category.

Creating a new service category also creates components for that service category.

The following components are added with Service Creator:

Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Category [catalog_category_request]</td>
<td>Stores all service categories.</td>
</tr>
<tr>
<td>Service Category Request User [catalog_category_request_user]</td>
<td>Tracks fulfillers for a service category. Use these records to grant or remove roles as needed.</td>
</tr>
<tr>
<td>Service [sc_cat_item_producer_service]</td>
<td>Stores all services.</td>
</tr>
<tr>
<td>Service Category App Menu [service_category_app_menu]</td>
<td>Stores the application menus for each service category.</td>
</tr>
</tbody>
</table>
### Service Creator tables (continued)

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Category User Role</td>
<td>Tracks users who have been granted a role due to being an editor of a service category.</td>
</tr>
<tr>
<td>[service_category_user_role]</td>
<td></td>
</tr>
</tbody>
</table>

### UI actions

<table>
<thead>
<tr>
<th>UI action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Category and Table</td>
<td>Approves a requested service category and creates system components for that category.</td>
</tr>
<tr>
<td>Request Category Publication</td>
<td>Lets a service creator request their category be published.</td>
</tr>
<tr>
<td>Create New Service</td>
<td>Creates a new service within the service category.</td>
</tr>
<tr>
<td>View Table Definition</td>
<td>Opens the task table definition [sys_db_object] for a service category.</td>
</tr>
<tr>
<td>View Task List</td>
<td>Opens the list of tasks associated with the service category.</td>
</tr>
</tbody>
</table>

### UI policies

<table>
<thead>
<tr>
<th>UI policy</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hide Due Date</td>
<td>Hides the Due date field on the Service Category form if State is Requested or Due date is empty.</td>
</tr>
<tr>
<td>Hide Category If Empty</td>
<td>Hides the Category field, if empty, on the Service Category form.</td>
</tr>
<tr>
<td>Show Published</td>
<td>Shows the Published check box on the Service Category form if State is Created but Unpublished or Ready for Publication.</td>
</tr>
<tr>
<td>Hide Table name</td>
<td>Shows Table and hides Table name on the Service Category form if Table has a value.</td>
</tr>
<tr>
<td>UI policy</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Hide Category Name</td>
<td>Hides Name on the Service Category form if State is Requested or Rejected.</td>
</tr>
<tr>
<td>Table name read only</td>
<td>Makes Department and Table name read only on the Service Category form if State is not Requested.</td>
</tr>
<tr>
<td>Hide Editors</td>
<td>Hides the Editors field on the Service Category form if State is Requested or Rejected.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>glide.citizen_developer.category.auto_publish</td>
<td>Automatically adds new service categories to the service catalog as subcategories of the Departmental Services category.</td>
</tr>
<tr>
<td></td>
<td>• Type: true</td>
</tr>
<tr>
<td></td>
<td>• Default value: true</td>
</tr>
<tr>
<td></td>
<td>• Location: System Properties [sys_properties] table</td>
</tr>
<tr>
<td>glide.citizen_developer.set_category_roles</td>
<td>Comma-separated list of roles that can set the category for a new service.</td>
</tr>
<tr>
<td></td>
<td>• Type: String</td>
</tr>
<tr>
<td></td>
<td>• Default value: admin, catalog_admin</td>
</tr>
<tr>
<td></td>
<td>• Location: System Properties [sys_properties] table</td>
</tr>
<tr>
<td>glide.service_creator.auto_add_to_category</td>
<td>Automatically adds new services to the Departmental Services service catalog category, in addition to the department-specific category.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Property Description

- **Type:** true | false
- **Default value:** true
- **Location:** System Properties [sys_properties] table

### Script includes

<table>
<thead>
<tr>
<th>Script include</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceCategoryIsUnpublished</td>
<td>Global function that returns true if the service category is unpublished.</td>
</tr>
<tr>
<td>getMyCatalogCategories</td>
<td>Global function that returns a list of categories for which the current user is the manager or an editor.</td>
</tr>
</tbody>
</table>

### Client scripts

<table>
<thead>
<tr>
<th>Client script</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duplicate Category Name Check</td>
<td>Displays a warning on the Service Category Request form when the requested service category has the same name as an existing service category.</td>
</tr>
<tr>
<td>Fix Table Name</td>
<td>Ensures a valid table name on the Service Category Request form.</td>
</tr>
<tr>
<td>Hide Draft Services</td>
<td>Hides the Draft Services related list on the Service Category Request form when appropriate.</td>
</tr>
<tr>
<td>Propose Table Name</td>
<td>Proposes a valid table name on the Service Category Request form for new service category requests.</td>
</tr>
<tr>
<td>Category Published</td>
<td>Displays a help message when Published is selected on the Service Category Request form.</td>
</tr>
<tr>
<td>Hide Fulfillers</td>
<td>Hides the Fulfillers related list on the Service Category Request form when appropriate.</td>
</tr>
<tr>
<td>Editors Message</td>
<td>Displays a help message for the Editors field when appropriate.</td>
</tr>
<tr>
<td>Client script</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Other Tables Message</td>
<td>Provides information about existing service category tables for the selected Department.</td>
</tr>
<tr>
<td>State Message</td>
<td>Displays a help message for the State field.</td>
</tr>
</tbody>
</table>

**Business rules**

<table>
<thead>
<tr>
<th>Business rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Query</td>
<td>Restricts users without the catalog_admin role to viewing service records within service categories they are the manager or editor of.</td>
</tr>
<tr>
<td>New Service</td>
<td>Provides a message when a new sc_cat_item_producer_service record is created.</td>
</tr>
<tr>
<td>Table Name Required</td>
<td>Ensures a service category request has a valid table name before approval.</td>
</tr>
<tr>
<td>Remove Fulfiller Role</td>
<td>Removes relevant role from service fulfillers when they are removed from a category.</td>
</tr>
<tr>
<td>Category Request query</td>
<td>Restricts users without the catalog_admin role to viewing service category records they are a manager or editor of.</td>
</tr>
<tr>
<td>Editor Role</td>
<td>Adds and removes relevant roles from service category editors.</td>
</tr>
<tr>
<td>Delete User Role</td>
<td>Removes the relevant role from service category editors when appropriate.</td>
</tr>
<tr>
<td>Category Published</td>
<td>Sets State to Published to Catalog when the Published check box is selected on the Service Category Request form.</td>
</tr>
<tr>
<td>Populate Service Name Empty</td>
<td>Populates a service name if none is provided.</td>
</tr>
<tr>
<td>Add Departmental Services Category</td>
<td>Adds a new service to the Departmental Services service catalog category.</td>
</tr>
<tr>
<td>Default Fulfillment User</td>
<td>Makes a category manager the assignee of service tasks if no assignee is specified.</td>
</tr>
<tr>
<td>Business rule</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Scratchpad Draft Services Count</td>
<td>Generates field help messages.</td>
</tr>
<tr>
<td>Catalog Category Request Approved</td>
<td>Creates components necessary to use of a new service category.</td>
</tr>
<tr>
<td>Manager Role</td>
<td>Grants relevant roles to category managers.</td>
</tr>
<tr>
<td>New Service Script</td>
<td>Populates the script of a new Service to set assignment group or user.</td>
</tr>
<tr>
<td>getDepartmentUsers</td>
<td>Returns the users of a department.</td>
</tr>
<tr>
<td>Draft Item Query</td>
<td>Restricts users without the catalog_admin role to viewing draft service records they are a manager or editor of.</td>
</tr>
<tr>
<td>Grant Fulfiller Role</td>
<td>Grants relevant role to service fulfillers.</td>
</tr>
<tr>
<td>Scratchpad Department Name</td>
<td>Generates field help messages.</td>
</tr>
<tr>
<td>Scratchpad</td>
<td>Generates field help messages.</td>
</tr>
<tr>
<td>Other Tables For Department</td>
<td>Generates field help messages.</td>
</tr>
<tr>
<td>Set Single Catalog from Single Category</td>
<td>Populates a default Catalog for a new service.</td>
</tr>
</tbody>
</table>

### Email notifications

#### Service Creator email notifications

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Category Published</td>
<td>Notifies the manager of a service category when the category request is approved.</td>
</tr>
<tr>
<td>Service Category Rejected</td>
<td>Notifies the manager of a service category when the category request is rejected.</td>
</tr>
<tr>
<td>Service Category Request Inserted</td>
<td>Notifies catalog administrators when a new category request is created.</td>
</tr>
<tr>
<td>Service Category Created</td>
<td>Notifies the manager of a service category when the category is created.</td>
</tr>
</tbody>
</table>
## Service Creator email notifications (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Category Publication Requested</td>
<td>Notifies catalog administrators when publication of a category has been requested.</td>
</tr>
<tr>
<td>Service Category Request Opened</td>
<td>Notifies the manager of a service category when a new category request is created on their behalf.</td>
</tr>
</tbody>
</table>

## Components created with new service categories

When you publish a new service category using the Service Creator application, the ServiceNow system creates components for the services in that category.

These components are distinct from the components installed with the Service Creator application. The following components are added for each new service category:

### Tables created with new service categories

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Department Name&gt; Tasks [&lt;service category table name&gt;]</code></td>
<td>The table that stores request task records for the service category. This table extends the Task table. The name of this table is defined by the department the service category is associated with, and the Table name field on the service category record. A new application menu and modules are created to allow users to access records on this table. Records on this table are numbered using a new Numbers [sys_numbers] record.</td>
</tr>
</tbody>
</table>

### User roles created with new service categories

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;service category table name&gt;_user</code></td>
<td>The user role required to access request records for a service category. The Table name for the service category determines the name of the role. Users designated as the manager, editors, or service fulfillers for a service category automatically receive this role. Only users with this role can be assigned task records for the service category. ACLs are created to allow users with this role to access the relevant service task table.</td>
</tr>
</tbody>
</table>
### Email notifications created with new service categories

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task commented for group</td>
<td>Notifies the group a service task record is assigned to when a user adds a comment.</td>
</tr>
<tr>
<td>Task commented for assignee</td>
<td>Notifies the user a service task record is assigned to when a user adds a comment.</td>
</tr>
<tr>
<td>Task closed for group</td>
<td>Notifies the group a service task record is assigned to when the record is closed.</td>
</tr>
<tr>
<td>Task worknoted for assignee</td>
<td>Notifies the user a service task record is assigned to when a user adds a work note.</td>
</tr>
<tr>
<td>Task assigned to group</td>
<td>Notifies the group a service task record is assigned to when the record is assigned to that group.</td>
</tr>
<tr>
<td>Task assigned to assignee</td>
<td>Notifies the user a service task record is assigned to when the record is assigned to that user.</td>
</tr>
<tr>
<td>Task worknoted for group</td>
<td>Notifies the group a service task record is assigned to when a user adds a work note.</td>
</tr>
<tr>
<td>Task closed for assignee</td>
<td>Notifies the user a service task record is assigned to when the record is closed.</td>
</tr>
<tr>
<td>Task opened for user</td>
<td>Notifies the user that opened a service task record when the record is created.</td>
</tr>
<tr>
<td>Task closed for user</td>
<td>Notifies the user that opened a service task record when the record is closed.</td>
</tr>
<tr>
<td>Task commented for user</td>
<td>Notifies the user that opened a service task record when a user adds a comment.</td>
</tr>
</tbody>
</table>

### Forms created with new service categories

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;department name&gt; Task</td>
<td>The form for viewing request task records for the service category. By default, this form uses a layout that includes a formatter to display the questions for the service and the answers provided by the requesting user.</td>
</tr>
</tbody>
</table>
### Service catalog categories

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;service category name&gt;</td>
<td>The default service catalog category for new services created within a service category.</td>
</tr>
</tbody>
</table>

### Service Creator roles

The Service Creator application uses the specific roles.

### Roles

<table>
<thead>
<tr>
<th>Role Title [Name]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Service Category&gt; User [&lt;service category table name&gt;_user]</td>
<td>Accesses request records for a service category. The table name for the service category determines the name of the role. Users designated as the manager, editors, or service fulfillers for a service category automatically receive this role.</td>
</tr>
<tr>
<td>Catalog Administrator [catalog_admin]</td>
<td>Creates, edits, and publishes service categories and services, and creates and edits notifications including template notifications. Catalog administrators are primarily responsible for approving service category requests.</td>
</tr>
<tr>
<td>Catalog Manager [catalog_manager]</td>
<td>Creates, edits, and publishes services, and designates editors and service fulfillers. A user designated as the manager of a service category receives this role automatically.</td>
</tr>
<tr>
<td>Catalog Editor [catalog_editor]</td>
<td>Creates and edits services. A user designated as an editor of a service category receives this role automatically.</td>
</tr>
</tbody>
</table>

### Manage a service

Using the Service Creator, department managers can request a new service category, designate editors and service fulfillers for that category, and create and publish services.

**About this task**

Editors create and modify services. Service fulfillers complete the tasks generated from service requests.

A service category request involves assigning a service category manager, which is typically the department manager who makes the request. After the request is submitted, a catalog administrator approves the request to publish...
the service category. When the category is published, the service category manager can assign service category editors and service fulfillers, and create services to offer in the service catalog.

To request a new service category:

**Procedure**
1. Navigate to **Self-Service > Service Catalog**.
2. Select the **Departmental Services** category.
   
   The *Departmental Services* category is part of the demo data available with service creator. If this category does not exist, a catalog administrator must add the *Service Category Request* catalog item to an existing category.
3. Select the **Service Category Request** item.
4. Change the default values, as necessary (see table).
5. Click **Submit**.

---

**Managing Services**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department</td>
<td>Department this category request is for. By default this value is the department of the current user. Changing this value also changes the Category name and Category manager values.</td>
</tr>
<tr>
<td>Category name</td>
<td>Name for the new service category. By default, ServiceNow uses a name based on the Department value.</td>
</tr>
</tbody>
</table>
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category manager</td>
<td>Designated manager for the new service category. By default, ServiceNow uses the manager for the selected department.</td>
</tr>
<tr>
<td>Needed by</td>
<td>Date that the new service category should be available.</td>
</tr>
<tr>
<td>Comments</td>
<td>Additional comments describing the service category. This information appears as a journal entry on the Service Category form.</td>
</tr>
</tbody>
</table>

### Designing services

Service creator includes an interface for designing services.

Using this interface, service category managers and editors can create and publish services, and edit service details.

All services must belong to a service category. If your department or group does not have an existing service category, you must create a new service category before you can design services for that category.

### Add a template notification

Adding a template notification.

**Procedure**

1. Navigate to **Service Creator > Template Notifications**.
2. Click **New**.
3. In the **Send when** field, select **Event is fired**.
4. In the **Event name** field, select **ccrTemplate**.
5. Enter other notification details.
6. Click **Submit**.

The new template notification creates a notification for all service categories published after that point.

### Notification configurations

All service categories start with a set of associated notifications, such as the notification when a task to fulfill a service request is assigned.

Notifications defined in the **Service Creator > Template Notifications** module are copied when a user creates a new service category.

Template notifications are distinct from the notifications for the Service Creator application itself, such as the notification when a new service category is
approved or rejected. Notifications for the Service Creator application are defined in Service Creator > Notifications.

A system administrator can add and delete template notifications.

Create the category and table

After the request has been submitted, a catalog administrator can approve or reject the request.

About this task

Approving the request creates a new table for the service category, adds an application to the application navigator using the Category name as the application label, and sets the State of the service category to Published to Catalog.

Procedure

1. Navigate to Service Creator > Category Requests.
2. Open a record with a State of Requested.
3. Review the requested service category. ServiceNow provides a suggested Table name based on the Department.
   - If a service category exists with the specified category name or table name, a message appears under that field. Use a unique value for these fields.
4. Click Create Category and Table to approve the request or Reject to reject the request.
   If notifications are enabled for the instance, the service category Manager is notified of the approval or rejection.
After publishing the service category, you can access the new table by navigating to the new application in the application navigator, or by clicking the View Task List related link on the Service Category form.

Related information
  Manage a service

Delete a template notification
Deleting a template notification prevents new service categories from using the notification, but does not delete notifications for service categories that have already been created.

Procedure
1. Navigate to Service Creator > Template Notifications.
2. Select a notification record.
3. Click Delete.
4. Click OK to confirm.

Designate an editor
Editors can create and modify services within a service category.

About this task
Editors automatically receive the catalog_editor role.
The service category manager can designate editors for a published service category.

**Procedure**

1. Navigate to **Service Creator > My Service Categories**.
2. Select a record with a **State** of **Published to Catalog**.
3. Click the lock icon beside the **Editors** field.
4. Select users to designate as editors using the reference lookup icon. Only users in the appropriate department are available for selection.
5. After adding all editors, click **Update**. Editors receive the Catalog Editor role.

---

**Service Category**

<table>
<thead>
<tr>
<th>Number</th>
<th>CCR0001001</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department</td>
<td>HR</td>
</tr>
<tr>
<td>Manager</td>
<td>Mariano Maury</td>
</tr>
</tbody>
</table>

**Editors**

- Steve Schorr
- Rubin Crotts

---

**Designate a service filler**

Service fillers can complete service requests submitted for a service category.

**About this task**

Service fillers can access applications for service categories they are assigned to, but cannot access the Service Creator application.

The **Service Fulfillers** related list on the Service Category form displays all users assigned as fillers for that service category. The service category manager can designate service fillers for a service category.
Procedure
1. Navigate to Service Creator > My Service Categories.
2. Select a service category with a State of Published to Catalog.
3. In the Service Fulfillers related list, click Edit.
4. Use the slushbucket to add the appropriate service fulfillers.
   Only users in the appropriate department are available for selection.
5. Click Save.

Related Links
Create New Service
View Task List

Fulfill a service request
End users can request published services through the service catalog.

About this task
When a user requests a service, the ServiceNow system creates a new task for that service category. Service fulfillers complete these tasks to fulfill the request.

New request tasks are automatically assigned to the group or user specified in the Fulfillment Group or Fulfillment User Service setting. If no fulfillment group or user is set, new records are assigned to the service category manager.

Questions for a particular service and the answers entered by the requesting user appear in the Variables section on the fulfillment task record.

Procedure
1. Navigate to <Your service application> > Assigned to me.
2. Select a record.
3. Review the information presented.
4. Complete the task in accordance with department policies and procedures.
5. Set the state of the service request record to **Closed Complete**.
6. Click **Update**.

**Publish a service**

A service must be published to appear in the service catalog. When first created, new services appear in the Draft Services related list for the service category. Published services appear in the Services related list for the service category. The manager of a service category can publish draft services.

**Procedure**

1. Navigate to **Service Creator > My Service Categories**.
2. Select a service category with a State of **Published to Catalog**.
3. On the Service Category form, right-click a service in the **Draft Services** related list.
4. Select **Publish**.

**Team Development**

Team Development supports parallel development on multiple, non-production ServiceNow instances.

**Important:** Team Development note here.

Team Development provides the following features:

- Branching operations, including pushing and pulling record versions between instances.
- The ability to compare a development instance to other development instances.
- A central dashboard for all Team Development activities.

**Team Development overview**

Team Development allows developers to work on separate development instances while sharing code and resolving collisions throughout the development process.
After setting up the instance hierarchy, you can develop changes on your local development instance. Use the team dashboard to manage Team Development activities, such as:

- Tracking local changes and determining which changes to promote to the parent development instance.
- Pulling changes from the parent instance and resolving any collisions with local changes.
- Comparing your instance with other development instances and resolving any collisions with other development projects.
- Pushing changes when a feature is tested and ready to promote to the parent development instance.

Developers with admin access to their development instance and the parent instance can use team development. For alternative access settings, see Granting access rights to developers.

Related information

Set up an instance hierarchy

When to use Team Development

Team Development allows multiple developers to work on applications.

<table>
<thead>
<tr>
<th>Deployment option</th>
<th>Good for</th>
<th>Future considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update Sets</td>
<td>Storing changes to a baseline or installed application. Storing and applying a particular version of an application. Producing a file for export.</td>
<td>You can manually create update sets to store a particular application version. Use update sets to deploy patches or changes to installed applications. <strong>Note:</strong> Do not use update sets to install applications. Instead, use the application repository or the ServiceNow Store to install applications.</td>
</tr>
<tr>
<td>Application Repository</td>
<td>Installing and updating applications on all company instances.</td>
<td>Consider uploading an application to the ServiceNow Store to share it with other users. Allows installation of and update to the latest application version only.</td>
</tr>
<tr>
<td>Deployment option</td>
<td>Good for</td>
<td>Future considerations</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------</td>
<td>-----------------------</td>
</tr>
</tbody>
</table>
|                   | Automatically managing application update sets.  
Restricting access to applications to the same company.  
Deploying completed applications to end users. | Use update sets to store prior application versions.  
**Note:** If used with team development, publish applications only from a parent instance. |

**Team Development**

|                   | Providing change management across multiple instances.  
Allowing multiple developers to work on applications.  
Organizations that have access to several non-production instances. | Consider providing each development team access to a dedicated development instance.  
Requires developers to manually merge colliding changes.  
Works only for instances owned by the same organization.  
**Note:** If used with the application repository, publish applications from a parent instance. |

---

**Local changes**

The Local Changes table tracks which customized records have current versions that exist on the development instance but not on the parent instance.

Use local changes to collect changes in preparation for a push.

You queue local changes that are ready to push. Each development instance maintains a single queue, regardless of who develops or queues the changes. You ignore local changes that you do not want to push. For example, you may want to ignore changes to the color scheme that visually distinguish a development instance from the production instance. You can remove a change from the queue or stop ignoring a change.
Changing the parent instance or reconciling recreates the list of local changes that have not been queued or ignored. If you had previously queued or ignored a local change, that designation is maintained.

**Local change lists**

On the team dashboard, the *Local Changes* list shows the local changes that have not been queued for the next push or ignored for all pushes. The *Ready to Push* list shows the changes that are queued, and the *Ignored* list shows the changes that are ignored. Use any of these methods to navigate a list of local changes.

<table>
<thead>
<tr>
<th>local changes list</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Action</strong></td>
</tr>
<tr>
<td>Click the reference icon beside the row</td>
</tr>
<tr>
<td>Click the link in the first column</td>
</tr>
<tr>
<td>Right-click the row and select <strong>Show Changes Since Last Pull</strong></td>
</tr>
<tr>
<td>Right-click the row and select <strong>Show Application File</strong></td>
</tr>
<tr>
<td>Right-click the row and select <strong>Show Version</strong></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

**Pull exceptions**

Pulling ignores versions when certain conditions occur.
Pull exceptions table

<table>
<thead>
<tr>
<th>Issue</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matched an exclusion policy</td>
<td>An exclusion policy prevents pulling changes for records matching the policy conditions. The pull identifies the changes but does not include versions for these records.</td>
</tr>
<tr>
<td>Private properties</td>
<td>A private property is excluded from all Update Sets and pulls.</td>
</tr>
<tr>
<td>Collisions</td>
<td>A collision is detected when the pulled version and the current local version both include modifications to the same record. You must resolve all collisions before you can pull.</td>
</tr>
<tr>
<td>Previously resolved collisions</td>
<td>When you resolved a collision by accepting either the pulled version or local version of a record, the pull remembers your decision and accepts the version that you indicated as a &quot;previously resolved collision&quot;.</td>
</tr>
<tr>
<td>Skipped</td>
<td>Pulls skip versions where there is a problem with the version record such as a corrupt or missing version.</td>
</tr>
</tbody>
</table>

Team dashboard

The team dashboard provides a central place to manage all Team Development activities on your development instance.

You can track local changes, pull and push changes between the local and parent instances, compare the local instance to other development instances, and resolve any collisions. You can also reconcile with the current parent instance or change the parent instance.

To access the dashboard, navigate to Team Development > Team Dashboard.
The control panel in the top left provides status indicators and Team Development actions.

- **Parent**: indicates the status of the connection to the parent instance. If a problem or warning is detected, point to the indicator to view the error messages, or click the indicator to open the remote instance record.

- **Change**: changes the parent instance. See Changing the Parent Instance.

- **Reconcile**: compares the development instance to the parent instance. See Reconciling.

- **Ready to Pull**: indicates the number of changes on the parent that have not been pulled to the local instance.

- **Pull**: initiates a pull. See Pulling Versions.

- **Push**: opens a page that allows you to review the changes before a push. See Pushing Versions.

- **Refresh**: updates the status indicators on the control panel. The dashboard updates only when you reload or refresh the page.

- **Local**: indicates the status of the most recent comparison with another instance. If collisions are detected, click the indicator to open the list and resolve the collisions. See Resolve a collision in Team Development.

- **Collisions**: appears only if any local changes collide with versions pulled from the parent and indicates the number of collisions. Click the indicator to open the list and resolve the collisions. See Resolve a collision in Team Development.

- **Compare to**: allows you to select another development instance to compare with the local instance. See Comparing to Peer Instances.
• Ready to Push: indicates the number of local changes that are queued for the next push. See Queuing and Ignoring Local Changes.

• Local changes: indicates the number of local changes that have not been queued or ignored. Click the indicator to open a list of these changes.

• Ignored: appears only if any local changes are ignored and indicates the number of ignored changes. Click the indicator to open a list of these changes.

The team dashboard includes lists for tracking local changes and viewing the history of Team Development activities.

• Local changes: lists the local changes that have not been queued or ignored.

• Pushes and Pulls: provides a history of pushes and pulls. Expand a row to see the customized records for which versions were transferred as part of the push or pull.

• Instance Comparisons: provides a history of comparisons with other development instances.

• Collisions: lists the collisions that must be resolved before the next pull or push. You can right-click a row and select Resolve Collision. See Resolving Collisions.

• Ready to Push: lists the local changes that have been queued for the next push.

• Ignored: lists the local changes that are ignored for all pushes.

Related reference
Pull exceptions

Related information
Team Development overview
Pull a version
Resolve a collision in Team Development
Resolve multiple collisions
Local changes
Local change lists
Queue a local change for a push
Ignore a local change
Back out a local change
Push a version
Approve or reject a push
Check the review status of a pushed change
Cancel a code review request
Compare to peer instances
Change the parent instance
Reconcile changes

Approve or reject a push

Code reviewers must approve or reject a push from the Team Development application.

About this task
Although reviewers can see the individual versions within a push, they must approve or reject the push as a whole.

Procedure
1. Log in to the parent instance that requires code review.
2. Navigate to **Team Development > Code Review Requests**.
3. Select a change in the **Awaiting Code Review** stage.
4. Review the changes in the **Push or Pull Versions** related list.
5. Click **Approve or Reject**.
6. Optional: Enter review comments in **Comments**. These comments are visible to anyone who can see the Pushes and Pulls history.
7. Click either **Approve** or **Reject**, as appropriate.

Note: The **URL** and **Remote Instance** fields list the address and name of the instance where the change originated.
Back out a local change

Back out all local changes and restore the last version reconciled with the parent instance.

Procedure

1. Define a parent instance.
2. Pull changes from the parent instance.
3. Navigate to Team Development > Team Dashboard.
4. Filter the Local Changes list to show only the changes that you want to back out.
5. Do one of the following actions:
   - Click Back Out All.
   - Right-click the local change you want to back out, and then click Back Out.

Cancel a code review request

Developers can cancel any push they submitted that is in the Awaiting Code Review stage.

About this task

Canceling a request sets the push to the Code Review Request Cancelled stage on the submitting instance. The submitting instance retains a version history of the push but the parent instance does not.

Procedure

1. Log in to the instance that pushed the changes.
2. Navigate to Team Development > Pushes and Pulls.
3. Filter for the push you want to cancel.

   ☑️ **Note:** You cannot cancel a push that has been approved or rejected.

4. Select the Push or Pull record.

5. Click **Cancel Code Review**.

**Change the parent instance**

If it becomes necessary to modify the instance hierarchy, you can change the parent for a development instance.

**About this task**

Changing the parent initiates a complete comparison between the development instance and the new parent instance. To optimize comparison speed and reduce the number of collisions and local changes that need review afterwards, ensure that the new parent instance was cloned recently from an appropriate instance (for example, the production instance). Before you change the parent instance, ensure that the change does not conflict with your change management process or other development efforts.

To change the parent for a development instance:

**Procedure**

1. On the development instance, navigate to **Team Development > Team Dashboard**.

2. In the control panel, click **Change**.

3. Select the remote instance you want to use as the parent and click **Select**.

   Alternatively, click the link to define a new remote instance. Then, repeat steps 1–3 and select the remote instance you defined.

![Change current parent instance](image)

The system initiates a reconcile, which compares the local instance to the parent, and then generates the list of local changes and calculates the number of changes that are ready to pull from the parent.

4. On the completion page, click **Team Dashboard**.
5. Pull versions from the parent instance and resolve any collisions.
6. Review the local changes list and queue or ignore changes, as appropriate.

Check the review status of a pushed change

If the parent instance requires pushed changes to undergo code review, changes are placed in the Awaiting Code Review stage.

About this task

If you configure the parent instance to send notifications, it sends the submitting developer a notification when the pushed changes are approved or rejected. Developers can also manually check the status of their pushed changes from the Pushes and Pulls module on the submitting instance.

Procedure

1. Log in to the instance that submitted code for review.
2. Navigate to Team Development > Pushes and Pulls.
3. Filter for the push you want to review.
   • Pushes in the Complete stage are approved and applied to the parent instance.
   • Pushes in the Collided stage are rejected because of a collision.
   • Pushes in the Awaiting Code Review stage are awaiting review.
   • Pushes in the Code Changes Rejected stage are rejected by a reviewer.
   • Pushes in the Code Review Request Canceled stage are canceled by the submitting developer.
4. Click the Reviews related list to see the following information.
• Who submitted a review decision.
• What the decision was: either approved or rejected
• What comments if any the reviewer provided.

Compare a pushed version to a local version

Code reviewers can compare the pushed versions to the local versions to see the potential effect of incoming changes.

Procedure
1. Log in to the instance requiring code review.
2. Navigate to Team Development > Code Review Requests.
3. Select a change in the Awaiting Code Review stage.
4. Review the changes in the Push or Pull Versions related list.
5. Right-click a row in the list and click Compare to Current. A comparison of the differences between the pushed and local versions appears.

Related information

Merge tool
Compare to the current version
Compare two versions of an article
Resolve conflicts for an individual record
Resolve a collision in Team Development
Compare to peer instances
You can compare the local instance to any other remote instance and commit any current versions from the remote instance on your development instance.

About this task
Comparing allows you to share code between instances without pushing to a common parent.

Comparing instances does not automatically commit any versions on the local instance. It initiates a full comparison of all changes on the remote instance and all changes on the local instance, and then reports which customized records have different current versions. You can selectively commit a version from the remote instance or compare it with the version on your local instance. You can delete the instance comparison record when you finish evaluating the differences.

To compare the local instance to a peer instance:

Procedure
1. Ensure that the peer instance is defined as a remote instance.
2. Navigate to Team Development > Team Dashboard.
3. In the control panel, click Compare to.
4. Select the peer instance you want to compare to the local instance and click Compare.
5. On the completion page, click Show Results. The instance comparison record opens.
6. Review the On Remote and not Local related list, which shows the customized records where the current version on the peer instance is not on the local instance. For each customized record, you can:

- Compare the current remote version to the current local version by right-clicking a row and selecting **Compare to Current**.
- Load the current remote version as the current local version by right-clicking a row and selecting **Load This Change**.

**Ignore a local change**

Ignoring a local change prevents updates to a record from generating new versions in the Local Changes list.

**About this task**

An ignored local change always points to the current version for the record. You cannot push ignored records to another instance.

**Local change action list**

<table>
<thead>
<tr>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ignore a record that has a version queued for push</td>
<td>The queued change is deleted</td>
</tr>
<tr>
<td>Ignore a record that has a version queued for code review</td>
<td>The queued change is deleted</td>
</tr>
<tr>
<td>Pull changes for an ignored record</td>
<td>Collision</td>
</tr>
</tbody>
</table>
Local change action list (continued)

<table>
<thead>
<tr>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resolve a collision by taking the parent</td>
<td>There is no longer a local change to ignore</td>
</tr>
<tr>
<td>version</td>
<td></td>
</tr>
<tr>
<td>Resolve a collision by keeping the local</td>
<td>The ignored change remains on the local instance</td>
</tr>
<tr>
<td>version</td>
<td></td>
</tr>
</tbody>
</table>

Procedure

1. Navigate to **Team Development > Team Dashboard**.
2. Filter the **Local Changes** list to show only the changes that you want to ignore.
   For example, filter the list to show all changes in the **Default** Update Set.
3. Click **Ignore All**.
4. [Recommended] Review the **Ignored** list to ensure that the correct changes are ignored.
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>To stop ignoring changes</td>
<td>Select the check boxes beside the rows and select <strong>Do Not Ignore</strong> from the Actions choice list.</td>
</tr>
<tr>
<td>To stop ignoring changes and add them to the queue instead</td>
<td>Select the check boxes beside the rows and select <strong>Queue for Push</strong> from the Actions choice list.</td>
</tr>
</tbody>
</table>

### Pull a version

Pulling retrieves versions of customized records from the parent instance and adds them on the development instance. Pulling does not retrieve any versions for changes made by system upgrades, but it retrieves all versions for changes made by users, not just the current version.

### About this task

Pulling retrieves all versions for changes made by users that have not already been pulled onto the development instance, and you cannot choose which versions to pull. The first time you pull from a parent instance, the pull retrieves all versions for changes made by users. Subsequent pulls retrieve the new versions since your last pull. Each pull is recorded in the Push or Pull [sys_sync_history] table on the development instance. Historical versions are saved with a state of **History**.

### Procedure

1. Navigate to **Team Development > Team Dashboard**.
2. In the control panel, click **Pull**.
3. On the completion page, click **Show Results**. The Push or Pull form opens.

   The **Push and Pull Versions** related list shows the customized records for which versions were retrieved and indicates if any pull exceptions exist.
4. Resolve any collisions.

**Push a version**

Pushing promotes changes from the development instance to the parent instance and commits the current version of a customized record on the development instance as the current version on the parent instance.

**About this task**

Pushing adds only the current development version to the parent, not all the development versions.

⚠️ **Note:** Updates to records from different applications cannot be pushed/pulled in the same push/pull. To resolve the error in the case that updates to other applications are mixed in: De-queue the updates to other applications. Push for one application. Re-queue the updates to one application. Push and then repeat as needed.

Pushing creates a local Update Set on the parent that is marked as complete. Pushed changes are also tracked as local changes on the parent. Therefore, you can promote changes through your development and test hierarchy by transferring the Update Set or by pushing the local changes. Each push is recorded in the Push or Pull table on the development instance.
Procedure

1. Navigate to **Team Development > Team Dashboard**.

2. **Queue the local changes** that are ready to push.

3. **Pull versions** from the parent instance and **resolve any collisions**.

   You cannot push changes to the parent instance if collisions are detected.

4. In the control panel, click **Push**. The Push Changes page opens.

5. **Provide a Name** for the changes.

6. **Review the list of changes** to ensure that the correct changes are included.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>To remove changes that you do not want to push</strong></td>
<td>Select the check boxes beside the rows and select <strong>Do Not Push</strong> from the <strong>Actions</strong> choice list</td>
</tr>
<tr>
<td><strong>To add changes</strong></td>
<td>Click <strong>Cancel</strong> and repeat the procedure from step 2</td>
</tr>
</tbody>
</table>

7. **Optional**: Edit the name. The name identifies the push record on the development instance and the local Update Set record on the parent instance.
8. **Optional:** Enter comments. The comments are added to the push record on the development instance and the local Update Set record on the parent instance.

9. Click **Push Changes.** The system initiates a pull to ensure that there are no collisions before the push proceeds.
   - If collisions are detected, the push is automatically canceled and you must repeat the procedure from **step 3.**
   - If no collisions are detected, the changes are staged on the parent instance. On the parent, each version is validated and then committed in the correct order to maintain dependencies between records. For example, a new table is committed before a field on that table to ensure the field is properly created.

**Note:** You cannot push if there is a version conflict between instances or the pushing instance has changes in the Awaiting Code Review stage.

10. On the completion page, click **Show Results.**

11. Review the push record for any errors or skipped changes.
   - Changes with a state of **Pushed** were committed on the parent instance.
   - Changes with a state of **Skipped** were not committed on the parent instance and remain queued as local changes on the development instance.

12. For each skipped change, review the log message to determine why the change was skipped. Develop any changes that are necessary to commit the desired version on the parent instance, and then push them. Some examples of why a change may be skipped include:
   - A table does not exist on the parent because it was created when you activated a plugin on the development instance. Ensure the plugin is activated on the parent and push the change again.
   - An error occurred during the push. Try to push again.
• The current version is invalid. Revert to a previous version and make the change again to ensure the version is valid.

• An error occurred on the parent during the push. The Log field on the push record contains the exception message. Review the system logs on the parent instance and troubleshoot any problems with the instance.

Back out a push
Application developers can back out a push to remove unwanted changes.

Procedure
1. Navigate to **Team Development > Pushes and Pulls**.
2. Select the push to back out.
3. Click **Back Out**.
4. Click **OK** when the confirmation message appears.

Queue a local change for a push
Application developers can queue a local change for a push to ensure the changes are available to other developers.
Procedure
1. Navigate to Team Development > Team Dashboard.
2. Filter the Local Changes list to show only the changes that are ready to push.
   For example, filter the list to show only the changes associated with a particular application.

3. Click Queue All For Push.
4. [Recommended] Review the Ready to Push list to ensure that the correct changes are in the queue.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>To remove changes from the queue</td>
<td>Select the check boxes beside the rows and select <strong>Do Not Push</strong> from the Actions choice list.</td>
</tr>
<tr>
<td>To remove changes from the queue and choose to ignore them instead</td>
<td>Select the check boxes beside the rows and select <strong>Ignore This Change</strong> from the Actions choice list.</td>
</tr>
</tbody>
</table>
Note: For the Local Changes list, click Reset Filter to remove any filter conditions you added and see all the local changes that have not been queued or ignored.

Reconcile changes
Reconciling first compares the local instance to the parent, and then generates the list of local changes and calculates the number of changes that are ready to pull from the parent.

About this task
A reconcile occurs automatically whenever you select a parent instance. You may need to manually reconcile after an external disruptive event on the parent instance, such as a clone or failover.

Note: This process may take a while to complete depending on the size and age of the instance.

Procedure
1. Navigate to Team Development > Team Dashboard.
2. In the control panel, click Reconcile.
3. In the confirmation dialog box, click OK.
   The list of local changes that have not been queued or ignored is recreated. If you had previously queued or ignored a local change, that designation is maintained.
4. Optional: On the completion page, click Show Results. Review the instance comparison record.
- The **On Remote and not Local** related list shows the versions that are ready to pull from the parent.
- The **On Local and not on Remote** related list shows the local versions that are ready to queue or ignore.

5. Click **Team Dashboard**.

6. Pull versions from the parent instance and then resolve any collisions.

7. Review the local changes list and queue or ignore changes, as appropriate.

**Resolve a collision in Team Development**

A collision is detected when the pulled version and the current local version are modifications of a different version, indicating that someone else has modified the same record that you have modified. The team dashboard displays the number of collisions between the local and the parent instance.

**About this task**

To ensure that your changes do not conflict with other development efforts, you should resolve collisions as soon as they are identified. You must resolve all collisions before you can pull or push.

**Procedure**

1. Navigate to **Team Development > Team Dashboard**.

2. In the control panel, click **Collisions** or click the count of collisions. A list of collisions opens.
3. Right-click a row and select **Resolve Collision**. (Alternatively, open the record and click the **Resolve Collision** related link.) The Resolve Collision page displays a comparison between the version that was pulled from the parent and your local record. The page highlights the differences.

4. Review the differences. You have the following options:

<table>
<thead>
<tr>
<th>Option</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To maintain the local record as the current version</td>
<td>Click <strong>Use Local Version</strong>. The pulled version is added to the version history for the record.</td>
</tr>
<tr>
<td>To load the version pulled from the parent as the current version</td>
<td>Click <strong>Use Pulled Version</strong>.</td>
</tr>
<tr>
<td>To move a setting from the selected version to update the current version</td>
<td>To move a change, click the &gt; (Move Right) button for the field. To work with scripts and text fields, click in the field and modify the text as needed. When the records meet your needs, click <strong>Save Merge and Resolve Collision</strong>.</td>
</tr>
</tbody>
</table>

| Note: Some types of record do not support this method. See **Limitations on updating records** for more information. |

The system performs that action and also clears the collision for future push/pulls.

5. Repeat the process for every remaining collision.

**Results**
The system saves the merged changes and resolves the collision.

**Related information**
- Compare to the current version
- Compare a pushed version to a local version
- Compare two versions of an article
- Resolve conflicts for an individual record
- Revert a change
- View customizations and compare with current version
Limitations on updating records

There are some types of records that you cannot merge while resolving differences on the Compare to Current and Resolve Collision pages.

Record types that allow a choice only between reverting or accepting the pulled or current record

The following record types do not allow you to merge individual values. Instead, differences involving the following record types display a read-only comparison and allow a choice between updating and reverting:

- sys_choice [Choice]
- sys_choice_set [Choice Set]
- sys_ui_form [Form]
- sys_ui_list [List]
- sys_ui_related_list [Related List]
- sys_ui_section [Form Section]
- wf_workflow [Workflow]
- wf_workflow_version [Workflow Version]

Team Development, Resolve Collision page: Use Pulled Version and Use Local Version options.

Upgrade History, Compare to Current page: Comparing non-current update versions to current update version. Allows only Revert to Base System option.

Field types that do not support merging

The following field types do not support individual merging between versions or updates:

- auto_increment [Auto Increment]
- auto_number [Auto Number]
- breakdown_element [Breakdown Element]
- catalog_preview [Catalog Preview]
- collection [Collection]
- color_display [Color Display]
- composite_field [Composite Field]
- compressed [Compressed]
- counter [Counter]
• currency [Counter]
• data_array [Data Array]
• data_object [Data Object]
• data_structure [Data Structure]
• date [Other Date]
• datetime [Basic Date/Time]
• days_of_week [Days of Week]
• document_id [Document ID]
• due_date [Due Date]
• Email [Email]
• external_names [External Names]
• field_list [Field List]
• float [Floating Point Number]
• glide_action_list [UI Action List]
• glide_precise_time [Precise Time]
• glide_var [Glide Var]
• image [Basic Image]
• index_name [Index Name]
• int [Integer String]
• integer_time [Integer Time]
• ip_address [IP Address]
• journal [Journal]
• journal_input [Journal Input]
• journal_list [Journal List]
• long [Long Integer String]
• mask_code [Mask Code]
• metric_absolute [Metric Absolute]
• metric_counter [Metric Counter]
• metric_derive [Metric Derive]
• metric_gauge [Metric Gauge]
• mid_config [MID Server Configuration]
• month_of_year [Month of Year]
• multi_small [Multiple Line Small Text Area]
• name_values [Name/Values]
• nl_task_int1 [NL Task Integer 1]
• order_index [Order Index]
• password [Password (1 Way Encrypted)]
• percent_complete [Percent Complete]
• ph_number [Phone Number]
• phone_number [Phone Number (Unused)]
• phone_number_e164 [Phone Number (E164)]
• price [Price]
• reference_name [Reference Name]
• related_tags [Related Tags]
• reminder_field_name [Reminder Field Name]
• repeat_count [Repeat Count]
• repeat_type [Repeat Type]
• replication_payload [Replication Payload]
• schedule_date_time [Schedule Date/Time]
• short_field_name [Short Field Name]
• short_table_name [Short Table Name]
• Slushbucket [slushbucket]
• source_id [Source ID]
• source_name [Source Name]
• source_table [Source Table]
• string_boolean [<none>]
• sys_class_name [System Class Name]
• sysrule_field_name [System Rule Field Name]
• table []
• text []
• time []
• timer [Timer]
Resolve a collision in Team Development

Related information

Resolve multiple collisions

You can resolve multiple collisions without reviewing the differences between the local and pulled versions.

Procedure

1. Navigate to **Team Development > Team Dashboard**.
2. In the control panel, click the number of collisions. A list of collisions opens.
3. Select the check boxes beside the rows you want to resolve.
4. In the **Actions** choice list, use one of the following methods to resolve the collision:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>To load the version pulled from the parent as the current version for all selected collisions</strong></td>
<td>Select <strong>Use Pulled Version</strong></td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>To maintain the local version (local record) as the current version</td>
<td>Select <strong>Use Local Version</strong></td>
</tr>
<tr>
<td>for all selected collisions.</td>
<td></td>
</tr>
<tr>
<td>The pulled versions are added to the version history for the records.</td>
<td></td>
</tr>
</tbody>
</table>

**Team Development setup**

To enable parallel development on multiple non-production instances, administrators can set up the Team Development instance hierarchy and grant access rights for developers.

**Access rights for developers**

To use Team Development, application developers must have a set of credentials for each instance in the Team Development hierarchy.

An instance’s placement in the **team development hierarchy** determines the credentials it requires.

**Credentials for Team Development access**

<table>
<thead>
<tr>
<th>Desired Access</th>
<th>Credential Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access to the Team Development application</td>
<td>A user with the admin role on the instance you are accessing</td>
</tr>
<tr>
<td>Right to register a remote instance</td>
<td>One of following:&lt;br&gt;• A user with the admin role on the instance you are registering&lt;br&gt;• A user with the teamdev_user role on the instance you are registering</td>
</tr>
<tr>
<td>Right to push changes to the parent instance from a</td>
<td>One of following:&lt;br&gt;• A user with the admin role on the parent instance&lt;br&gt;• A user with the teamdev_user role on the parent instance</td>
</tr>
<tr>
<td>development instance</td>
<td>Right to compare to a registered remote instance &lt;br&gt;One of following:&lt;br&gt;• A user with the admin role on the instance you are accessing</td>
</tr>
</tbody>
</table>
Credentials for Team Development access (continued)

<table>
<thead>
<tr>
<th>Desired Access</th>
<th>Credential Requirements</th>
</tr>
</thead>
</table>
| • A user with the admin role on the registered development instance
• A user with the teamdev_user role on the registered development instance |

Access to the Code Review Requests module

<table>
<thead>
<tr>
<th>Access to the Code Review Requests module</th>
<th>One of following:</th>
</tr>
</thead>
</table>
| • A user with the admin role on the parent instance
• A user with the teamdev_code_reviewer role on the parent instance |

⚠️ Note: The teamdev_user role does not grant access to the Team Development application and is not intended for developers to work on local development instances. It is intended to grant developers non-admin access to remote instances such as the parent instance or a peer development instance.

Create an exclusion policy

Application developers can create an exclusion policy to prevent pushes or pulls to particular instances in the team development hierarchy.

Procedure

1. Navigate to Team Development > Exclusion Policy.
2. Click New.
3. Complete the Exclusion Policy form (see table).
4. Click Submit.

Exclusion policy form

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique description of the policy.</td>
</tr>
<tr>
<td>Policy</td>
<td>Select when the policy applies. Options include:</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>• Push only</td>
<td></td>
</tr>
<tr>
<td>• Push and Pull</td>
<td></td>
</tr>
<tr>
<td>• Pull only</td>
<td></td>
</tr>
<tr>
<td>Remote Instance</td>
<td>[Optional] Select a specific remote instance to ignore changes from during pull operations. Leaving this field blank ignores changes from all remote instances.</td>
</tr>
<tr>
<td>Table</td>
<td>Select which table to ignore changes for.</td>
</tr>
<tr>
<td>Conditions</td>
<td>Select any additional criteria a change must meet to be ignored other than the table name. This field is only visible when the Policy is Push only.</td>
</tr>
</tbody>
</table>

**Define a remote instance**
For each instance, define other instances in the hierarchy as remote instances.

**About this task**
For example, to set up remote instances for Sub-Dev 1:

**Procedure**
1. If IP address access control is enabled, log in to the remote instance and add Sub-Dev 1 as an exception.
2. On Sub-Dev 1, navigate to Team Development > Remote Instances.
3. Click New.
4. Define the remote instance, such as Dev-Parent, by completing the form (see table).

**Related Links**
Retrieve Completed Update Sets
Compare to Local Instance
Make This Your Parent
5. Click **Submit**.

6. Repeat step 1 through step 5 for each instance in the hierarchy that this instance needs to push and pull with (for example, Sub-Dev 2 and Sub-Dev 3).

Remote instance form

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique name describing the instance.</td>
</tr>
<tr>
<td>Type</td>
<td>Specify whether the remote instance is a development, test, or UAT instance.</td>
</tr>
<tr>
<td>Active</td>
<td>Specify whether the local instance communicates with the remote instance as a member of Team Development. Team Development operations such as comparing changes between instances or selecting a parent instance are only available for active remote instances.</td>
</tr>
<tr>
<td>URL</td>
<td>Specify the URL of the remote instance using the appropriate transfer protocol. Each remote instance record should have a unique URL. Creating duplicate records with the same URL can cause errors.</td>
</tr>
<tr>
<td>Username</td>
<td>Enter the user on the remote instance who authorizes Team Development operations on the instance. This user account must have an appropriate role on the remote instance.</td>
</tr>
<tr>
<td>Password</td>
<td>Enter the password of the authorizing user.</td>
</tr>
<tr>
<td>Short description</td>
<td>[Optional] Enter any other relevant information about the remote instance.</td>
</tr>
</tbody>
</table>

Enable a code review

You can require a code review of all changes pushed to an instance.

Procedure

1. Navigate to **Team Development > Properties**.

2. Select the **Yes** check box for *If this property is set to Yes, code review is required before pushing to this instance* (com.snc.teamdev.requires_codereview).

3. Click **Save**.
Setting this property adds the Code Review Requests module to the application menu and requires all changes pushed to this instance to remain in the Awaiting Code Review stage until someone in the Team Development Code Reviewers group approves them.

Select the parent instance
An instance can have multiple peer instances but only one parent instance.

About this task
The parent instance is the only instance you can pull changes from and push changes to.

The parent instance must be on the same release family as the local instance. For example, a development instance on the Geneva release family must have a parent instance also on the Geneva release family. If you select a parent from a different release family, the Team Development dashboard displays an error message and prevents you from pulling changes and reconciling. If you select a parent from a different patch release, the dashboard displays a warning message but allows you to pull changes and reconcile.

Do not use Team Development with production or test instances.

• Do not use a test or production instance as the parent instance in Team Development.
• Do not make any instance the parent of a production instance.
• Production instances should never have a parent.

When you back out a change on a Team Development instance, it backs out the change all the way back down the chain, including undoing the work on the source instance. This behavior can cause major problems on test and production instances.

Procedure
1. Navigate to Team Development > Team Dashboard.
2. In the control panel, click the appropriate link:

<table>
<thead>
<tr>
<th>Team dashboard control panel options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use &lt;instance name and URL&gt;</td>
<td>Selects the most recently defined remote instance as the parent instance.</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Select a different instance</td>
<td>Opens a dialog box where you can select another remote instance or define a new remote instance.</td>
</tr>
<tr>
<td>Register a new instance or List all remote instances</td>
<td>Opens the remote instance form or list, where you can define a new remote instance. These options are available when no remote instances are defined.</td>
</tr>
</tbody>
</table>

3. If you defined a new remote instance in step 2, repeat step 1 through step 2 and select the remote instance you defined.

The system initiates a reconcile, which compares the local instance to the parent. It then generates the list of local changes and calculates the number of changes that are ready to pull from the parent. The reconcile also validates the instance versions.

4. Pull all changes from the parent instance if both instances are in the same release family.

   **Note:** The parent instance is saved in the `glide.apps.hub.current` system property.

### Set up an instance hierarchy

Set up an instance hierarchy that best supports your development life cycle.

**About this task**

This example demonstrates how to set up an instance hierarchy where several peer sub-development instances have the same parent development instance, but a more complex configuration may be required to handle multiple project teams or other customer requirements.

Do not use Team Development with production or test instances.

- Do not use a test or production instance as the parent instance in Team Development.
- Do not make any instance the parent of a production instance.
- Production instances should never have a parent.
When you back out a change on a Team Development instance, it backs out the change all the way back down the chain, including undoing the work on the source instance. This behavior can cause major problems on test and production instances.

**Procedure**

1. Provision a parent development instance on the same software version, such as Dublin, as the target instance, such as production.
2. [Recommended] Clone the production instance to the parent development instance.
3. Provision sub-development instances on the same software version as the parent development instance.
4. **Optional:** Log in to the parent development instance and clone it to the sub-development instances.
5. On each sub-development instance:
   a. Define remote instance connections to other instances in the hierarchy that this instance needs to push and pull with.
   b. Select the parent instance.
   c. Pull all changes from the parent instance.
   d. Grant access rights to appropriate developers.

**Code reviews**

Team Development administrators can require that pushes undergo code review before accepting pushes.
When code review is enabled, pushing a change to the parent instance triggers the code review workflow. By default, users with the teamdev_code_reviewer role receive notifications to review changes and can approve or reject changes. The Team Development Code Reviewers has the teamdev_code_reviewer role.

For each change, reviewers can see the following information.

- Which remote instance the pushed change comes from.
- Who pushed the change to the parent.
- What the change is called.
- When the change was created.
- Which versions the change includes.

Reviewers must approve or reject a push from the Team Development application.

While changes are being reviewed on the parent instance, a child instance cannot do the following activities involving the parent instance:

- Push changes to the parent instance.
- Pull changes from the parent instance.
- Reconcile changes with the parent instance.
- Change the parent instance to another instance.
- Delete the remote instance record for the parent instance.

**Code review notifications**

You must enable email notifications on the instance requiring code review for that instance to send code review notifications.

The Team Development Code Review workflow sends notifications to members of the Team Development Code Reviewers group when:

- A push requires code review.
- A user cancels a push.

If the user who pushed the changes has a user record with an email address on the instance where code review was required, the user receives a notification when the approval stage is set to Complete (approved) or Code Changes Rejected.

The code review notifications contain the following information:
### Code review notification table

<table>
<thead>
<tr>
<th>Notification name</th>
<th>Table</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code review update for developer</td>
<td>Push or Pull [sys_sync_history]</td>
<td>• The push name&lt;br&gt;• The approval stage of the push (approved or rejected)&lt;br&gt;• A link to the instance where the code review request was made</td>
</tr>
<tr>
<td>Notify code reviewer of canceled review</td>
<td>Push or Pull [sys_sync_history]</td>
<td>• The user who canceled review&lt;br&gt;• The push that was canceled</td>
</tr>
</tbody>
</table>

### Code review workflow

The Team Development Code Review workflow manages how changes are pushed to the parent.

By default this workflow:

- Starts when changes are pushed to the parent instance.
- Verifies that the code review property is active on the parent instance.
- Sets the stage of changes requiring approval to **Awaiting Code Review**.
- **Notifies** the Team Development Code Reviewers group to review pushed changes, if configured.
- Loads approved changes or sets the stage to **Code Changes Rejected**.

⚠️ **Warning:** Use caution when modifying this workflow, as the code review feature may not function properly.
Exclusion policies
You can exclude certain files from change tracking by creating an exclusion policy.

When a change matches an exclusion policy, the change does not generate records in the local changes list. The change still generates local version records and Update Set records as normal. See Creating an Exclusion Policy.

Note: The exclusion policy applies to changes identified during a reconciliation operation. If you create an exclusion policy after a reconciliation, Team Development still tracks the changes until the next reconciliation.

Instance hierarchies
Team Development allows you to set up a distributed version control system between multiple ServiceNow instances where each instance acts as a source repository, or branch.

Developers use separate instances to work on different features, applications, or product releases at the same time. With Team Development, developers can share code between these instances and resolve collisions throughout the development process.

Team Development allows you to establish hierarchical relationships between instances and provides a mechanism for transferring changes between instances that integrates with the Update Set process where necessary. In a Team Development instance hierarchy, each non-production instance has a parent instance. Instances that have the same parent instance are peer instances. The shared parent instance becomes the central hub, or repository, and all peer instances synchronize to it.

Pulls and pushes
Developers synchronize their instances to the parent instance by pulling and pushing versions of customized records and resolving collisions between versions on the parent instance and the development instance.

Developers can compare peer instances to one another and share code or resolve collisions before pushing versions to the parent instance.

Pulling from the parent retrieves versions of records that have customer updates. Pulling retrieves all versions that have not already been pulled onto the development instance, including historical versions, and you cannot choose which versions to pull. You must resolve any collisions before proceeding with further pulls or pushes.
Pushing to the parent adds only the current development version to the parent, not all the development versions. You can choose which changes to push to the parent. Pushing creates a local Update Set on the parent that is marked as complete. Pushed versions are also tracked as local changes on the parent. Therefore, you can promote changes through your development and test hierarchy by transferring the Update Set or by pushing the local changes.

Comparing reports the differences between two peer instances. You can choose which versions to pull from a peer instance.

The Pushes and Pulls related list on the team dashboard displays the user who created a change and the remote instance where the change was created.

**Team Development process**

The basic Team Development process sets up the instance hierarchy, grants developer access rights, manages the movement of development changes from development instances to test instances, and promotes applications to the production instance.

**Procedure**

1. Set up the development instance hierarchy as described in Set up an instance hierarchy.
   - a. Provision development instances on the same software version as the target instance. For example, use the software version that is running on your production instance.
   - b. [Recommended] Clone the target to the development instances.
   - c. For each instance, define the parent instance.
   - d. [Optional] For each instance, define the peer instances.
   - e. For each instance, pull all changes from the parent instance.
2. For sub-development instances, grant access rights to appropriate developers.
3. Develop customizations on sub-development instances. Use the team dashboard to track development activities.
   - Pull versions from the parent instance, such as versions that were pushed from other sub-development instances. Reconcile any conflicts with the current local version, as necessary.
   - Track local changes. Queue changes that are ready to push to the parent development instance.
   - Compare versions on peer instances. Reconcile any conflicts.
4. When a feature is ready to promote to the parent development instance, push the current version of the customized records.

5. [Optional] Have code reviewers approve or reject the pushed version.

6. Test and promote the feature into production according to your testing and release management process.

**Related reference**
- Access rights for developers

**Related information**
- Team Development
- Push a version

**Team Development roles**
To use Team Development, developers must have admin access to their development instance.

To allow pushes to the parent instance, a remote instance connection must be defined with a user account that has admin access to the parent instance.

To limit developer access to the parent instance, see Granting Access Rights to Developers.

To use code review features, users must have the teamdev_code_reviewer role. See Code Review.

**Versions**
Version records track changes to a customized record over time so that administrators can compare or revert to specific versions later.

Administrators can also transfer versions between instances with Update Sets or team development.

**Version record navigation**
There are a variety of methods for viewing a list of versions for an object.

- For forms, right-click the header and select **Configure > Form Layout**.

- For lists, perform the appropriate action for the list version.
  - List v2: Right-click the header and select **Configure > List Layout**.
  - List v3: Open the list title menu and select **List Layout**.
• Click the **Show Versions** related link.

• For tables that use the update_synch attribute, add the **Versions** related list to the form. This list is on several forms by default, including, business rules, UI actions, and client scripts.

• For any customizable object, right-click the form header and select **Show Application File**, then scroll down to the **Related Record Versions** related list.

You can navigate from a version record to:

• The customized object: Click the **Show Related Record** related link.

• The application file record for the object: Click the **Show Application File** related link.

---

**Version record**

![Version record screenshot](image)

**Related Links**

- Revert to this version
- Show Related Record
- Show Application File

---

**Version List**

![Version list screenshot](image)

---

**Versions transferring**

Administrators transfer version records between instances by moving customizations with Update Sets or the Team Development application.
• Update sets: committing an Update Set adds versions. For each update in the Update Set, the version that corresponds to the update is added on the local instance.

• Team Development:
  ◦ Pulling retrieves from the parent instance all versions of customized records that have not already been pulled and then adds them on the local instance.
  ◦ Pushing adds to the parent instance only the current local version, not all the local versions.
  ◦ Loading changes from peer instances adds selected versions to the local instance.

Version records

The Update Versions [sys_update_version] table contains records that represent the state of a customizable object at a particular time.

A customizable record is any object that is tracked by Update Sets, such as business rules or script includes. A new version record is created automatically whenever a user changes a customizable record or changes the application file for the customizable record.

A baseline version record represents the version of a base system object as it was delivered in the most recent upgrade. Baseline versions are created only for objects that have been modified by a user, and they are updated each time the system is upgraded.

Update versions table

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A unique identifier for coalescing versions of the same customized record.</td>
</tr>
<tr>
<td>Record name</td>
<td>Name of the customized record.</td>
</tr>
<tr>
<td>Source</td>
<td>Indicator of how the version was added on the instance.</td>
</tr>
</tbody>
</table>
Update versions table (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>System Upgrade</td>
<td>from a software upgrade (the baseline version).</td>
</tr>
<tr>
<td>Update Set</td>
<td>from an update set that was created or committed on the instance.</td>
</tr>
<tr>
<td>Pull History</td>
<td>from a pull in Team Development.</td>
</tr>
<tr>
<td>State</td>
<td>Indicator of whether the version is or has ever been loaded on the instance.</td>
</tr>
<tr>
<td>Current</td>
<td>the version is currently loaded.</td>
</tr>
<tr>
<td>Previous</td>
<td>the version has previously been loaded on the instance. When a current version is replaced by a new version, it becomes a previous version.</td>
</tr>
<tr>
<td>History</td>
<td>the version was never loaded on the instance and was only inserted for historical purposes, such as when pulling versions from the parent in Team Development.</td>
</tr>
<tr>
<td>Application</td>
<td>The application for the customized record, if it is assigned to an application.</td>
</tr>
<tr>
<td>Payload</td>
<td>The data for this version of the customized record.</td>
</tr>
<tr>
<td>Additional fields on the list view</td>
<td></td>
</tr>
<tr>
<td>Reverted from</td>
<td>A reference to the older version record, if this version was created by reverting to an older version.</td>
</tr>
<tr>
<td>Fields that can be added by configuring the form</td>
<td></td>
</tr>
<tr>
<td>Instance Name</td>
<td>The name of the remote instance where the version was originally created.</td>
</tr>
<tr>
<td>Instance ID</td>
<td>The URL of the remote instance where the version was originally created.</td>
</tr>
<tr>
<td>Related lists on the form view</td>
<td></td>
</tr>
<tr>
<td>Version List</td>
<td>All versions of the customized record that are available on the instance.</td>
</tr>
</tbody>
</table>

Related information

Team Development
**Merge tool**

The Diff Merge tool enables administrative users to compare differences between two versions of a record. Administrators can compare field-level changes between two versions, apply changes using Move Right field-level copy functionality and then merge results, or choose to revert to the non-current version. You can access the Diff Merge tool by comparing versions, resolving conflicts, or resolving collisions, during development or after upgrades.
Accessibility Functions

The platform includes accessibility features that support Web Content Accessibility Guidelines (WCAG) 2.0 level A and make the interface accessible to users with disabilities. These features improve the user experience when accessing platform functions with screen readers and keyboard navigation.

In general, you can use the following set of standard keyboard navigation functions:

- Press **Tab** to navigate major groupings in a pre-defined sequence, including moving between standard interface controls (fields and lists) in a module, or between records within a tab.

- Press **Shift Tab** to move backwards in a pre-defined sequence.

Visually impaired users can navigate the Diff Merge tool. Screen readers can read all critical page content. All links and buttons can be reached when a section that is critical must be read. VoiceOver audible cues describe the content of the section that is necessary to read.

To enable accessibility functions, administrators should set these sys_properties:
1. Setting `glide.ui.javascript_editor` to false makes the following functions accessible:
   - Script fields (such as Script Include).
   - Side-by-side script comparison.

2. Setting `sys_properties` color settings enables high contrast visibility, which makes the left and right columns more accessible and easier to read by visually impaired users.
   - `mergetool.bg.left.highlight` - Left column cell color when values differ between versions.
   - `mergetool.bg.right.highlight` - Right column cell color when values differ between versions.
   - `mergetool.bg.left` - Left column cell color when version values are the same.
   - `mergetool.bg.right` - Right column cell color when version values are the same.

Related information

- Compare to the current version
- Compare a pushed version to a local version
- Compare two versions of an article
- Resolve conflicts for an individual record
- Resolve a collision in Team Development
- Revert a change
- View customizations and compare with current version

**Compare to the current version**

You can compare a version to the current version for any customizable object that a user has modified, such as a form layout or business rule. You can also compare the local and current pulled version of an object in Team Development. Administrators can suppress versions for specific tables.

**About this task**

To compare a version to the current version of an object:
Procedure

1. Open the Compare to Current page using one of the following methods:

<table>
<thead>
<tr>
<th>Option</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>From a Versions list</td>
<td>Right-click the version and select <strong>Compare to Current</strong>.</td>
</tr>
<tr>
<td>From the Update Versions form</td>
<td>Click the <strong>Compare to Current</strong> related link.</td>
</tr>
</tbody>
</table>

2. The Compare to Current page highlights the fields that differ. Review the differences. You have the following options:

<table>
<thead>
<tr>
<th>Option</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>To resolve the differences by choosing the previous version</td>
<td>• Team Development: Click <strong>Revert to Selected Version</strong>.</td>
</tr>
<tr>
<td></td>
<td>• For a version of an object: Click <strong>Use Local Version</strong> to maintain the local record as the current version. The pulled version is added to the version history for the record.</td>
</tr>
</tbody>
</table>
To resolve the differences by modifying the current version and saving the merged changes

**Note:** Some types of record do not support this method. See Limitations on updating records for more information.

<table>
<thead>
<tr>
<th>Option</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>You can either update the setting in the current record or move a setting from the selected version to the current version. To move a change, click the &gt; (Move Right) button for the field in the diff/merge tool. To work with scripts and text fields, click in the field and modify the text as needed. When the records meet your needs, click:</td>
<td></td>
</tr>
<tr>
<td>• Team Development: Click <strong>Save Merge</strong> to save the changes to the current version.</td>
<td></td>
</tr>
<tr>
<td>• Team Development: Click <strong>Use Pulled Version</strong> or <strong>Use Local Version</strong> option to accept or reject all changes, as appropriate.</td>
<td></td>
</tr>
<tr>
<td>• On the upgrade history Compare to Current form, the only option is <strong>Revert to Base System</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Related information**

- Merge tool
- Compare to the current version
- Compare a pushed version to a local version
- Compare two versions of an article
- Resolve conflicts for an individual record
- Resolve a collision in Team Development
- Revert a change
- View customizations and compare with current version

**Revert a change**

You can undo changes to a customized record by reverting to an older version.
Procedure

1. View a list of version records for an object.

2. Optional: Compare the current version to the older version to ensure that you are reverting the desired changes.

3. Right-click the older version and select **Revert to this version**. A confirmation dialog box appears.
   If reverting to this version results in data loss due to a database schema change, a warning message appears in the dialog box.

4. Click **OK** to confirm the action.

   - The current version is marked as a previous version.
   - A new version record is added that duplicates the version that you selected in the preceding step. This new version is marked as the current version.

**Note:** You can revert to the most recent baseline version. You cannot revert to an older baseline version.

Related information

- Merge tool
- Compare to the current version
- Compare two versions of an article
- Compare a pushed version to a local version
- Resolve a collision in Team Development
- Resolve conflicts for an individual record
- View customizations and compare with current version

Suppress versions

Administrators can configure a table so that it does not track customizations in the **Versions [sys_update_version]** table.

About this task

⚠️ **Warning:** If you suppress versions for tables, Team Development may work incorrectly, and you may be unable to compare and revert versions of records on the tables.
Procedure

1. Navigate to `sys_properties.list`.

2. Create a new property:
   - Name: `glide.update.suppress_update_version`
   - Type: string
   - Value: a comma-separated list of tables. The default value is `sys_user,sys_import_set_row`.

Versions and local changes

Version records track changes to a customizable record over time so that you can compare or revert to a specific version later.

A version record is created every time a developer changes a customizable record, so a single record can have multiple version records associated with it. A local change record is created or updated to reference the current version every time a developer modifies a customizable record, so a single record can have only one local change record associated with it.

Local change records track which customized records have changes on the development instance that are not on the parent instance so that you can collect changes in preparation for a push.
Team Development concepts

Developers can back out a local change to restore a previous version of a customizable record. The back out action sets the local customizable record to the last revision identified by a reconciliation action.

Custom applications tables

Custom applications table can help you review specific changes you have made during upgrades and determine ways to resolve conflicts.

You may learn that your file in the application has either been claimed by another application or skipped. Use these custom application tables to help you review your changes or determine how to take action for conflicts.
• Skipped Changes to Review related list
• Claim Outcomes to Review related list
• Upgrade Details related list

System update sets

An update set is a group of configuration changes that can be moved from one instance to another. This feature allows administrators to group a series of changes into a named set and then move them as a unit to other systems for testing or deployment.

An update set is an XML file that contains:

• A set of record details that uniquely identify the update set.
• A list of configuration changes.
• A state that determines whether another instance can retrieve and apply configuration changes.

Update sets track changes to applications and system platform features. This allows developers to create new functionality on a non-production instance and promote the changes to another instance.

⚠️ Warning: Update sets allow moving changes between instances that may be running different family release versions and different features. You can always load an update set created on an older family release on an instance running a newer family release. Loading an update set created on a newer family release on an instance running an older family release requires additional testing to determine compatibility. Updates from newer family releases may not produce the same functionality when moved to older family releases. In extreme cases, newer family release updates may cause outages or data loss on an older family release instance. Where possible, avoid moving updates from newer family releases to older family releases. Similar constraints apply to moving updates between instances running different versions of ServiceNow Store apps.

System properties

Administrators can exclude system properties from update sets by making them private. Keeping system properties private prevents settings in one instance from overwriting values in another instance. For example, you may not want a system property in a production instance to use a particular value from a development instance. See Add a system property.
Applications
Application developers have additional options with update sets such as:

• Create an update set for a specific version of an application.
• Specify which application tables to track in update sets.

Update set tables
Each update set is stored in the Update Set [sys_update_set] table, and the customizations that are associated with the update set, which are entries in the Customer Update [sys_update_xml] table, appear as a related list on the update set record.

When a tracked object is customized, a corresponding record is added or updated in the Customer Update [sys_update_xml] table and is associated with the user current update set. The associated application file properties are tracked and transferred along with the customized object in a single update record. A corresponding record is also added to the Versions [sys_update_version] table.

The Customer Update table contains one record per customized object, per update set. The Versions table contains one record per change to a customized object.

• Administrators can compare two versions and revert to a specific version of an object.
• Administrators can suppress versions for specific tables.
• Administrators can specify fields on tracked tables that you can change without skipping updates to the rest of the record (exclude the field from the update).

⚠️ Note: Do not directly modify Customer Updates [sys_update_xml] records.

Customizations tracked by update sets
Update sets can track customizations to application tables, fields, and records.

Update sets track customizations under these conditions:
• Where the table has an update_synch dictionary attribute.
• Where there is a special handler to track changes to multiple tables.
• Where the administrator has not excluded a field from updates.
In general, update sets capture configuration information but not task or process data. For example, update sets track service catalog item definitions and related configuration data like variables and variable choices. However, if you test the service catalog by placing orders, update sets do not track order requests, items, and catalog tasks.

Update sets have a limited capacity to transfer data as application files. For larger data transfers, export data and import it with an import set or web service.

**update_synch attribute**

To see the list of tables where customizations are tracked, navigate to System Definition > Dictionary and filter on attributes CONTAINS update_synch.

⚠️ Warning: Do not add the `update_synch` attribute to a dictionary record. When improperly used, this attribute can cause major performance issues or cause the instance to become unavailable. Adding this attribute is not supported.

A default rule blocks the use of the `update_synch` attribute on a table for which it is not predefined to avoid the following issues:

- Some core tables require special update handling because they represent information on multiple tables. When the `update_synch` attribute is added to these tables, duplicate update records are created, causing major conflicts that are difficult to troubleshoot and repair.

- Using the `update_synch` attribute to migrate data records between instances can cause performance issues, because it is not intended for this purpose. To migrate data, use an instance-to-instance import.

See Import sets.

**Special handlers**

Some changes require special handlers because they represent information on multiple tables. These changes are packaged into one update set entry so that all records are properly updated when the customization is committed. The following changes are tracked with special handlers:

- Workflows
- Form sections
- Lists
- Related lists
- Choice lists
• System dictionary entries
• Field labels

⚠️ **Warning:** The form sections, lists, related lists, choice lists, and field labels special handlers delete and reinsert records. This might cause unexpected results and data loss if there are fields referencing the tables.

### Choice lists

Update sets store both new and updated choice options as separate records in the Update Version [sys_update_version] and Customer Update [sys_update_xml] tables. For example, you create a new Activity [u_activity] table that extends the Task table. You then add a new choice option to the Task state field that is only visible for your extended table (for example, My State).

When you publish these changes as an update set, the update only contains update and version records for the choice you added to the u_activity table. The choice options in the task table are unaffected.

### Dictionary changes

Usually, using update sets prevent you from applying dictionary changes that result in data loss. Blocked dictionary changes include:

• Removing tables
• Changing a column data type

Update sets do not track the removal of tables from the system dictionary. Instead, customers must manually remove tables from the target instance. While update sets track data type changes, the target instance skips any change that results in data loss and instead adds a log message about the action. Customers can use the log to manually make data type changes on the target instance.

⚠️ **Note:** Update set previews do not check for type mismatch problems since the target instance skips changes resulting in data loss. Also, using update sets to delete a column from a table can cause data loss in certain circumstances. If there is data in the column on the target instance, that data is deleted, as well as the column itself, when the update set is committed. A warning message appears if you attempt to commit an update set that deletes a column. The message states that there are one or more delete updates that cause the data to be deleted, and it specifies what delete updates there are.
Homepages and content pages
Homepages and content pages are not added to update sets by default. Add pages to the current update set by unloading them.

Application changes
The system creates a separate update set for each application that only contains changes associated with the application. This ensures that access settings for each application are properly evaluated and applied when committing update set changes.

Default update set
Only one update set can be the default set for any application scope.

To set an update set to be the default set, you set the Default set field to true. When you set Default set = true, the following actions occur:

- The update set becomes the default update set for its scope.
- The system sets Default set = false for all other update sets with the same scope. This ensures that there is only one default update set for each scope.

Global default set
Use the global default update set to make changes to an instance without adding the changes to any user-created update sets. The global default update set is the set where Default set = true and application scope is global. The global default set (regardless of the Name of the set) provides system functionality and should not be changed, deleted, or moved between systems. Use this update set to make changes to an instance without adding the changes to any user-created update sets.

Auto-generated default set
At all times, to ensure that no updates to an instance are lost, the system ensures that there is a default set for the user’s current scope. If the system finds that a default update set does not exist (or is marked Ignored or Completed) for the current scope, then the system auto-generates an update set and sets Default set = true.

These are some common cases where the system auto-generates a default update set.
The very first time that an admin logs in, the system sets the system’s global default update set as the administrator’s update set. In addition, the application picker sets the administrator’s application scope to global. If a global default update set does not exist (or is marked *Ignored* or *Completed*), the system creates a new update set for the global application scope and performs the following actions:

- The system sets *Default set = true* for the new set.
- The system sets the name of the new set to start with the name of the former default set and appends the next numeral (in the sequence SetName, SetName 1, SetName 2, …, SetName n).
- The system sets the newly created set as the administrator’s update set.

When a user marks the default set for a scope as *Ignored* or *Completed* (not a recommended practice), the system immediately auto-generates a new default set for the scope.

The system auto-generates a new default update set for a scope when all the following conditions occur:

- You change application scope.
- Your preferred update set is *Complete* or *Ignored*.
- There is no In-Progress default update set for the new scope.

**Get started with update sets**

Because update sets make changes to an instance, review this information to avoid errors and performance issues. Learn how to plan the update process and avoid common mistakes.

**When to use update sets**

<table>
<thead>
<tr>
<th>Deployment option</th>
<th>Good for</th>
<th>Future considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update Sets</td>
<td>Storing changes to a baseline or installed application.</td>
<td>You can manually create update sets to store a particular application version. Use update sets to deploy patches or changes to installed applications.</td>
</tr>
<tr>
<td></td>
<td>Storing and applying a</td>
<td></td>
</tr>
<tr>
<td>Deployment option</td>
<td>Good for</td>
<td>Future considerations</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------</td>
<td>-----------------------</td>
</tr>
<tr>
<td></td>
<td>particular version of an application. Producing a file for export.</td>
<td><strong>Note:</strong> Do not use update sets to install applications. Instead, use the application repository or the ServiceNow Store to install applications.</td>
</tr>
<tr>
<td><strong>Application Repository</strong></td>
<td>Installing and updating applications on all company instances. Automatically managing application update sets. Restricting access to applications to the same company. Deploying completed applications to end users.</td>
<td>Consider uploading an application to the ServiceNow Store to share it with other users. Allows installation of and update to the latest application version only. Use update sets to store prior application versions. <strong>Note:</strong> If used with team development, publish applications only from a parent instance.</td>
</tr>
<tr>
<td><strong>Team Development</strong></td>
<td>Providing change management across multiple instances. Allowing multiple developers to work on applications. Organizations that have access to several non-production instances.</td>
<td>Consider providing each development team access to a dedicated development instance. Requires developers to manually merge colliding changes. Works only for instances owned by the same organization. <strong>Note:</strong> If used with the application repository, publish applications from a parent instance.</td>
</tr>
</tbody>
</table>

**Plan the update process**
Before working with update sets, create a standard process for moving customizations from instance to instance using this check list:
1. Check that both instances are on the same version. Customizations may not work if they rely on code that has changed between versions.

2. Determine the changes to make in a single update set. Complete your update sets as you finish small to medium-sized tasks. As update sets get larger, it becomes harder to review them, takes longer to identify specific changes within them, increases the risk of conflicts with other update sets, and takes more time to preview and commit them. This is especially true if the update sets contain schema changes or revisions to large workflows or if the set has to be backed out.

3. Ensure that all base system records have matching sys_id fields. Some base system records are created on an instance after provisioning and do not match between different instances, leading to problems with update sets. The best way to avoid this issue is to:
   - Provision production and non-production instances.
   - Clone the production instance onto the non-production instance.

4. Identify a common path for update sets to move from instance to instance and maintain that model. Never migrate the same update set from multiple sources. Move update sets from dev to test and then from test to production.

5. Plan for when to commit the update sets to production. Avoid committing an update sets to a production instance during business hours. The instance may perform slower temporarily as the update sets applies.

6. Make sure update set names are clear. Create a naming convention to coordinate changes from multiple developers and to reference when committing the changes to another instance.
   - If update sets are being generated as fixes for problems, consider including the problem ticket in the name (for example, PR10005 - Duplicate Email Issues Fix).
   - If you need more than one update set to address a problem, include a sequence number in the naming convention so that update sets are applied in the order that they were created (for example, PR10005 - Duplicate Email Issues Fix and PR10005.2 - Duplicate Email Issues Fix).

7. Understand the following about update sets:
   - What records are generated.
   - Which customizations are tracked.
• Which dictionary changes are valid.
• Which customizations can be backed out (reversed) once applied.

8. Before making any customizations, double-check that the correct update set is selected.

Working with update sets
Review this information to avoid errors and performance issues.

• Do not delete update sets. If an update set is deleted, any updated records may be overwritten in the next update.

• Do not include the system_id field from the ldap_server_config record in an update set. An update set from a working configuration points to the wrong system_id node for the target instance and does not work.

• Do not back out the Default update set. This action causes damage to the system.

• Never change the Update Set field value (update_set) in a Customer Update record (sys_update_xml). If a customization is made in the wrong update set, take the following action:
  1. Switch to the desired update set.
  2. Modify the object (record) that was originally changed. You can make a trivial change, such as adding a field.
  3. Save the record.
  4. Back out the change just performed, and then save the record again.

     This action ensures that the latest version of the object is included in the desired update set and prevents duplicate updates for the same object in a single update set.

• Do not mark an update set as Complete until it is ready to migrate. Once an update set is complete, do not change it back to In progress. Instead, create another update set for the rest of the changes, and make sure to commit them together in the order that they were created. Naming conventions may help in this case (for example, Performance Enhancements and Performance Enhancements 2).

• Do not manually merge updates into an update set. Always use the Merge Update Sets module. This tool compares duplicate files between update set and selects the newest version.

• If a committed update set has a problem in the test instance, build the fix in another update set in the development instance. Commit this set to the test instance, and then make sure both sets are migrated to the production instance and committed in the order they were made.
Always preview an update set before committing it.

Set completed update set on the production instance to **Ignore**. This state ensures the update set is not reapplied when cloning the instance.

Keep a to-do list of manual changes and data loads that need to be completed after an update set is applied.

Do not make too many changes at one time. Verify that the correct changes have been made incrementally.

### Update set administration

Manage how update sets store, retrieve, preview, and apply configuration changes between instances.

Administrators have the following options with update sets.

- Create an update set to store local changes.
- Select the current update set to store local changes.
- Commit an update set to prepare it for distribution.
- Compare update sets to determine what differences they contains.
- Merge separate update sets into a single update set.
- Create an external file from an update set.
- Retrieve update sets from remote instances.
- Apply retrieved update sets.
- Back out changes applied from an update set.
- Set system properties related to update sets

### Grant access to the update set picker

The update set picker allows users to choose an update set for making and tracking customizations. By default, only administrators can use the update set picker. You can **grant access to additional users**.

### Automatically preview retrieved update sets

By default, the system automatically previews update sets as soon as it has retrieved them. To turn off this behavior, set the system property `glide.update_set.auto_preview` to **false**: in the navigator filter, type
sys_properties.list then navigate to the glide.update_set.auto_preview property and edit the value field.

Track an application table
Application developers can track application changes in an update set to save or distribute a particular version of an application. During table creation, set Extends Table to Application File [sys_metadata].

Grant access to the update set picker
Enable a non-administrative user to use the update set picker.

Before you begin
Role required: admin

About this task
The update set picker appears on the Settings panel. The picker allows users to choose an update set for making and tracking customizations. By default, only administrators can use the update set picker. You can grant access to additional users.
Procedure

1. Grant the user role read access to the Update Set table [sys_update_set].

2. Enable users to see the update set picker on the Settings panel.
   a. Add the system property `glide.ui.update_set_picker.role` to the System Properties table.
   b. Set the value of `glide.ui.update_set_picker.role` to the role for which you want to give access.
Overwrite customizations during an upgrade

Specify which customized objects you want to replace during the next upgrade. By default, the upgrade process skips changes to customized objects.

About this task

The systems tracks the configuration changes you make such as modifying the dictionary record for a table or field. Each configuration change you make has a corresponding record in the Customer Update [sys_update_xml] table where the Replace on upgrade field is set to false. You may want to overwrite your customizations with the next software version. For example, you may change a script to implement a temporary workaround for a problem that is fixed in the next version. You would want to overwrite your workaround when upgrading to the next version to ensure that you receive any future enhancements to the script.

Procedure

1. Open the customized object (for example, the ArrayUtil script include).
2. Right-click the header and select Show Latest Update.
3. Configure the form to add the Replace on upgrade field, if necessary.
4. Select the Replace on upgrade check box and click Update.
   The customized object will be replaced on the next upgrade.

Creating, testing, and moving customizations

Use these procedures to create, test, and move customizations from a development system to a production system.

Three-step import process

A common process for developing customizations with update sets involves moving changes from development, to test, to production instances.

1. Create an update set on the development instance.
2. Make customizations and changes on the development instance.
3. Mark the update set as Complete.
4. Log in to the test instance and retrieve the completed update set from the development instance.
5. Commit the update set on the test instance, and test customizations thoroughly.
6. If the update set has problems in the test instance, repeat steps 1 - 5 to develop the fix on the development instance with another update set.
7. Log in to the production instance and retrieve the completed update set from the development instance. If the update set required a fix, retrieve both update sets.

8. Commit the update set on production. If the update set required a fix, commit both update sets in the order they were made.

**Two-step import process**
If your development environment consists of only two instances, you can combine your development and testing instances into a single staging instance.

1. Create an update set on the staging instance.
2. Make customizations and changes on the staging instance.
3. Mark the update set as Complete.
4. Test customizations thoroughly on the staging instance.
5. If the update set has problems, repeat steps 1 - 4 to develop the fix on the staging instance with another update set.
6. Log in to the production instance and retrieve the completed update set from the staging instance. If the update set required a fix, retrieve both update sets.
7. Commit the update set on production. If the update set required a fix, commit both update sets in the order they were made.

**Update set use**
These procedures help you manage your customizations and resolve potential collisions before you move them to another instance.

Before using update sets, review Get started with update sets to learn when to use and when not to use update sets, and how to plan the update process and avoid common mistakes. Then, create an update set and use it to make changes on a development instance. You can report on updates, merge update sets, and compare update sets to ensure the desired changes are ready to move.

When the update set is completed, you can transfer the update set to another instance according to your test process. See Update set transfers for details.

**Create and select an update set as the current set**
Create an update set to store customization changes and select it as the current set.
About this task
You can select an update set as the current set when you create it, or you can select it later from the Settings panel.

Procedure
1. Navigate to System Update Sets > Local Update Sets and click New.
2. Complete the form from the fields in the table.
3. Click Submit to create the update set. If the picker is enabled and the update set is in the In progress state, click Submit and Make Current to select the new update set as the target for configuration changes.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique name for the update set. You can use naming conventions to organize update sets. For example, add the problem number to the name of the update that fixes it, identify the application scope, or use sequence numbers to keep track of the order in which update sets need to be committed.</td>
</tr>
<tr>
<td>State</td>
<td>Select <strong>In progress</strong> for a new update set. Selecting an In progress update set tracks customizations in the update set. The update set picker only displays In progress update sets. Select <strong>Completed</strong> only when you are certain that the update set is complete. After it is marked Completed, do not set it back to In progress. Instead, create a new update set with further customizations, and make sure to commit the update sets in the order that they were marked Completed. Use Completed update sets to transfer changes from one instance to another.</td>
</tr>
</tbody>
</table>
Field | Description
--- | ---
Select **Ignore** when you are no longer working on an update set but do not want it to be transferred to another instance. You should always set **Completed** update sets on the production instance to **Ignore**. This state ensures the update set is not committed again when cloning the instance.

Update set picker only displays **In progress** update sets. See **Select the current update set in System Settings**.

Select **Completed** only when you are certain that the update set is complete. After it is marked **Completed**, do not set it back to **In progress**. Instead, create a new update set with further customizations, and make sure to commit the update sets in the order that they were marked **Completed**. Use **Completed** update sets to transfer changes from one instance to another.

Select **Ignore** when you are no longer working on an update set but do not want it to be transferred to another instance. You should always set **Completed** update sets on the production instance to **Ignore**. This state ensures the update set is not committed again when cloning the instance.

| Release date | Enter the date on which you plan to release the update set. |
| Application | Populates the application scope that is currently selected in the Application picker. All changes in the update set apply only to the current scope. |
| Description | Enter a description of the update set. |

**Select the current update set in System Settings**

You can change the current update set at any time, using the update set picker in the System Settings panel.

**Before you begin**
Role required: admin
Procedure

1. To select the update set from the picker, click the gear icon in the header bar to open the System Settings panel.

2. In the System Settings panel, select **Developer**.

3. Select the update set you want to make current from the choice list in the **Update Set** field and close the settings panel.
View customizations and compare with current version

View the customizations that make up an update set and compare the update to the current version.

Before you begin
Role required: admin

About this task
The Customer Update \[sys_update_xml\] table contains one record per customized object.
The customer update record specifies:
• The update set containing the customized object.
• The type of action applied to the customized object.
  ◦ INSERT
  ◦ INSERT_OR_UPDATE
  ◦ UPDATE
  ◦ DELETE
• The type of object customized.
• The target object of the update.
• The sys_id of the customized object (if it is a change to a particular record).
• The user who customized the object.
• The date and time the object was customized.

Procedure
1. Navigate to System Update Sets > Local Update Sets.
2. Click the update set name.
3. View the Customer Updates related list.
You can compare any update to the current version. Right-click the update record and select **Compare to Current**.

### Related information

- Merge tool
- Compare to the current version
- Compare a pushed version to a local version
- Compare two versions of an article
- Resolve conflicts for an individual record
- Resolve a collision in Team Development
- Revert a change

### Navigation between records

You can navigate between a customer update record and the customized object or the application file for the object.

Navigate from an update record to:
• The customized object, such as the application menu record: Click the Show Related Record related link.

• The application file record for the object: Click the Show Application File related link.

Navigating to the application file

To navigate from a customized object or an application file to the current customer update record: Click the form header and selecting Show latest Update.
View a report on customizations and configuration changes

The base system provides reports for changes to the Incident table and changes by the current user.

Procedure

1. Navigate to **Reports > View / Run** and locate the **Customer Update** section.
2. Run any of the available reports or create a new report. The following reports are available:
   - **Application Changes (Incident)**: Displays all changes made to the Incident table. Select a different table and run the report again to view all changes to another application.
   - **My Changes**: Displays all changes created or updated by the current user, grouped by table name.

Merge update sets

You can merge multiple update sets into a single update set. This capability is supported for backward compatibility with earlier releases of the ServiceNow® platform. The newer batch update sets feature accomplishes the same outcome with a more predictable and robust solution.

Before you begin

ℹ️ **Note**: You cannot "unmerge" update sets once they have been merged. To learn more, see [Update set batching](#).

Procedure

1. Navigate to **System Update Sets > Merge Update Sets**. By default, the list is filtered to only show update sets that are **In progress**.
   Alternatively, navigate to **System Update Sets > Merge Completed Sets**. By default, the list is filtered to only show update sets that are in the **Complete** state. For example, you may want to use this filter after pushing changes or transferring update set from a development to a test instance.
2. Filter the list to show only the update set that you want to merge. You can only merge update sets that belong to the same application.
3. Enter a **Name** for the new update set. Updates are moved to this new update set during the merge process.

4. **Optional:** Enter a **Description** for the update set.

5. Click **Merge**.

6. In the confirmation dialog box, click **OK**.
   - The new update set is created.
   - The most recent change for each object is moved from the original sets to the new set. Only changes that are not merged into the new set remain in the original sets. A message indicates how many updates were moved and how many were skipped. For example, if both update sets modify the Incident form, only the most recent change is moved to the new update set. The other modification remains in its original update set to provide a record of the changes that were not moved.

   **Note:** The system determines which record is the most recent by comparing the **Updated** field for the records, NOT the **sys_updated_on** value in the payload.

7. [Recommended] Verify that the correct changes were moved to the new set by scrolling down to the **Merged Update Sets** related list and opening the old update set records.
8. [Recommended] Delete or empty the original update sets to avoid committing an older change by mistake. The system does not remove updates that were not merged into the new set. DO NOT move updates "left behind" in old sets into the new set.

Compare local update sets
Administrators can preview local and remote (retrieved) update sets and compare the sets with one another to resolve conflicting changes.

About this task
Compare local update sets to identify collisions and ensure that the proper changes are being committed. Resolve all conflicts before moving an update set between instances.

Procedure
1. Navigate to System Update Sets > Local Update Sets.
2. Select the check boxes beside the update sets to compare.
3. In the Action choice list, select Compare Update Sets. The progress screen appears as ServiceNow generates the collision report.
4. Click Go to the Collision Report when the report is complete.
   The Update Set Collisions list appears, showing all the changes in the selected sets.
5. Inspect the list for collisions by locating duplicate Collision Numbers that show the same change in separate update sets.
6. Resolve the collision by deleting the unwanted update record from one of the update sets.

   a. Click the link in the Sys update column for the unwanted update (sys_ui_list_incident_null in the example).
   
   b. Click Delete.

   **Note:** You must open the update record to delete the record. You cannot delete the update by selecting the check box for the entry in the Update Set Collisions list and using the Delete action. When you delete the update record, the customization is not backed out of the instance. Only the record of the customization is deleted.

   **Customer updates**

   ![Customer updates](image)

7. Run the comparison again to make sure all collisions have been resolved.

**Update set collision resolution**

A collision is an update that has a newer local update.

The platform detects collisions by comparing the values in the Name and Updated fields of the customer update record from each update set. If the name matches but there are different update date values, then there is a collision.

When a customer update is moved from one instance to another, it may be re-written to match the target instance. The re-write can involve changing the update name of the customer update and one or more sys_ids within the update. The re-writes are done when the record or the reference field is for a table that uses a coalesce strategy. This ensures that the customer update will...
be applied to the correct record. For example, if the sys_dictionary record for tablename.fieldname has sys_id 123456789 on instance A and sys_id 987654321 on instance B, when a customer update that refers to that record is retrieved from instance A and recorded in the sys_update_xml table on instance B, references to 123456789 are updated to read 987654321.

Coalesce strategies

Update sets can detect collisions between identical records that you independently create on separate instances. To detect such collisions, the record must have a coalesce strategy based on coalescing columns. Because collision detection depends on uniqueness of tables, the tables must be unique when the coalescing columns are combined. Records that are not listed here will not collide if the same record is created separately on different instances.

<table>
<thead>
<tr>
<th>Type</th>
<th>Coalescing Columns</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_db_object</td>
<td>name</td>
</tr>
<tr>
<td>sys_dictionary</td>
<td>name, element</td>
</tr>
<tr>
<td>sys_choice_set</td>
<td>name, element, language</td>
</tr>
<tr>
<td>sys_documentation</td>
<td>name, element, language</td>
</tr>
<tr>
<td>sys_properties</td>
<td>name</td>
</tr>
<tr>
<td>sys_report_chart_color</td>
<td>name, element, value</td>
</tr>
<tr>
<td>sys_ui_form</td>
<td>name, view, sys_domain</td>
</tr>
<tr>
<td>sys_ui_message</td>
<td>documentkey, language</td>
</tr>
<tr>
<td>sys_ui_list</td>
<td>name, view, sys_domain, element, relationship, parent</td>
</tr>
<tr>
<td>sys_ui_section</td>
<td>name, view, caption, sys_domain</td>
</tr>
<tr>
<td>sys_ui_related_list</td>
<td>name, view, related_list, sys_domain</td>
</tr>
<tr>
<td>sys_ui_view</td>
<td>name</td>
</tr>
<tr>
<td>sys_user_role</td>
<td>name</td>
</tr>
<tr>
<td>sys_user_group</td>
<td>name</td>
</tr>
<tr>
<td>sys_wizard</td>
<td>name</td>
</tr>
</tbody>
</table>

How customer update record names affect collisions

To understand coalescing, it helps to understand how records that do not coalesce work. For most record types, when a customer update is moved to a new instance, the system does not detect collisions for the following reason:
• When you create a record, it receives a unique sys_id. For most record types, the sys_id becomes part of the customer update record name. For example: sysevent_email_template_9e1998c078b71100a92ecac80df1d39.

• Creating an identical record in the same table on another instance produces a customer update record name with a different sys_id. For example: sysevent_email_template_10b958c8653311005840134572f8e020

As a result, even though the records might be otherwise identical, the records have different names so the system does not detect the collision.

Coalescing records, in contrast, use the following approach to naming records and determining collisions: The following customer update record types use some or all of their coalescing columns instead of the sys_id in their names.

• sys_dictionary
• sys_documentation
• sys_choice_set
• sys_ui_list
• sys_ui_related_list

The resulting identical record name in each instance helps the system to identify collisions even if the records have different sys_ids.

When a customer update is moved from one instance to another, it may be re-written to match the target instance. The re-write can involve changing the update name of the customer update and one or more sys_ids within the update. The re-writes are done when the record or the reference field is for a table that uses a coalesce strategy. This ensures that the customer update will be applied to the correct record. For example, if the sys_dictionary record for tablename.fieldname has sys_id "123456789" on instance A and sys_id "987654321" on instance B, when a customer update that refers to that record is retrieved from instance A and recorded in the sys_update_xml table on instance B, references to "123456789" are updated to read "987654321".

Preventing duplicate records

• Transfer data with update sets rather than recreating it on separate instances to ensure the records have the same sys_id.

• Export and import records as XML files to ensure the records have the same sys_id. See Export and import XML files.

• Enable a unique index for the table from the system dictionary. See Table administration.
Note: The default records included in the baseline system will always have the same Sys ID because the instance imports the records as XML files during instance provisioning.

Mark an update set complete

When you have completed the customizations and compared local update sets to resolve conflicts, mark the update set as Complete.

About this task

Mark an update set as Complete only when it is ready to migrate. Once an update set is complete, do not change it back to In progress. Instead, create another update set for the rest of the changes, and be sure to commit them together in the order that they were created. Naming conventions may help in this case (for example, Performance Enhancements and Performance Enhancements 2).

Procedure

1. Open the update set record.
2. Change the State of the update set from In progress to Complete.
   • The update set is available for other instances to retrieve.
   • No additional customizations are tracked in the update set.

Save an update set as a local XML file

Administrators can export an update set as an XML file to save a specific version of an application or set of changes.

About this task

Typically you create an XML file of an update set when one of the following conditions apply:

• The two instances do not have network connectivity so you cannot retrieve update sets from the remote instance nor create a data source to pull, or import, data directly from the source instance.

• You do not want to provide administrator credentials to the source instance (for example, you do not want to share an administrator password with
people outside your company) so you cannot retrieve update sets nor create a data source.

- You want to back up important customizations locally.

The ability to export and import customizations as an XML file is provided by the following UI Actions:

- **Export to XML** on the Update Set [sys_update_set] table.

The **Export to XML** UI Action on Update Set [sys_update_set] table calls a processor called `UnloadRetrievedUpdateSet`, which transforms a local update set into a retrieved update set, exports the retrieved update set with its related list, and then deletes the temporary update set, if necessary.

Both **Export to XML** UI actions depend on the script include `ExportWithRelatedLists`, which exports a record and manually defined related lists to a single XML file.

**Procedure**

1. Navigate to **System Update Sets** and click either **Local Update Sets** or **Retrieved Update Sets**.
2. Select an update set that is in the **Complete** state.
3. On the Update Set form, click the **Export to XML** Related Link.
4. Save the XML file.
   An XML file is created. When the file is uploaded to another instance, it appears as a retrieved update set regardless of whether it is local or retrieved on the instance where it is created.

**Load customizations from a single XML file**

Administrators can load an update set XML file to apply a specific version of an application or set of changes.

**Procedure**

1. Elevate privileges to the security_admin role.
2. Navigate to **System Update Sets > Retrieved Update Sets**.
3. Click the link **Import Update Set from XML**.
4. Click **Choose File** and select an XML file.

5. Click **Upload**.

   The customization is now available as a retrieved update set with state **Loaded**.

6. Follow standard procedure to commit the update set.

### Retrieved update sets

<table>
<thead>
<tr>
<th>Name</th>
<th>State</th>
<th>Update source</th>
<th>Description</th>
<th>Loaded</th>
<th>Released</th>
<th>Committed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Loaded</td>
<td>Automatically created by the system</td>
<td>2010-01-14 10:04:01</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dummy</td>
<td>Loaded</td>
<td></td>
<td>2010-01-14 09:53:39</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Related Links

- Import Update Set from XML

### Update set transfers

When an update set is completed, you can transfer it between instances to move customizations from development, through testing, and into production.

⚠️ **Warning:** Update sets allow moving changes between instances that may be running different family release versions and different features. You can always load an update set created on an older family release on an instance running a newer family release. Loading an update set created on a newer family release on an instance running an older family release requires additional testing to determine compatibility. Updates from newer family releases may not produce the same functionality when moved to older family releases. In extreme cases, newer family release updates may cause outages or data loss on an older family release instance. Where possible, avoid moving updates from newer family releases to older family releases. Similar constraints apply to moving updates between instances running different versions of ServiceNow Store apps.

### Transferring with IP access control

If IP address access control is enabled on the source instance or the source instance resides in a different datacenter than the target instance, complete these tasks before transferring an update set:

1. Contact Customer Service and Support to find out the IP addresses of all application nodes supporting your instance.

2. On the source instance, navigate to System Security/IP Address Access Control. Add the IP address from step one as an exception.
Transferring with basic authentication
If the source instance has basic authentication turned on for SOAP requests, you must use valid credentials to retrieve update sets.

Transferring with an XML file
You can unload an update set as an XML file and then transfer it to another instance. See Save an update set as a local XML file for details.

Retrieve an update set
Retrieve completed update sets from another instance.

About this task

⚠️ Warning: Update sets allow moving changes between instances that may be running different family release versions and different features. You can always load an update set created on an older family release on an instance running a newer family release. Loading an update set created on a newer family release on an instance running an older family release requires additional testing to determine compatibility. Updates from newer family releases may not produce the same functionality when moved to older family releases. In extreme cases, newer family release updates may cause outages or data loss on an older family release instance. Where possible, avoid moving updates from newer family releases to older family releases. Similar constraints apply to moving updates between instances running different versions of ServiceNow Store apps.

Procedure
1. If IP address access control is enabled on the source instance, set up the target instance as an exception.
2. On the target instance, navigate to System Update Sets > Update Sources and click New.
3. Specify the connection settings as described in the table.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique name for the instance.</td>
</tr>
<tr>
<td>Type</td>
<td>Specify whether the remote instance is a development, test, or UAT instance.</td>
</tr>
</tbody>
</table>
### Field | Description
--- | ---
Active | Specify whether the local instance can transfer update sets to the remote instance. You can transfer update sets only to active remote instances.
URL | Specify the URL of the remote instance using the appropriate transfer protocol. Each remote instance record should have a unique URL. Creating duplicate records with the same URL can cause errors. The remote instance must be on the same release family as the local instance.

**Note:** You cannot change the URL after the system verifies the connection. Use the Active field to deactivate unwanted remote instances.
Username | Enter the user on the remote instance who authorizes transferring update sets to this the instance. This user account must have the admin user role on the remote instance.
Password | Enter the password of the authorizing user.
Short description | [Optional] Enter any other relevant information about the remote instance.

4. Click **Test Connection**.
   - If the connection is successful, a confirmation message appears.
   - If the connection fails, a warning message identifies the cause of the failure.
5. If the connection fails, modify the settings to establish connectivity.
   - You must establish connectivity before you can save the connection settings.
   - You may want to modify the source instance (for example, change the password).
6. Right-click the form header and select **Save**.
   - Any update sets marked as **Completed** are transferred from the source instance to the target instance. Update sets that already exist on the target instance are skipped.
   - The confirmation page provides detailed messages about how many update sets were transferred and how many were skipped.
   - To view retrieved update set, navigate to **System Update Sets > Retrieved Update Sets**.
What to do next

If the system property `glide.update_set.auto_preview` is set to `true`, the system automatically starts the preview process after the update set is retrieved. If this property is `false`, you must start the process manually. For more information on the preview process, see Preview a remote update set.

Preview a remote update set

Previewing compares an update set retrieved from a remote instance to updates on the local instance to detect potential problems. You must preview an update set and address all problems before you can commit the update set.

Procedure

1. If the system property `glide.update_set.auto_preview` is set to `true`, the system automatically starts the preview process after the update set is retrieved. If this property is `false`, the preview process must be started manually.

   
   b. Click Preview Update Set.

   For large update sets, the preview process may require a significant amount of time. If necessary, you can cancel the preview process by clicking the Cancel button on the progress dialog box.

   The Update Set Preview page shows results and lists problems. Read the information and click Close.

2. On the Retrieved Update Set form:

<table>
<thead>
<tr>
<th>Option</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>If no problems were detected</td>
<td>Click Commit Update Set to commit all changes on the instance without reviewing the preview results.</td>
</tr>
<tr>
<td>If problems were detected</td>
<td>Address each problem in the Update Set Preview Problems related list.</td>
</tr>
</tbody>
</table>
3. Optional: Open the update set record and click **Show All Preview Records** to make sure the correct updates are being committed.

4. Optional: Open the update set record and click **Run Preview Again** to run the comparisons again. Review the **Update Set Preview Problems** related list to ensure that the correct updates are being committed.

**Review a preview record for an update set**

The process of previewing an update set creates a preview record for each update. You can review the preview records to make sure that the correct updates are being committed.
Procedure

1. Open the Update Set record and preview the update set.
2. Click the Show All Preview Records related link.
3. Click the Disposition to open a preview record and then review the information (see table).
4. Fill in the fields on the form, as appropriate.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disposition</td>
<td>Indicates when a collision is detected:</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>• <strong>Collision/Update. Collision/Insert, or Collision/Delete:</strong> the change is older than a change to the same object on the local instance.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Update, Insert, or Delete:</strong> the change does not conflict with a change on the local instance.</td>
</tr>
<tr>
<td>File differences</td>
<td>Compares the most recent version of the object on the local instance with the version in the update set. Differences are marked with a color key. Deletions are highlighted in red, additions in green, and modifications in yellow.</td>
</tr>
<tr>
<td>Proposed action</td>
<td>Indicates how to handle the change when the update set is committed.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Commit:</strong> Accept the change in the remote update. The default proposed action for every preview record is Commit, even if a newer update exists on the instance.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Skip:</strong> Reject the change.</td>
</tr>
</tbody>
</table>

5. If necessary, resolve any problems listed in the **Update Problems** related list.
6. In the Proposed action field, select the action to take when committing the update set.
7. Click **Update** to save the action.
8. Repeat the process for every preview record.

**Resolve a preview problem in an update set**

Preview an update set to detect and resolve problems that may occur if you commit the updates on the local instance.

**Procedure**

1. Navigate to **System Update Sets > Retrieved Update Sets**.
2. Open the update set record and scroll to the **Update Set Preview Problems** related list.
3. Review each problem description to determine the cause and resolve the problem.

**Update Set Preview Problems**

**Missing Object**

**Example problem text:** Could not find a record in sys_ui_policy referenced in this update.

**Description:** The object or a referenced object does not exist on the target instance. For example:

- An update modifies the form layout for a table that has not been created in the local instance.
- A UI policy action is included in the update set, but the parent UI policy is not.

**Resolution:** Create another update set on the source instance to transfer the missing object to the local instance, or create the object on the local instance. Use these **Available Actions** to assist in problem resolution:

- **Find missing field** or **Find missing record:** Opens a new window and searches the source instance for the missing field (dictionary entry) or record.
- **Find missing update:** Opens a new window and searches the source instance for the update record that corresponds to the missing field or record.

**Collision**
Example problem text: Found a local update that is newer than this one

Description: A change in the update set is older than a change to the same object on the local instance.

Resolution: Compare the two updates and determine which version to use. To use the version on the local instance, select **Skip remote update**. To use the version in the update set, select **Accept remote update**. Use these **Available Actions** to assist in problem resolution:

- **Compare with local**: opens the preview record, which provides a comparison of the differences between the local version and the version in the update set.
- **Show local field** or **Show local record**
- **Show local update**

### Uncommitted update

Example problem text: Could not find a table field (u_case.u_reference) referenced in this update, but did find it in another uncommitted update set

Description: The object exists in another remote update set that has not been committed.

Resolution: Commit the other remote update set first or move this update to the other update set. Use these **Available Actions** to assist in problem resolution:

- **Show uncommitted update**: opens the update record in the other remote update set.
- **Show uncommitted Update Set**: open the other remote update set record.

### Table to be deleted has data

Example problem text: Found a row in the table that is going to be deleted

Description: One difference between table deletes and other metadata deletes is that the table data is lost when the table is deleted. (If the table is empty (no rows), then no problem is generated.)
Resolution: The problem must be ignored (delete will happen) or skipped (delete will not happen) before the update set can be committed. You can restore the table, but the restore does not bring back the data.

You are not allowed to delete system tables (ServiceNow tables) or tables outside your application scope.

Application scope validation issue
Description: The previewer identifies the following combination of states as a problem:
• The scope for the update set is not Global and
• The application is not found on the target instance and
• The application is not included with the update set and
• The application is not found on the ServiceNow Store.

Resolution: Transfer the update set only to instances that include the application scope or ensure that the update set includes the application.

Conflict within a single batch
Example problem text: This update has conflicts within the update set with the same name. Resolve the issue on the source system and re-preview or choose a specific update to use.

Description: Two or more update sets within the same batch have conflicting changes. The Update Set Preview Problems list contains a record for each update set with a conflicting change.

Resolution: Compare the conflicting update sets and determine which version to use. If you know which update set is the correct one to use, select the row for that set and click Accept this collision. Otherwise, click Compare Collisions to compare the conflicting update sets.
Compare two Updates

From this screen, you can compare any two of the conflicting update sets and choose the update set to commit.

Commit an update set

When you have previewed an update set and have resolved any issues, commit the update set. Committing an update set applies all changes to the instance and creates a local copy of the update set that contains an update record for every change.

Procedure

1. Navigate to **System Update Sets > Retrieved Update Sets** and open the update set.

2. Resolve any problems. You cannot commit an update set until all problems are resolved.

3. Click **Commit Update Set**.
   - Click **Cancel** to return to the preview and reevaluate the change. None of the updates are committed.
   - Click **OK** to skip the change and continue committing the changes that are marked as **Commit**.

If the update set contains one or more DELETEs for schema, the system displays a warning. The warning lists up to five updates that may contain problems. If more than five updates have potential problems, the system provides a link.

When the system successfully commits an update set, it displays a completion page.

4. [Recommended] Click **Commit log** on the confirmation page, or navigate to **System Update Sets > Update log** and filter for the update set name.
• Look for warnings that contain the text **unsafe edit**. The system automatically skips any changes that results in data loss, such as changing the type of a field that contains data. You must manually make any of these changes, if necessary. Use caution when making changes that affect production data.

• Look for errors that indicate which records failed to commit and why.
Create a new update set to address those failures, if necessary.

5. **Recommended** When you are no longer working on the update set but do not want it transferred to another instance, navigate to **System Update Sets > Local Update Sets** and open the local update set record. Change the State to **Ignore**.

**What to do next**
For completed update set on the production instance, you should always change the state to **Ignore**. This state ensures that the update set is not committed again when cloning the instance.

**Back out an update set**
You can back out changes to existing records for any committed update set.

**About this task**
Backing out an update set creates delete updates in the current update set. If you commit, back out, and then reapply a remote update set, errors appear in the previewer because the deleted updates are considered more recent changes and cause collisions.

⚠️ **Warning:** Do not back out the **Default** update set. This action can damage the configuration of the instance.

The back out process reverses both record-level updates and changes to the dictionary. Some changes caused by a back-out can result in data loss. These are the expected results of the back-out process:

<table>
<thead>
<tr>
<th>Customer Update</th>
<th>Result of the back out action</th>
</tr>
</thead>
<tbody>
<tr>
<td>A new table</td>
<td>The table is dropped from the database, deleting any data from it.</td>
</tr>
<tr>
<td>A new field</td>
<td>The field is dropped from the database, deleting any data from it.</td>
</tr>
<tr>
<td>Customer Update</td>
<td>Result of the back out action</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>A deleted field</td>
<td>The field is restored to the database, but the original data is lost.</td>
</tr>
<tr>
<td>A resized field</td>
<td>The field resize is reversed. If the field has been increased, data is</td>
</tr>
<tr>
<td></td>
<td>truncated first to avoid errors.</td>
</tr>
<tr>
<td>A configured form</td>
<td>The form is reverted to its previous state.</td>
</tr>
<tr>
<td>A record is inserted</td>
<td>The record is deleted. (See Note below)</td>
</tr>
<tr>
<td>A record is deleted</td>
<td>The record is restored with its original data.</td>
</tr>
</tbody>
</table>

⚠️ **Warning:** Backing out an update set that belongs to an update set batch may affect other update sets in the batch. For more information, see Back out batched update set.

⚠️ **Note:** Regarding a record being deleted:
- If the sys_package is global, it is deleted.
- If the sys_package is not global, and it has a value, a warning displays that there is no deletion. Rather, the sys_update_xml is put into the default update set and the record is left in place.

### Procedure

1. Navigate to **System Update Sets > Retrieved Update Sets** or **System Update Sets > Local Update Sets**.

2. Open the update set record.

   ⚠️ **Note:** The currently selected application affects what options are available for the update set. Make sure you select the application, such as Global, that matches the contents of the update set.

3. Carefully review the contents of the update set and consider whether there will be problems if it is backed out.
   If backing out an update set is not sufficient or will cause issues, then, instead, create and commit a new update set to reverse the customizations.

4. Click **Back Out**.
A Progress page displays actions, progress, and problems. Problems are changes in more recent update sets that affect the update set that is being backed out. The backout preview process generates a warning for each problem.

5. Resolve each problem before proceeding with the back out.

- To keep the latest version, click **Decide to Keep Current**.
- To back out to the previous version, click **Decide to Use Previous**.

All changes are reversed as described in the table. The current update set tracks all of the new changes that occur.

The update set and all associated update records are deleted. If needed, you can still navigate to the retrieved update set, preview it, and commit it again.

**Note:** If you commit, back out, and then reapply a remote update set, errors appear in the previewer because backing out an update set creates delete updates in the current update set. The deletes are considered more recent changes and cause collisions.

### Cautions about deleting update sets

Deleting an update set is a bad practice. To revert a customization, back out the update set rather than deleting it.

Administrators can delete an update set only when it is not the current update set and it is empty (no `sys_update_xml` entries are associated with it). For example, after merging update sets, you might want to delete the original sets. This function is restricted by an access control rule (ACL) on the Update Set [sys_update_set] table.

ServiceNow strongly recommends that you do not delete `sys_update_xml` entries, because this action:

- Does not undo the updates.
- Removes any record of who applied the customizations.
- Removes the `sys_update_xml` entries associated with the update set, so customizations are overwritten when the instance is upgraded.

When you try to delete an update entry, a warning message appears. Click **OK** to confirm the deletion.

### Update set batching

Batch update sets enable you to group update sets together so you can preview and commit them in bulk.
Dealing with multiple update sets can lead to problems, including committing update sets in the wrong order or inadvertently leaving out one or more sets. You can avoid these problems by grouping completed update sets into a batch.

The system organizes update set batches into a hierarchy. One update set can act as the parent for multiple child update sets. A given set can be both a child and parent, enabling multiple-level hierarchies. One update set at the top level of the hierarchy acts as the base update set.

When you preview or commit the base update set, you preview or commit the entire batch. The system determines the processing order, and checks for collisions, based on the dates the changes were recorded, and on their sequential ancestry. Their ancestries are the specific instances in which the changes in the update sets took place.

Note: For more details, see Compare local update sets and View customizations and compare with current version.

Example of batched update sets
The list of update set records reflects the batch hierarchy in the Parent and Batch Base columns.

List of batched update sets

![List of batched update sets screenshot]

Diagram of batched update set hierarchy
Create a batch update set
You include an update set in a batch by specifying another update set as its parent.

Before you begin
Role required: admin
Adding a WIP update set to a completed batch resets the batch base to WIP.

Procedure
1. Navigate to System Update Sets > Local Update Sets.
2. Select the record for an update set that you want to include as a child in the batch.
3. On the Update Set record, navigate to the Parent field and select the update set to act as the parent.
4. Click Update.
   The system returns to the list of Update Sets. If the Parent column is visible, it shows the parent for the newly-created child.
Retrieve batched update sets
You retrieve a batch of update sets using the same process you would as for any individual update set.

Before you begin
Role required: admin

Procedure
To retrieve a batch of update sets, follow the same process for the batch base as you would for any individual update set. The system will process the entire batch at once. For details, see Retrieve an update set.

Preview a batch of update sets
You can preview at once all the update sets belonging to a batch.

Before you begin
Role required: admin
You must have retrieved the update sets from the source instance.

Procedure
1. Navigate to System Update Sets > Retrieved Update Sets
2. From the list of retrieved update sets, select the batch base for the batch you want to preview.
   You cannot separately preview an update set that is a child in a batch. You must preview the entire batch by previewing the batch base. If necessary, you can remove the child update set from the batch by editing its record’s parent field.
3. Click Preview Update Set Batch.
4. If the system found problems, click the Preview Problems for Batch and resolve the problems as you normally would for any update set. When you have resolved all the problems, click Run Preview Again for Batch.

Commit a batch of update sets
You can commit at once all the update sets belonging to a batch.

Before you begin
Role required: admin
Before committing, you must have previewed the update sets from the source instance and resolved any collisions.
Procedure
1. Navigate to **System Update Sets > Retrieved Update Sets**

2. From the list of retrieved update sets, select the batch base for the batch you want to preview.

   You cannot separately commit an update set that is a child in a batch. You must commit the entire batch by committing the batch base. If necessary, you can remove the child update set from the batch by editing its record’s **parent** field.

3. Click **Commit All Update Sets**.

**Reorganize a batch of update sets**

You can remove an individual update set from the batch or change its parent.

**Before you begin**

Role required: admin

Procedure
1. Navigate to **System Update Sets > Local Update Sets**.

2. Select the record for an update set that you want to move or remove as a child in the batch.

3. On the update set record, navigate to the **Parent** field and select the new update set to act as the parent. To remove the update set from the batch, delete any text from the **Parent** field and leave it blank.

4. Click **Update**.

   The system returns to the list of update sets. If the **Batch Base** column is visible, it shows the parent for the newly-created child.

**What to do next**

If the system property `glide.update_set.auto_preview` is set to **true**, the system automatically starts the preview process after the record is updated with a new parent. If this property is **false**, you must start the process manually. For more information on the preview process, see **Preview a batch of update sets**.

**Back out batched update set**

Back out a batched update set by following the backout procedure for the base update set for the batch. You can also back out any child update set independently.

The following rules apply when backing out an update set that belongs to a batch:
• If the update set has a parent value, the system clears the parent value and treats the update set as an independent update set, or as a new batch base if it has any children.

• The system backs out the selected update set, plus any children of the backed-out update set.

To learn more, see Back out an update set and Update set batching

**Example of backing out a batched update set**

If you back out Update Set 1.1 from the batch shown in List of batched update sets before backing out an update set., the result is the batch shown in List of batched update sets after backing out Update Set 1.1.

List of batched update sets before backing out an update set.
Hierarchical diagram of Update Set batch shows the hierarchy both before and after the back out. The red boxes show the update sets the system backs out if you back out Update Set 1.1.
Automated Test Framework (ATF)

The Automated Test Framework (ATF) enables you to create and run automated tests to confirm that your instance works after making a change. For example, after an upgrade, during application development, or when deploying instance configurations with update sets. Review failed test results to identify the changes that caused the failure and the changes that you should review.

ℹ️ **Note:** By default, the system property to run automated tests is disabled to prevent you from accidentally running them on a production system. Run tests only on development, test, and other non-production instances to avoid data corruption and outage.

For developer training, see Using the Automated Test Framework on the ServiceNow Developer Site.

**Benefits**
Automated Test Framework provides these benefits for change managers and developers.

- Reduce upgrade and development time by replacing manual testing with automated testing.
- Design tests once and reuse them in different contexts and with different test data sets.
- Keep test instances clean by rolling back test data and changes made after each test run.
- Create test suites to organize and run tests in batches.
- Schedule test suite runs.
- Enable non-technical test designers to create tests of standard Now Platform functionality.
- Reduce test design time by copying quick start tests and test suites.
- Create custom test steps to expand test coverage.

**Automated Test Framework records and components**
The Automated Test Framework consists of these records and components.

**Test**

A test is a logical grouping of related automated test steps that verify some functionality or feature. Each test is a record in the Test [sys_atf_test] table. Test designers typically create a test to verify one feature or a group of related features. For example, the **CSM: Create Product Case** test validates the creation of Product Case records. Each test has a related list of test steps and test results.
Test suite

A test suite is a collection of tests that run in a specific order. Test designers typically create a test suite to test an application or a group of related features. For example, the CSM: Case Management test suite validates the functionality of the Customer Service Management application. Test designers can schedule running test suites and starting any required client test runners.

Quick start test

A quick start test is a ServiceNow-provided test or test suite installed with the demo data of an application. Use quick start tests as templates to build your own tests and test suites.

Test step

A test step combines a step configuration with the runtime test data necessary to run a step. The test step always specifies the order in which it runs in the test. Test steps have their own related list of step results. Each test step is a record in the Test Step [sys_atf_step] table that specifies a test action, the step configuration, and an execution order. Test designers add test steps to tests to verify functionality. For example, the first test step of the CSM: Create Product Case test is to impersonate the demo user John Jason who is authorized for Case Management.

Step configuration

A step configuration is a specific test action the Automated Test Framework can run. Step configurations do not contain any runtime test data and can only be run when test designers add them as part of a test step. Each step configuration is a record in the Test Step Config [sys_atf_step_config] table that specifies the input variables used to run the step configuration and the output variables available to other step configurations. For example, the Impersonate step configuration allows a test to impersonate another user.

The Automated Test Framework provides a default list of step configurations for most use cases and allows test designers to create their own custom step configurations.

Step variable

A step variable stores step-specific input and output values. For example, the Open a New Form step configuration has variables to specify the table and form view names. Use step variables to specify a particular test step target or to pass information to other test steps.
Test result

A test result stores the output of a test or test suite run. Each test result is a record in the Test Result [sys_atf_test_result] table that specifies the duration of the test run, the status of the test, and screenshots where available. Use test results to identify failing or non-running tests, and use the test logs to see more information about test results. By default, the system deletes test and test suite results 30 days after creation unless you enable the option to retain the test result indefinitely.

Step result

A step result stores the output of a test step run. Each step result is a record in the Step Result [sys_atf_test_result_step] table that specifies the status of the test step, a summary of the output, and a complete log of the output generated by the test step. Use step results to identify failures and functionality needing review.

Assert type

The Assert type field specifies the conditions that must be met for a test to pass. Test designers can use assertions to specify whether the results of an operation are expected or unexpected. For example, suppose you want to test that a record cannot be updated. In this case, you would add a Record Insert test step and set the Assert type field to Record was not inserted. The test passes when the record insert fails.

Some test step categories that include steps with an Assert type field include:

- **Server category**: Assert which CRUD operations cause a test to pass or fail.
- **Custom UI category**: Assert which component states cause a test to pass or fail, and whether visible text causes a test to pass or fail.
- **Forms in Service Portal category**: Assert whether a form canceled in the browser due to validation errors, or a form successfully submitted to the server causes a test to pass.

Client test runners

A client test runner is a browser tab that runs client-side test steps within a Now Platform user interface. Client test runners require a browser tab to function. If no client test runner is available when you run a test, the system prompts you to open one. Testers can manually start a client test runner or select an existing client test runner. Test designers can schedule starting client test runners when they schedule running a test suite.

User roles

Assign roles to define Automated Test Framework permissions.
<table>
<thead>
<tr>
<th>Role Title [name]</th>
<th>Role Description</th>
</tr>
</thead>
</table>
| atf_test_admin   | Create or edit Automated Test Framework properties. Has permission to:  
|                  | • View the tests page  
|                  | • Create/edit/delete tests  
|                  | • Create/edit/delete test steps  
|                  | • View the step config page  
|                  | • View the test runner page  
|                  | • View the test suite results, test results and result items pages  
|                  | • Execute user tests  
|                  | • View, create, edit, delete and execute test suites  
|                  | • Create/edit step config records  
|                  | • Create/edit Automated Test Framework properties  
| atf_test_designer| View Automated Test Framework properties only (cannot create or edit properties). Has permission to:  
|                  | • View the tests page  
|                  | • Create/edit/delete tests  
|                  | • Create/edit/delete test steps  
|                  | • View the step config page  
|                  | • View the test runner page  
|                  | • View the test suite results, test results and result items pages  
|                  | • Execute user tests  
|                  | • View, create, edit, delete and execute test suites  
|                  | • View Automated Test Framework properties  
| atf_ws_designer  | View or set basic authentication profiles needed for REST endpoints that require authentication. See REST category for more information. |
Getting started with the Automated Test Framework

If you are new to the Automated Test Framework, read this overview to learn what the framework can do. Next, follow the tutorial to create and run a test that uses the most basic of ATF features. After you feel comfortable with the basics, explore more advanced features provided by the ATF.

ATF features provide flexibility in how you test your instance.

Test step configuration categories

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Catalog in Service Portal</td>
<td>Perform end-to-end testing for a catalog item in the Service Portal. • Open a record producer, catalog item, or order guide. • Set variable values and catalog item quantity. • Validate variable values, states, price, and items included in an order guide. • Navigate in an order guide. • Open and toggle catalog items in an order guide. • Add an item or an order guide to a shopping cart. • Order a catalog item or an order guide. • Submit a record producer.</td>
</tr>
<tr>
<td>Application Navigator</td>
<td>Create tests to check navigation features. • Verify that application menus are listed in the left navigation bar. • Verify that application modules are listed in the left navigation bar. • Navigate to a module as if a user clicked the module in the left navigation bar.</td>
</tr>
<tr>
<td>Category</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Custom UI</td>
<td>Create simple tests that mimic user actions with no scripting.</td>
</tr>
<tr>
<td></td>
<td>• Set component values.</td>
</tr>
<tr>
<td></td>
<td>• Assert that specified text is or is not on a page.</td>
</tr>
<tr>
<td></td>
<td>• Validate component values.</td>
</tr>
<tr>
<td></td>
<td>• Click components.</td>
</tr>
<tr>
<td></td>
<td>• Validate the states of components (read-only or not read-only).</td>
</tr>
<tr>
<td>Form</td>
<td>Create tests of forms.</td>
</tr>
<tr>
<td></td>
<td>• Open a new form or an existing record.</td>
</tr>
<tr>
<td></td>
<td>• Set field values.</td>
</tr>
<tr>
<td></td>
<td>• Validate field values or field states (such as mandatory, not mandatory, read only, not read only, visible, and not visible).</td>
</tr>
<tr>
<td></td>
<td>• Validate whether a UI action is visible.</td>
</tr>
<tr>
<td></td>
<td>• Click a button on a modal page.</td>
</tr>
<tr>
<td></td>
<td>• Click a UI action.</td>
</tr>
<tr>
<td></td>
<td>• Submit a form.</td>
</tr>
<tr>
<td>Service Catalog</td>
<td>Perform end-to-end testing for a catalog item.</td>
</tr>
<tr>
<td></td>
<td>• Open a catalog item or a record producer.</td>
</tr>
<tr>
<td></td>
<td>• Search for a catalog item.</td>
</tr>
<tr>
<td></td>
<td>• Set variable values and catalog item quantity.</td>
</tr>
<tr>
<td></td>
<td>• Validate variable values, states, and price.</td>
</tr>
<tr>
<td></td>
<td>• Add an item to a shopping cart.</td>
</tr>
<tr>
<td>Category</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>- Order a catalog item.</td>
</tr>
<tr>
<td></td>
<td>- Submit a record producer.</td>
</tr>
<tr>
<td>Forms in Service Portal</td>
<td>Create tests of forms in the Service Portal.</td>
</tr>
<tr>
<td></td>
<td>- Open a form.</td>
</tr>
<tr>
<td></td>
<td>- Set field values.</td>
</tr>
<tr>
<td></td>
<td>- Validate field values or field states (such as mandatory, not mandatory, read only, not read only, visible, and not visible).</td>
</tr>
<tr>
<td></td>
<td>- Validate whether a UI action is visible.</td>
</tr>
<tr>
<td></td>
<td>- Click a UI action.</td>
</tr>
<tr>
<td></td>
<td>- Submit a form.</td>
</tr>
<tr>
<td>REST</td>
<td>Create and send an Inbound REST request and verify the response.</td>
</tr>
<tr>
<td></td>
<td>- Test any REST endpoint on the instance.</td>
</tr>
<tr>
<td></td>
<td>- Use a REST request to create records, as well as retrieve, update, or delete records created in a previous test step or that already existed on the instance.</td>
</tr>
<tr>
<td></td>
<td>- Verify the response status code, response headers, response time, and response payload.</td>
</tr>
<tr>
<td>Server</td>
<td>Perform more complex operations, including the following:</td>
</tr>
<tr>
<td></td>
<td>- Perform unit tests using JavaScript, including tests using the Jasmine test framework.</td>
</tr>
<tr>
<td></td>
<td>- Test business rules, script includes, and other scripts.</td>
</tr>
<tr>
<td></td>
<td>- Create tests that operate on data that you define.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Output variables
Many test steps return output variables whose values you can use as inputs to a later step. For example, you can use output variables to accomplish the following tasks:

- Perform a server-side assert on a record that you previously inserted.
- Create a record as one user, and then reopen its form as a different user.

Custom test step configurations
In addition to the steps built into the Automated Test Framework, you can create custom test step configurations. These custom steps can take input variables and return output variables that you define.

ℹ️ Note: You can only define custom test steps that run on the server. The Automated Test Framework does not support creating custom step configurations that run on the browser.

Data preservation
The Automated Test Framework automatically tracks and deletes any data created by running tests, and automatically rolls back changes after testing.

Test suites
Test suites enable you to execute a batch of tests in a specified order. In addition, test suites can be hierarchical, with suites nested within other suites. You can associate test suites with schedules that determine when the system runs the test suites.

Build and run your first automated test
Follow these step-by-step instructions to create and run your first automated test. This test creates a new user record.

Before you begin

- The Automated Test Framework (com.glide.automated_testing_framework) plugin must be activated. It is activated by default on zboot or upgrade.
- If necessary, enable test execution. For instructions, see Enable or disable executing Automated Test Framework tests.

ℹ️ Note: By default, the system property that is used to run automated tests is disabled to prevent you from accidentally running these tests on a production system. To avoid data corruption or an outage, run tests only on development, test, and other non-production instances.

- Role required: admin

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Create new test
Create a new automated test record.

Procedure
1. Navigate to Automated Test Framework > Tests.
2. Click New.
3. On the Test new record form, enter a name for your test in the Name field. The system identifies this test by this name whenever it displays a list of tests (for example, under the Tests module).

Example

4. In the Description field, enter a description for your test.
5. Click Save.

Results
The system creates a new test record and returns to the list of tests. For more information about creating new automated tests, see Create a new automated test.

Add the first step to the new test
Add the first of three steps to the automated test.

About this task

Procedure
1. Navigate to Automated Test Framework > Tests.
2. Click the test that should contain the new test steps.
3. In the Test Steps related list at the bottom of the Test form, click Add Test Step.
4. In the middle column, click the row for the step type Open a new form, then click Next.
The system displays the Add Test Step form for the Open a new form step.

5. From the Table field, select the User [sys_user] table. Then click Submit.
The system creates the new step and returns to the test record.

6. Click **Update**.

**Results**
The system creates a new test record and returns to the list of tests. For more information about adding steps to automated tests, see [Add steps to an automated test](#).

**What to do next**
For some ideas on how to continue learning about the Automated Test Framework, see [Next steps with the Automated Test Framework](#).

**Add the second step to your automated test**
Add the second of three steps to the automated test.

**Procedure**

1. Click the test that should contain the new test steps.
2. In the Test Steps related list at the bottom of the Test form, click **Add Test Step**.
3. In the middle column, click **Set Field Values**, then click **Next**. The system displays the **Add Test Step** form for the **Set values** step.
4. In the field values section, set **Last name** to **Test** and **First name** to **Otto** (or other names of your choice).
5. Click **Submit**.

### Results

The system creates the new step and returns to the test record. For more information about adding steps to automated tests, see [Add steps to an automated test](#).

### Add the third step to your test

Add the last of three steps to the automated test.

### Procedure

1. Click the test that should contain the new test steps.
2. In the Test Steps related list at the bottom of the Test form, click **Add Test Step**.
3. In the middle column, click **Submit a Form**, then click **Next**. The system displays the **Add Test Step** form for the **Submit a form** step.
4. Leave all values as set by default and click **Submit**.

### Example
The system creates the new step and returns to the test record. The test record should now show the three steps you just added.

5. Click **Update**.
Results
The system returns to the list of tests. For more information about adding steps to automated tests, see Add steps to an automated test.

Run your test
After adding test steps, run and monitor the progress of the automated test.

Procedure
1. If necessary to view the Tests list, click Tests.
2. Click the row containing the test you just created.
   The system displays the Test form.
3. Click Run Test.
   Because this test includes a form step (any step involving a UI), the system displays a dialog box asking you to choose among any currently-running test clients or start a new test runner. To continue, select start a new test runner and click Run Test.

Results
The system displays the Run Test progress dialog. For more information about running automated tests, see Run an automated test.

Monitor test progress and view test results
Monitor the progress of the automated test and view its test results.

Procedure
1. Monitor the progress of the test in the Run Test progress dialog. If needed, you can cancel a test even while it’s running.

2. Note: If your test creates data, the system rolls back that data after all steps in the test complete.
   When complete, click Go to Results on the Run Test progress dialog to display the Test Results list, where you can view and analyze the results.

Next steps with the Automated Test Framework
After you feel comfortable creating and running simple tests, explore the more advanced features of the Automated Test Framework.

Batch tests together with test suites
If you commonly run a set of tests together, you can group them using test suites. To learn about test suites, see Building and running automated test suites.
Pass data from one test step to another with input and output variables

You can pass data from one test step to another using input variables and output variables. For more information, see Passing data from one automated test step to another.

Reuse common sequences of steps with templates

If you frequently add the same sequence of steps to your tests, save time by creating a template. To learn more about templates, see Add a predefined list of steps (template) to an automated test.

Domain separation and Automated Test Framework

Domain separation is supported in the Automated Test Framework. Domain separation enables you to separate data, processes, and administrative tasks into logical groupings called domains. You can control several aspects of this separation, including which users can see and access data.

Support level: Standard*

The support level is Standard but has some exceptions or special conditions.

- Includes Basic level
- Business logic: The service provider (SP) creates or modifies processes per customer. The use cases reflect proper use of the application by multiple SP customers in a single instance.
- The instance owner must be able to configure minimum viable product (MVP) business logic and data parameters. This configuration is done per tenant, as expected for the specific application.

Sample use case: An admin must be able to make comments required when a record closes for one tenant, but not for another.

For more information on support levels, see Application support for domain separation.

ATF use case

Automated test framework design and runtime access are solely for the owner of the instance to validate domain-specific processes. By designing tests for my customers we confirm results per tenant.

Related information

Domain separation for service providers
Headless Browser for Automated Test Framework

In the Rome release, ServiceNow® improved UI testing by automating the creation of browsers to process Automated Test Framework (ATF) User Interface (UI) tests. This feature is known as a headless browser, and helps you test UI functionality without having to manually open a browser to the Client Test Runner, which is what processes the UI tests in your local browser.

Background

ServiceNow customers use Automated Test Framework (ATF) to test applications and instances. When developing in the ServiceNow platform, your changes both have the desired behavior and don’t break existing features.

Since the ServiceNow Orlando release, customers can automate the testing and deployment of their applications via the Continuous Integration/Continuous Delivery (CI/CD) API.

Overview

The automation provided by the ServiceNow Headless Browser for ATF skips the step of manually opening a browser during testing. There are several sequential procedures to follow in the one-time setup. Below are the instructions for both Linux and Windows setup.

Headless browser setup for Linux

The ServiceNow® headless browser for Automated Test Framework provides automation so you can skip having to manually open a browser during testing. The headless browser setup is available in both Linux and Microsoft Windows. This topic covers the setup for Linux.

There are several sequential procedures to follow in the one-time setup. Below are the instructions for the Linux setup. There are several sequential procedures to follow in the one-time setup.

Prerequisites

Role required: admin on your ServiceNow® instance and local administrator on the host machine.

- Docker application installed
- Java keytool installed
- OpenSSL
• Two-way communication
  ◦ There must be two-way communication between the instance URL and your server.
  ◦ Find the IP address of your server and get your hostname. You can use one or both of them, but you need at least one.

⚠️ Note: If you don’t have a hostname and are connecting via the IP address, you can enter the IP address and put “localhost” in the Hostname environment variable.

💡 Tip: To make remembering these easier, set the following environment variables:
  ◦ export PASSWORD="<password to generate the certificates with>"
  ◦ export SERVERIP="<THIS SERVER’S IP ADDRESS>"
  ◦ export HOSTNAME="<HOSTNAME OF THIS SERVER>"

• Port
  ◦ Use Port 2376 or your own default port for this procedure. Make sure your firewall rules allow inbound requests on this port.

• To learn more, see Use TLS (HTTPS) to protect the Docker daemon socket.

Generate certificates for Headless Browser setup for Linux

Step 1 in the Linux setup for the ServiceNow® Headless Browser for Automated Test Framework: Generate TLS/SSL certificates to secure the Docker REST API and authenticate HTTP requests.

Before you begin
By default when exposing the Docker API, requests are not authenticated, which can leave your host machine vulnerable to attack. Docker API, however, does support TLS authentication where requests are verified against public private keys provided in the HTTPS encryption. In this step you create those keys for the server and the client.

Role required: admin on your ServiceNow® instance and local administrator on the host machine.

• Docker application installed
• Java keytool installed
• OpenSSL
About this task

- Two-way communication
  - There must be two-way communication between the instance URL and your server.
  - Find the IP address of your server and get your hostname. You can use one or both of them, but you need at least one.

⚠ Note: If you don’t have a hostname and are connecting via the IP address, you can enter the IP address and put "localhost" in the Hostname environment variable.

💡 Tip: To make remembering these easier, set the following environment variables:
  - export PASSWORD="<password to generate the certificates with>"
  - export SERVERIP="<THIS SERVER'S IP ADDRESS>"
  - export HOSTNAME="<HOSTNAME OF THIS SERVER>"

⚠️ Warning: It is recommended to get self-signed certificate authority keys from a trusted certificate authority.

- Port
  - Use Port 2376 or your own default port for this procedure.

⚠ Note: Make sure your firewall rules allow inbound requests on this port.

- To learn more, see Use TLS (HTTPS) to protect the Docker daemon socket.

Procedure

1. Open a command line.
2. Generate a self-signed certificate authority key or retrieve a keypair from a trusted certificate authority. To generate a self-signed certificate, see the example below; your configuration may vary:
   - openssl genrsa -aes256 -passout pass:$PASSWORD -out ca-key.pem 4096
   - openssl req -passin pass:$PASSWORD -new -x509 -days 365 -key ca-key.pem -sha256 -out ca.pem
   - chmod 0400 ca-key.pem
   - chmod 0444 ca.pem
3. Generate the server keypair using the certificate authority key. See the example below; your configuration may vary:
• openssl genrsa -out server-key.pem 4096
• openssl req -subj "/CN=$HOSTNAME" -new -key server-key.pem -out server.csr
• echo "subjectAltName = DNS:$HOSTNAME,IP:$SERVERIP,IP:127.0.0.1" > extfile.cnf
• openssl x509 -passin pass:$PASSWORD -req -days 365 -in server.csr -CA ca.pem -CAkey ca-key.pem -CAcreateserial -out server-cert.pem -extfile extfile.cnf

4. Create the client keypair using the certificate authority key. See the example below; your configuration may vary:
• openssl genrsa -out client-key.pem 4096
• openssl req -subj "/CN=$HOSTNAME" -new -key client-key.pem -out client.csr
• echo "extendedKeyUsage = clientAuth" > extfile.cnf
• openssl x509 -passin pass:$PASSWORD -req -days 365 -in client.csr -CA ca.pem -CAkey ca-key.pem -CAcreateserial -out client-cert.pem -extfile extfile.cnf

Now you’ve created all your encryption keys.

5. Import the CA Public Key and Client Keypair to a java keystore. See the example below; your configuration may vary:

Create the keystore file and create a password for it (and save for later use).

• keytool -genkey -keyalg RSA -alias dse -keystore my.keystore

Delete a default entry from the keystore file.

• keytool -delete -alias dse -keystore my.keystore

Import the CA public key to the keystore.

• keytool -import -keystore my.keystore -trustcacerts -alias ca -file ca.pem
• keytool -importkeystore -destkeystore my.keystore -srckeystore clientkeypair.p12 -srcstoretype pkcs12 -alias clientkeypair

Now that you have added all of the certificates to the keystore file, save this file for later, as it will be uploaded to the ServiceNow instance. In addition, be

Note: You are creating a new pkcs12 keystore file and importing the keypair to it. Then copy the contents to your original keystore file.
sure to remember the password that you entered when prompted to create the keystore file; you will need to enter that into a form in the ServiceNow instance.

**Configure Docker for Headless Browser setup in Linux**

Step 2 in the Linux setup for the ServiceNow® Headless Browser for Automated Test Framework: Configure Docker Server to authenticate all requests.

**Before you begin**

After creating your client and server keys as directed in Step 1, now you configure the Docker Server to authenticate all requests using those keys, and expose the Docker API on port 2376.

Role required: admin on your ServiceNow instance and local administrator on the host machine.

Complete step 1: Generate certificates for Headless Browser setup for Linux

**Procedure**

1. Configure Docker to use the certificates you generated in Step 1.
   
a. Find or create the `/etc/docker/daemon.json` file. You can run: `touch /etc/docker/daemon.json`

b. Edit the daemon.json file and add the following JSON values. Be sure to replace with the correct paths to your certificates:

   ```json
   {
     "debug": true,
     "tlscacert": "<path to your certificates>/ca.pem",
     "tlscert": "<path to your certificates>/server-cert.pem",
     "tlskey": "<path to your certificates>/server-key.pem",
     "tlsverify": true
   }
   
   To learn more, see https://docs.docker.com/config/daemon/#configure-the-docker-daemon.
   
2. Configure Docker to expose the remote API on a port (Port 2376 is recommended).
   
   You can configure Docker to accept remote connections with the docker.service systemd unit file for Linux distributions using **systemd**, such as recent versions of RedHat, CentOS, Ubuntu and SLES, or with the daemon.json file, which is recommended for Linux distributions that do not use systemd.

   If using **systemd (systemctl)**:
a. Use the command `sudo systemctl edit docker.service` to open an override file for `docker.service` in a text editor.

b. Add or modify the following lines, substituting your own values.

```
[Service]
ExecStart=
ExecStart=/usr/bin/dockerd -H tcp://0.0.0.0:2376
```

c. Save the file.

d. Reload the systemctl configuration.

```
sudo systemctl daemon-reload
```

e. Restart Docker

```
sudo systemctl restart docker.service
```

3. To enable Docker access for the current user via a command line, copy certificates to the user's Docker home directory:

- `mkdir -pv ~/.docker`
- `cp ca.pem ~/.docker`
- `cp client-key.pem ~/.docker/key.pem`
- `cp client-cert.pem ~/.docker/cert.pem`

4. Set `DOCKER_HOST` and `DOCKER_TLS_VERIFY` environment variables for your user:

```
export DOCKER_HOST=tcp://${SERVERIP}:2376
export DOCKER_TLS_VERIFY=1
```

source ~/.bash_profile

To learn more, see Manage Docker as a non-root user.

Create the Docker image and containers for Headless Browser setup in Linux

Step 3 in the Linux setup for the ServiceNow® Headless Browser for Automated Test Framework.

Before you begin
In this step you will pull the Docker image from the Public Registry.

Role required: admin on your ServiceNow instance and local administrator on the host machine.

Complete Step 2: Configure Docker for Headless Browser setup in Linux
Procedure

1. In the command line, run `docker ps` to verify that Docker is working. Your results should look like this:

![docker ps output]

2. Pull the Docker image from the DockerHub repo Docker ATF Headless Runner using this command: `docker pull docker.pkg.github.com/servicenow/atf-headless-runner/atf-headless-runner:<tagname>`

   **Note:** Make sure you find the correct tag name. To learn more, see Headless Browser image-instance version compatibility.

Add secrets to Docker for Headless Browser setup in Linux

Step 4 in the Linux setup for the ServiceNow® Headless Browser for Automated Test Framework. Docker Secrets is a feature for securely storing the passwords that will be used in containers. You will be creating a Docker secret, which stores the password of the ServiceNow user who will log into the instance to execute the tests.

**Before you begin**

Inside of the Docker container is an automation script that opens a web browser, logs into the instance, and opens the client test runner page. In order to log into the ServiceNow instance, you will need a user password. In this step you use a Docker feature called Docker Secrets so you can securely store passwords. Then when you run containers, the password is automatically available to log in to your instance.

Role required: admin on your ServiceNow instance and local administrator on the host machine.

Complete Step 3: Create the Docker image and containers for Headless Browser setup in Linux

**Procedure**

1. In the command line, enter `docker swarm init`

2. Enter `echo "<your users password>" | docker secret create my_secret_data -`

   **Note:** Replace `<your users password>` with the user's ServiceNow password.

   Your results should look like this:
3. The result is your secret ID, which you must save for later use. The secret ID will be added to the ServiceNow instance in the `sys_property sn_atf.headless.secret_id`

**Set up your instance for Headless Browser in Linux**

Step 5 in the Linux setup for the ServiceNow® Headless Browser for Automated Test Framework. Set up your instance so it can support the Headless Browser.

**Before you begin**

Now that you have configured the host machine, you will set up the instance so that it successfully communicates and authenticates with the host machine.

Role required: admin on your ServiceNow instance and local administrator on the host machine.

Complete Step 4: Add secrets to Docker for Headless Browser setup in Linux.

**Procedure**

1. Create a user - User ID, which can be whatever you want, as well as a browser name and password. The password should be the same as the one you created for your Docker Secret container.

2. Add the role `atf_test_designer` for this user. (To learn more, see the User Roles section in the Automated Test Framework (ATF) topic.)

3. Navigate to **System Definition > Certificates** to open the `sys_certificate` table. Create a new certificate (the name can be whatever you prefer):
   - **Type**: javakeystore
   - **Password**: Password for the keystore that you created in Step 2 (Add secrets to Docker for Headless Browser setup in Linux).

4. Upload the keystore file you saved earlier and attach it to this record.

5. Select **Submit**.

6. Click **Validate certificate**, which returns a success message.

7. Navigate to **System Security > Protocol Profiles** to open the `sys_protocol_profile` table. Create a new protocol profile record:
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol</td>
<td>One word &quot;myname&quot;</td>
</tr>
<tr>
<td>Default port</td>
<td>2376 (or the one you chose to use)</td>
</tr>
<tr>
<td>Keystore</td>
<td>Docker host keystore</td>
</tr>
</tbody>
</table>

8. Select **Submit**.

9. Create a Connections alias: Navigate to **Connections & Credentials > Connection & Credential Aliases** to open the **sys_alias** table.

10. Select the alias with the name **Docker**.

11. Under the Connections related list, select **New**.

12. Fill in these fields:
   - **Name**: Any text you prefer
   - **Credential field**: Leave blank

13. Select the **URL Builder** check box.

14. Select the **Mutual authentication** check box.

15. In the **Protocol profile** field, select the protocol profile you created earlier.

16. In the Host field, add the IP address or Host name of your server.

17. Select **Submit**.

   The Connection URL is automatically created by the system.

   **Note**: You only need to do this when using self-signed certificates. To learn more see **Generate certificates for Headless Browser setup for Linux**.

18. Modify properties:

   **Warning**: By default, the `com.glide.communications.trustmanager_trust_all` property is set to **false**. The Now Platform only trusts certificates that it can verify against the JVM certificate store. Self-signed and enterprise-signed certificates are not trusted. To learn more see the **Certificate trust** topic.

   - `com.glide.communications.httpclient.verify_hostname`: **false**
   - `com.glide.communications.trustmanager_trust_all`: **true**
Configure ATF for Headless Browser in Linux

Step 6 in the Linux setup for the ServiceNow® Headless Browser for Automated Test Framework.

Before you begin
Now that your connection and authentication are set up, you will configure ATF with several properties so that it can start containers successfully on the host machine.

Role required: admin on your ServiceNow instance and local administrator on the host machine.

Complete Step 5: Set up your instance for Headless Browser in Linux.

Procedure

1. In your instance, navigate to **ATF > Administration > Properties**.

2. Enable the top two properties: **Enable test/test suite execution** and **Enable scheduled test suite execution**.

3. Scroll down to the **Headless Runner Properties** section.

4. Enable the top check box.

5. Enter the following values in the Headless Browser Properties form:
### Form label

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Value to Imput</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_atf.headless.username</td>
<td>Username of the integration user that you created in Add secrets to Docker for Headless Browser setup in Linux</td>
</tr>
<tr>
<td>sn_atf.headless.secret_id</td>
<td>Docker “Secret ID” that you obtained earlier, as well as the Docker secret name that you created in Step 2 of Add secrets to Docker for Headless Browser setup in Linux.</td>
</tr>
<tr>
<td>sn_atf.headless.docker_image_name</td>
<td>Image name, with tag that you downloaded: servicenowdevx/atf-headless-runner:tagname</td>
</tr>
</tbody>
</table>

**Note:** If you forgot the secret ID, go to the Docker host and run the command `docker secret list`.

6. Leave the rest of the fields as they are.

7. Select **Save**.

To verify:

a. Go to **ATF Schedules** > and next to the Suite Schedules link select **New**.

b. In the Run box select **On Demand**.

c. Select **Save**.

d. In the Scheduled Suite related list, add a new record for your test suite by double-clicking the + sign and choose your browser name of **Chrome** or **Firefox**. Use **Child A** as a good example to run.
e. OS name: Linux  
f. Select **Execute Now**.  
g. Verify that your test passed.  
   When your verification is successful, any suites with UI tests run by scheduled suites or via CICD now automatically create headless test runners without the need to manually open the "scheduled client test runner" page.

**Headless Browser setup for Microsoft Windows**

The ServiceNow® Headless Browser for Automated Test Framework provides automation so you can skip having to manually open a browser during testing. The Headless Browser setup is available in both Linux and Microsoft Windows. This topic covers the setup for Windows.

There are several sequential procedures to follow in the one-time setup. Below are the instructions for the Microsoft Windows setup.

**Prerequisites**

Role required: admin on your ServiceNow instance and local administrator on the host machine.

⚠️ **Warning:** The only ServiceNow-supported version of Microsoft Windows as a host is: Windows Server 2019 v10.0.17763.737. No other versions are supported. If you are unable to meet these requirements, a Linux host is recommended.

- Make sure that the following programs are installed on your Windows server:
  - Install Docker for Headless Browser setup for Windows
  - Java keytool Chocolatey tool for javaruntime
  - OpenSSL Chocolatey tool for openssl
• Two-way communication
  ◦ There must be two-way communication between the instance URL and your server.
  ◦ Find the IP address of your server and get your hostname. You can use one or both of them, but you need at least one.

  **Note:** If you don’t have a hostname and are connecting via the IP address, you can enter the IP address and put "localhost" in the Hostname environment variable.

  **Tip:** To make remembering these easier, set the following environment variables:
  ◦ set PASSWORD=<password>
  ◦ set SERVERIP=<server ip>
  ◦ set HOSTNAME=<hostname>

• Port
  ◦ Use Port 2376 or your own default port for this procedure. Make sure your firewall rules allow inbound requests on this port.

• To learn more, see Use TLS (HTTPS) to protect the Docker daemon socket.

Install Docker for Headless Browser setup for Windows

Before you begin setting up your ServiceNow® Headless Browser for Automated Test Framework for Microsoft Windows, you must install Docker.

**Before you begin**
Role required: admin
Install Docker for Windows before you begin Step 1 of the Headless Browser for Windows setup.

**Warning:** The only ServiceNow-supported version of Microsoft Windows as a host is: Windows Server 2019 v10.0.17763.737. No other versions are supported. If you are unable to meet these requirements, a Linux host is recommended.

**Procedure**

1. Open an elevated PowerShell session and install the Docker-Microsoft PackageManagement Provider from the PowerShell Gallery.

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Install-Module -Name DockerMsftProvider -Repository PSGallery -Force
If you're prompted to install the NuGet provider, type Y to install it as well.

2. Use the PackageManagement PowerShell module to install the latest version of Docker.
PowerShell
Install-Package -Name docker -ProviderName DockerMsftProvider
When PowerShell asks you whether to trust the package source 'DockerDefault', type A to continue the installation.

3. After the installation completes, restart the computer.
PowerShell
Restart-Computer -Force

Generate certificates for Headless Browser setup for Windows
Step 1 in the Microsoft Windows setup for the ServiceNow® Headless Browser for Automated Test Framework: Generate TLS/SSL certificates to secure the Docker REST API and authenticate HTTP requests.

Before you begin
By default when exposing the Docker API, requests are not authenticated, which can leave your host machine vulnerable to attack. Docker API, however, does support TLS authentication where requests are verified against public private keys provided in the HTTPS encryption. In this step you create those keys for the server and the client.

Role required: admin on your ServiceNow instance and local administrator on the host machine.

⚠️ Warning: The only ServiceNow-supported version of Microsoft Windows as a host is: Windows Server 2019 v10.0.17763.737. No other versions are supported. If you are unable to meet these requirements, a Linux host is recommended.

- Make sure that the following programs are installed on your Windows server:
  - Docker application for Windows
  - Java keytool Chocolatey tool for javaruntime
  - OpenSSL Chocolatey tool for openssl
About this task

• Two-way communication
  ◦ There must be two-way communication between the instance URL and your server.
  ◦ Find the IP address of your server and get your hostname. You can use one or both of them, but you need at least one.

⚠ Note: If you don’t have a hostname and are connecting via the IP address, you can enter the IP address and put "localhost" in the Hostname environment variable.

⚠ Tip: To make remembering these easier, set the following environment variables:
  ◦ set PASSWORD=<password>
  ◦ set SERVERIP=<server ip>
  ◦ set HOSTNAME=<hostname>

⚠ Warning: It is recommended to get self-signed certificate authority keys from a trusted certificate authority.

• Port
  ◦ Use Port 2376 or your own default port for this procedure.

⚠ Note: Make sure your firewall rules allow inbound requests on this port.

• To learn more, see Use TLS (HTTPS) to protect the Docker daemon socket.

Procedure

1. Open a command line.

2. Generate a self-signed certificate authority key or retrieve a keypair from a trusted certificate authority. To generate a self-signed certificate, see the example below; your configuration may vary:

   • openssl genrsa -aes256 -passout pass:%PASSWORD% -out ca-key.pem 4096
   • openssl req -passin pass:%PASSWORD% -new -x509 -days 365 -key ca-key.pem -sha256 -out ca.pem

3. Generate the server keypair using the certificate authority key. See the example below; your configuration may vary:

   • openssl genrsa -out server-key.pem 4096
   • openssl req -subj /CN=%HOSTNAME% -new -key server-key.pem -out server.csr
4. Create the client keypair using the certificate authority key. See the example below; your configuration may vary:
• openssl genrsa -out client-key.pem 4096
• openssl req -subj /CN=%HOSTNAME% -new -key client-key.pem -out client.csr
• echo extendedKeyUsage = clientAuth > extfile.cnf
• openssl x509 -passin pass:%PASSWORD% -req -days 365 -in client.csr -CA ca.pem -CAkey ca-key.pem -CAcreateserial -out client-cert.pem -extfile extfile.cnf

Now you've created all your encryption keys.

5. Import the CA Public Key and Client Keypair to a java keystore. See the example below; your configuration may vary.
Create the keystore file and create a password for it (and save for later use).
• keytool -genkey -keyalg RSA -alias dse -keystore my.keystore
Delete a default entry from the keystore file.
• keytool -delete -alias dse -keystore my.keystore

ℹ️ Note: This entry is auto-generated, so is not needed.
Import the CA public key to the keystore
• keytool -import -keystore my.keystore -trustcacerts -alias ca -file ca.pem
Import the client keypair

ℹ️ Note: You are creating a new pkcs12 keystore file and importing the keypair to it. Then copy the contents to your original keystore file.
• openssl pkcs12 -export -name clientkeypair -in client-cert.pem -inkey client-key.pem -out clientkeypair.p12
• keytool -importkeystore -destkeystore my.keystore -srckeystore clientkeypair.p12 -srcstoretype pkcs12 -alias clientkeypair

Now that you have added all of the certificates to the keystore file, save this file for later, as it will be uploaded to the ServiceNow instance. In addition, be
sure to remember the password that you entered when prompted to create the keystore file; you will need to enter that into a form in the ServiceNow instance.

**Configure Docker for Headless Browser setup in Windows**

Step 2 in the Microsoft Windows setup for the ServiceNow® Headless Browser for Automated Test Framework: Configure Docker Server to authenticate all requests.

**Before you begin**

After creating your client and server keys as directed in Step 1, now you configure the Docker Server to authenticate all requests using those keys, and expose the Docker API on port 2376.

Role required: admin on your ServiceNow instance and local administrator on the host machine.

Complete Step 1: Generate certificates for Headless Browser setup for Windows

**Procedure**

1. Configure Docker to use the certificates you generated in Step 1.
2. Find or create the C:\ProgramData\docker\config\daemon.json file.
3. Add the following properties to the daemon.json file. Be sure to replace with the correct paths to your certificates:

   ```json
   {
       "tlscacert": "C:\Users\Administrator\certs\ca.pem",
       "tlscert": "C:\Users\Administrator\certs\server-cert.pem",
       "tlskey": "C:\Users\Administrator\certs\server-key.pem",
       "tlsverify": true,
       "hosts": ["tcp://0.0.0.0:2376", "npipe://"]
   }
   ``

   To learn more, see [https://docs.docker.com/config/daemon/#configure-the-docker-daemon](https://docs.docker.com/config/daemon/#configure-the-docker-daemon).

4. In administrator PowerShell, run: `#restart-service *docker*`
Create the Docker image and containers for Headless Browser setup in Microsoft Windows

Step 3 in the Microsoft Windows setup for the ServiceNow® Headless Browser for Automated Test Framework: Pull the Docker image from the Public Registry.

Before you begin
In this step you will pull the Docker image from the Public Registry.
Role required: admin on your ServiceNow instance and local administrator on the host machine.
Complete Step 2: Configure Docker for Headless Browser setup in Windows

Procedure
1. In an administrator command line, run `docker ps` to verify that Docker is working.
   Your results should look like this:

2. Pull the Docker image from the DockerHub repo Docker ATF Headless Runner using this command: `ghcr.io/servicenow/atf-headless-runner:<tagname>`

   Note: Make sure you find the correct tag name. To learn more, see Headless Browser image-instance version compatibility.

Add secrets to Docker for Headless Browser setup in Microsoft Windows

Step 4 in the Microsoft Windows setup for the ServiceNow® Headless Browser for Automated Test Framework. Docker Secrets is a feature for securely storing the passwords that will be used in containers. You will be creating a Docker secret that stores the password of the ServiceNow user who will log into the instance to execute the tests.

Before you begin
Inside of the Docker container is an automation script that opens a web browser, logs into the instance, and opens the Client Test Runner page. In order to log into the ServiceNow instance, you will need a user password. In this step you use a Docker feature called Docker Secrets so you can securely store passwords. Then when you run containers, the password is automatically available to log in to your instance.

Role required: Go to your ServiceNow instance and create a new user to be used by the Docker container to log in. Give this user the roles of admin
or `atf_test_admin`. Save the password to be used in Step 2 of the following procedure.

Complete Step 3: Create the Docker image and containers for Headless Browser setup in Microsoft Windows

**Procedure**

1. In the admin command line, enter `docker swarm init`
2. In the admin command line, enter `echo <your users password> | docker secret create sn_password`

   **Note:** Replace `<your users password>` with the user's ServiceNow password.

   Your results should look like this:

   ![Command Line Output](image)

3. The result is your secret ID, which you must save for later use. The secret ID will be added to the ServiceNow instance in the `sys_property sn_atf.headless.secret_id`

**Set up instance for Headless Browser in Microsoft Windows**

Step 5 in the Microsoft Windows setup for the ServiceNow® Headless Browser for Automated Test Framework: Set up your instance so it can support the Headless Browser.

**Before you begin**

Now that you have configured the host machine, you will set up the instance so that it successfully communicates and authenticates with the host machine.

Role required: admin on your ServiceNow instance and local administrator on the host machine.

Complete Step 4: Add secrets to Docker for Headless Browser setup in Microsoft Windows.
Procedure

1. Create a user - User ID, which can be whatever you want, as well as a user name and password. The password should be the same as the one you created for your Docker Secret container.

2. Add the role `atf_test_designer` for this user. (To learn more, see the User Roles section in the Automated Test Framework (ATF) topic.

3. Navigate to **System Definition > Certificates** to open the `sys_certificate` table. Create a new certificate (the name can be whatever you prefer):
   - **Type**: `javakeystore`
   - **Password**: Password for the keystore that you created in Step 2 (*Add secrets to Docker for Headless Browser setup in Microsoft Windows*).

4. Upload the keystore file you saved earlier and attach it to this record.

5. Select **Submit**.

6. Click **Validate certificate**, which returns a success message.

7. Navigate to **System Security > Protocol Profiles** to open the `sys_protocol_profile` table. Create a new protocol profile record:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol</td>
<td>One word “myname”</td>
</tr>
<tr>
<td>Default port</td>
<td>2376 (or the one you chose to use)</td>
</tr>
<tr>
<td>Keystore</td>
<td>Docker host keystore</td>
</tr>
</tbody>
</table>

8. Select **Submit**.

9. Create a Connections alias: Navigate to **Connections & Credentials > Connection & Credential Aliases** to open the `sys_alias` table.

10. Select the alias with the name `Docker`.

11. Under the Connections related list, select **New**.

12. Fill in these fields:
   - **Name**: Any text you prefer
   - **Credential field**: Leave blank

13. Select the **URL Builder** check box.

14. Select the **Mutual authentication** check box.

15. In the **Protocol profile** field, select the protocol profile you created earlier.

16. In the Host field, add the IP address or Host name of your server.
17. Select **Submit**.
   The Connection URL is automatically created by the system.

18. Modify properties:

   **Warning:** By default, the
   `com.glide.communications.trustmanager_trust_all` property is set to `false`.
   The Now Platform only trusts certificates that it can verify against the JVM
   certificate store. Self-signed and enterprise-signed certificates are not
   trusted. To learn more see the Certificate trust topic.

   • `com.glide.communications.httpclient.verify_hostname`: false
   • `com.glide.communications.trustmanager_trust_all`: true

Configure Automated Test Framework (ATF) for Headless Browser in Microsoft Windows

Step 6 in the Microsoft Windows setup for the ServiceNow® Headless Browser for
Automated Test Framework (ATF): Configure ATF with properties.

**Before you begin**
Now that your connection and authentication are set up, you will configure ATF
with several properties so that it can start containers successfully on the host
machine.

Role required: admin on your ServiceNow instance and local administrator on
the host machine.

Complete Step 5: Set up instance for Headless Browser in Microsoft Windows.

**Procedure**
1. In your instance, navigate to **ATF > Administration > Properties**.
2. Enable the top two properties: **Enable test/test suite execution** and **Enable
   scheduled test suite execution**.
3. Scroll down to the **Headless Runner Properties** section.

4. Enable the top check box.

5. Enter the following values in the Headless Runner Properties form:

<table>
<thead>
<tr>
<th>Form Label</th>
<th>Property Name</th>
<th>Value to Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>The user account used to login from the headless Client Test Runner and begin the tests</td>
<td>sn_atf.headless.username</td>
<td>Username of the integration user that you created in Add secrets to Docker for Headless Browser setup in Microsoft Windows</td>
</tr>
<tr>
<td>The Docker secret ID that has the password stored of the user account</td>
<td>sn_atf.headless.secret_id</td>
<td>Docker “secret ID” that you obtained earlier, as well as the Docker secret name that you created in Step 2 of the Add secrets to Docker for Headless Browser setup in Microsoft Windows procedure.</td>
</tr>
</tbody>
</table>
### Form Label | Property Name | Value to Imput
---|---|---
The Docker image that is used for headless Client Test Runner | sn_atf.headless.docker_image_name | Image name with tag that you downloaded: ghcr.io/servicenow/atf-headless-runner:<tagname>

**Note:** If you forgot the secret ID, go to Windows host and run the command docker secret list.

6. **At the The absolute path of the secret file on a docker container** field, enter: C:\ProgramData\Docker\secrets\<YOUR_SECRET_NAME>

7. Leave the rest of the fields as they are.

8. Select Save.
   To verify:
   a. Go to **ATF Schedules** and next to the Suite Schedules link select **New**.
   b. In the Run box select **On Demand**.
   c. Select **Save**.
   d. In the Scheduled Suite related list, add a new record for your test suite by double-clicking the + sign and choose your browser name of **Chrome** or **Firefox**. Use **Child A** as a good example to run.

![Scheduled Suites](image)

   e. **OS name:** Windows
   f. Select **Execute Now**.
   g. Verify that your test passed.

   When your verification is successful, any suites with UI tests run by scheduled suites or via CICD now automatically create headless test runners without the need to manually open the “scheduled client test runner” page.
Headless Browser image-instance version compatibility

The instance-to-image compatibility makes sure that the automation script inside the Docker image is compatible with the instance code. Elements such as the user interface might change over time to support new features or upgrades.

Image-instance version compatibility

Below is a list of the image tags to install based on the ServiceNow instance version that you have.

<table>
<thead>
<tr>
<th>Docker image tag</th>
<th>Compatible instance versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>lin-1.0.0</td>
<td>Rome +</td>
</tr>
<tr>
<td>win-1.0.0</td>
<td>Rome +</td>
</tr>
</tbody>
</table>

ServiceNow® Headless Browser system properties

Below is a table of the properties you must have as you set up the ServiceNow® Headless Browser for Automated Test Framework.

<table>
<thead>
<tr>
<th>Property properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Property name</strong></td>
</tr>
<tr>
<td>sn_atf.headless.browser_options</td>
</tr>
<tr>
<td>sn_atf.headless.default_browser</td>
</tr>
<tr>
<td>sn_atf.headless.default_os</td>
</tr>
<tr>
<td>sn_atf.headless.docker_image_name</td>
</tr>
<tr>
<td>sn_atf.headless.docker_window_seconds</td>
</tr>
</tbody>
</table>
System properties (continued)

<table>
<thead>
<tr>
<th>Property name</th>
<th>Type</th>
<th>Default value</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_atf.headless.enabled</td>
<td>boolean</td>
<td>false</td>
<td>Determines whether the instance attempts to create headless client test runners for scheduled UI test runs</td>
</tr>
<tr>
<td>sn_atf.headless.heartbeat_enabled</td>
<td>boolean</td>
<td>true</td>
<td>When property is true, Docker container sends a REST API request every minute to check that the sys_atf_agent is still online, and stops the container if the record status is &quot;offline&quot; or the record no longer exists.</td>
</tr>
<tr>
<td>sn_atf.headless.heartbeat_uri</td>
<td>string</td>
<td>/api/now/atf_agent/online</td>
<td>The URL of the heartbeat endpoint so container can verify browser is still responsive</td>
</tr>
<tr>
<td>sn_atf.headless.images_check.enabled</td>
<td>boolean</td>
<td>false</td>
<td>When this property is true, the instance verifies that the requested Docker image:tag is present on the</td>
</tr>
</tbody>
</table>
### System properties (continued)

<table>
<thead>
<tr>
<th>Property name</th>
<th>Type</th>
<th>Default value</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>host before test execution</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sn_atf.headless.login_button_id</td>
<td>string</td>
<td>sysverb_login</td>
<td>The HTML ID of the login page Submit/Login button</td>
</tr>
<tr>
<td>sn_atf.headless.login_page</td>
<td>string</td>
<td>login.do</td>
<td>URL of login page that the browser navigates to</td>
</tr>
<tr>
<td>sn_atf.headless.password_field_id</td>
<td>string</td>
<td>user_password</td>
<td>The HTML ID of the input field for the password on the Login page</td>
</tr>
<tr>
<td>sn_atf.headless.request_timeout_sec</td>
<td>int</td>
<td>200</td>
<td>Number of seconds that HTTP requests being sent to the Docker host have until timeout</td>
</tr>
<tr>
<td>sn_atf.headless.retry_count</td>
<td>int</td>
<td>10</td>
<td>Number of times the instance checks for agent coming online before cancelling the test run</td>
</tr>
<tr>
<td>sn_atf.headless.runner_banner_id</td>
<td>string</td>
<td>test_runner_banner</td>
<td>The ID of the element verifying that client test runner page loaded correctly</td>
</tr>
<tr>
<td>sn_atf.headless.runner_url</td>
<td>string</td>
<td>atf_test_runner.do?sysparm_nostack=true&amp;sysparm_scheduled_tests_only=true&amp;sysparm_headless=true</td>
<td>URL of the test runner page and its query parameters</td>
</tr>
</tbody>
</table>
### System properties (continued)

<table>
<thead>
<tr>
<th>Property name</th>
<th>Type</th>
<th>Default value</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_atf.headless.secret_gid</td>
<td>string</td>
<td>1000</td>
<td>The GID of the Docker container default user</td>
</tr>
<tr>
<td>sn_atf.headless.secret_id</td>
<td>string</td>
<td></td>
<td>The ID of the Docker secret on host machine</td>
</tr>
<tr>
<td>sn_atf.headless.secret_name</td>
<td>string</td>
<td></td>
<td>Name of the Docker secret on host machine</td>
</tr>
<tr>
<td>sn_atf.headless.secret_path</td>
<td>string</td>
<td>/run/secrets/&lt;secret_name&gt;</td>
<td>Path where Docker secret file exists: (Learn more: <a href="https://docs.docker.com/engine/swarm/secrets/#how-docker-manages-secrets">https://docs.docker.com/engine/swarm/secrets/#how-docker-manages-secrets</a>)</td>
</tr>
<tr>
<td>sn_atf.headless.secret_uid</td>
<td>string</td>
<td>1000</td>
<td>The UID of the Docker container default user</td>
</tr>
<tr>
<td>sn_atf.headless.service_clean_exclude_list</td>
<td>string</td>
<td></td>
<td>Service ID exceptions that should NOT be deleted during the instance service cleanup job. This job runs every night and deletes any services that are on the host and past their expiration time.</td>
</tr>
<tr>
<td>sn_atf.headless.service_stop_deletes</td>
<td>boolean</td>
<td>false</td>
<td>On completing of a test execution, if this property is</td>
</tr>
<tr>
<td>Property name</td>
<td>Type</td>
<td>Default value</td>
<td>Purpose</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------</td>
<td>---------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sn_atf.headless.timeout_mins</td>
<td>int</td>
<td>1440</td>
<td>Number of minutes before Docker service automatically shuts down</td>
</tr>
<tr>
<td>sn_atf.headless.username</td>
<td>string</td>
<td>user_name</td>
<td>The HTML ID of the input field for the username on the Login page</td>
</tr>
<tr>
<td>sn_atf.headless.user_field_id</td>
<td>string</td>
<td></td>
<td>Username of user who logs in to the instance</td>
</tr>
<tr>
<td>sn_atf.headless.validation_id</td>
<td>string</td>
<td></td>
<td>The ID of the element confirming verification page has loaded</td>
</tr>
<tr>
<td>sn_atf.headless.validation_page</td>
<td>string</td>
<td>atf_headless_validation_page</td>
<td>The URL of the validation page on the instance</td>
</tr>
<tr>
<td>sn_atf.headless.vp_has_role_id</td>
<td>string</td>
<td></td>
<td>The ID of the element verifying that the user has correct roles</td>
</tr>
<tr>
<td>sn_atf.headless.vp_success_id</td>
<td>string</td>
<td></td>
<td>The ID of the element signifying</td>
</tr>
</tbody>
</table>
System properties (continued)

<table>
<thead>
<tr>
<th>Property name</th>
<th>Type</th>
<th>Default value</th>
<th>Purpose</th>
</tr>
</thead>
</table>

Related information

Headless Browser for Automated Test Framework
Automated Test Framework (ATF)

Headless Browser troubleshooting

These tips can help you troubleshoot your Linux or Microsoft Windows setup of the ServiceNow® Headless Browser for Automated Test Framework.

There are three basic areas to examine when troubleshooting your Headless Browser setup.

Docker container

Error logs: When the headless test completes, if there are errors they are generated in the Docker container. Whether the operation fails or succeeds, the container’s stdout/stderr logs are placed in the sn_atf_docker_service table.

Headless client test runner did not start in the time allotted message: This message generally means an error occurred in the Docker container while it was initializing or starting up. This may indicate something is incorrect in your Docker container setup. Navigate to the sn_atf_docker_service table to read the logs and see the error message.

Instance

Error caught running Docker flow when starting the ATF tests messages: Follow the URL address to find the flow context, which contains the error logs. For example, the above error message can display when the instance can’t access the Docker host.

Network errors

Firewalls: Make sure your firewalls are set up so that the instance can access the host and port and the server can access the instance.
Building and running automated tests with the Automated Test Framework

Basic tasks in the Automated Test Framework.

To build and run any test, you always perform certain operations:

- Create a new automated test
- Add steps to an automated test
- Run an automated test
• View the progress of automated tests
• View test results

You perform other operations often, but not necessarily for every test.
• For tests involving form steps, you might need to View results screenshots from an automated test.
• For some tests, you might need to know about Passing data from one automated test step to another.
• Some steps frequently occur in the same sequence in many different tests, so you can Add a predefined list of steps (template) to an automated test.

Create a new automated test
Create a named automated test containing a series of steps to execute.

Before you begin
Role required: atf_test_admin or atf_test_designer

Procedure
1. Navigate to Automated Test Framework > Tests.
2. Click New.
3. On the Test new record form, enter a Name for your test. The system identifies this test by this name wherever it displays a list of tests (for example, in the Tests module).
4. Optional: Enable parameterized testing to run a test multiple times with different test data for each run. For more information, see Parameterized tests.
5. Enter a Description for your test.
6. Click Save.
   The system creates a new test record and returns to the list of tests.

What to do next
Add steps to an automated test.

Add a predefined list of steps (template) to an automated test
With test templates you can add a predefined list of steps to a test. Any list of steps that follows a set pattern makes a good candidate for a template.
Before you begin
You must have created the test to which you want to add steps.
Role required: atf_test_admin or atf_test_designer

About this task
Many tests follow similar patterns. One common pattern, for example, is to open a form, set some field values, validate some field values, click a UI action, and submit the current form. If a template exists containing these steps, you can add them to a test all at once. The Automated Test Framework comes with default templates. You can also create custom test templates.

Procedure
1. Navigate to Automated Test Framework > Tests.
2. Click the row for the test to which you want to add steps.
   The system displays the Test form.
3. On the Test Steps related list, click Add Test Template.
   The system displays the Add a test template dialog.
4. From the Table field, select the table you want to test with these steps.
5. From the Template field, select the template containing the steps you want to add.
6. Click Add.
   The system adds the template steps to the test. It also adds to the test description a set of instructions on how to complete the test from this template.
7. Following the instructions in the test description, edit each step added by the template to include the necessary information.

What to do next
Proceed to edit or save the test as you normally would.

Related information
Create an automated test steps template

Add steps to an automated test
Create a series of steps for an automated test to run in a specified order.

Before you begin
You must create a test before you can add steps to the test.
Role required: atf_test_admin or atf_test_designer
About this task
This procedure lets you add one step at a time. You can also add a batch of steps at once. For more information, see Add a predefined list of steps (template) to an automated test.

Procedure
1. Navigate to Automated Test Framework > Tests.
2. Click the test that should contain the new test steps.
3. In the Test Steps related list at the bottom of the Test form, click Add Test Step.
4. From the left panel of the Add Test Step form, select the test step category, or select All Steps to view all available test steps for all test categories. For example, if selecting a specific test step category, select Form for form-related test steps, Application Navigator to test application menu or module visibility, or Server for tests executed on the server.

   ☢️ Note: When selecting certain types of server tests, you can specify whether a positive or negative test result constitutes a successful test outcome. For more information, see Assert type in Automated Test Framework (ATF).

5. Click the type of test step you want to select. For example, if you select the Form category, select Set Field Values to set the field values on a form.

   Start any sequence of steps that work with forms with the Open a new form or Open an existing form step. Close with the Submit form step.

6. If applicable, from the Insert after drop-down list, select the step that you want to precede this step. If this is the first step in a test, the Insert after drop-down list does not appear.

7. Click Next.

8. From the Table list on the Add Test Step form, select the table that you want to test in this step.

9. Optional: In the Execution Order field, enter an integer representing the order in which you want the test to execute this step. For more information on Execution Order, see Edit automated test step order.

10. Fill in the fields that apply to this step. For instructions, see Test step categories. Some steps return output values that you can pass to the inputs for a subsequent step. For more information, see Pass values from one automated test step to another.

11. Click Submit.
The system creates the step and displays the test record.

12. Repeat Steps 3 through 11 to add additional steps for this test.

What to do next
Run an automated test.

Related information
Automated Test Framework use case examples

Change automated test step
If necessary, edit a test step after you create it.

Before you begin
Role required: atf_test_admin or atf_test_designer

Procedure
1. Navigate to Automated Test Framework > Tests.
2. Click the row containing the test you want to edit.
3. On the Test Steps related list, click the row containing the step you want to edit.
   The system displays the Set Field Values form.
4. Edit the fields you want to change.
5. Click Update.

Edit automated test step order
By default, steps execute in the order in which you created them. You can change this order by editing the Execution Order field.

Before you begin
Role required: atf_test_admin or atf_test_designer

About this task
By default, the system assigns the value 1 to Execution Order for the first step created for a test. When you add a step, the system assigns it the next-highest available integer value. In other words, if the highest Execution Order for any step in the test has the value 7, the system assigns 8 to the new step. By editing these values, you can change the order in which the test executes the steps.
**Procedure**

1. Navigate to **Automated Test Framework > Tests**.
2. Click the row containing the test you want to edit. The system displays the **Test** form.
3. In the **Test Steps** related list, edit the values in the **Execution order** column to determine the new order for the steps.
4. Click **Update**.

**Copy automated test**

Copy an existing test, which you can then re-name and modify.

**Before you begin**

Role required: atf_test_admin or atf_test_designer

**Procedure**

1. Navigate to **Automated Test Framework > Tests**.
2. Click the row containing the test you want to copy. The Test record form opens.
3. Near the top-right of the Test record form, click **Copy Test**. An annotation displays confirming that the system has copied the test. After you dismiss this annotation, the system displays a new test record identical to the copied record, with the exception of the **Name**.
4. In the **Name** field, enter the name you want to assign to this new test.
5. Edit the test steps as desired and proceed as you would for any new test.
6. When you are finished making changes, click **Update**.

**Note:** Scope management in ATF tests helps identify and restrict copying of tests in other scopes. If you want to copy a test, you must be in the same scope as the test. See **Application Scope** for more information.

**Run an automated test**

After creating an automated test, run it on a non-production instance.

**Before you begin**

You must have created the test you want to run.

The **test execution property** must be enabled. You must have an admin or atf_test_admin role to do so.
Note: The test execution property is disabled by default to prevent running tests on a production system. Run tests only on development, test, and other sub-production instances.

Role required: atf_test_admin, atf_test_designer, or admin

Procedure

1. Navigate to Automated Test Framework > Tests.
2. If necessary to view the tests list, click Tests.
3. Click the row containing the test you want to run. The system displays the Test form.
4. Click Run Test.

Note: If the test execution property is not enabled, the Run Test button does not appear. In this case, see the annotation at the top of the form, and click the link to enable running tests.

5. If the test includes a form step (any step involving a UI), or other kinds of UI test steps, the Pick a browser dialog appears before executing the tests. Use it to choose among any currently running test clients, or start a new runner. For more information, review Browser recommendations for all tests and suites. If the test only includes server test steps, the system executes the tests without displaying the Pick a Browser dialog.

What to do next

Monitor the progress of the test in the Run Test progress dialog. When complete, click Go to Results (on the Run Test progress dialog) to display the Test Results list, where you can view and analyze test results.

Note: If your test creates data, the system rolls back that data after all steps in the test complete.

Related information

Cancelling automated tests and test suites

View test results

View test results from completed test and test suite runs. Carefully consider the results of automated test runs and perform any corrective actions required to resolve any revealed issues.
About this task
Test and test suite results show how long it took the system to execute a test, which steps failed, and can include screenshots of form steps. You can also view reports comparing different runs of the same test suites.

View automated test results
You have multiple options for navigating to the test results, depending on where you are in the user interface. For example, if the Run Test progress dialog is displayed, you can click Go to Result. This procedure enables you to view test results from any location in the user interface.

Procedure
1. Navigate to Automated Test Framework > Test Results.
2. Click a row to access the results for a specific test. The system displays the Test results record.
3. Click the Step Results related list to view step results for the selected test. Scroll down and click the row for the step result you want to view. The system displays the Step results record.
4. Click the Test Log related list to view test logs for this test result. Scroll down and click the row for the test log you want to view. The system displays the Test logs record.

What to do next
Note: When viewing test results, step results or test logs, you can allow client errors as ignored or warning entries in the Allowed Client Errors table. This allows test executions to continue past client errors in future test runs.

View results screenshots from an automated test
If the test has a UI component, the system takes screenshots of the UI. View these screenshots to gain further insight into the test results.

Before you begin
You must have run the test whose results screenshots you want to view.
Note: For best results with screen shots, leave the browser zoom level set to 100%.

Role required: atf_test_admin or atf_test_designer

Procedure
1. Navigate to Automated Test Framework > Test Results.
2. Click the row containing the test whose results you want to view.
The system displays the **Test Results** form.

3. To view a screenshot taken during the test, find the screenshot you want in the attachments list. Screenshots are named with the word **screenshot** followed by the timestamp (always in UTC time) for when the system recorded the shot. You can match the screenshot to the test step by comparing the step and screenshot timestamps.

**View automated test suite results**
View results from an automated test suite.

**About this task**

ℹ️ **Note:** You have multiple options for navigating to the test suite results, depending on where you are in the user interface. For example, if the Run Test progress dialog is displayed, you can click **Go to Result** instead. The procedure described here enables you to view test suite results from any location in the user interface.

**Procedure**

1. Navigate to **Automated Test Framework > Suite Results**.
2. Click the row containing the test whose results you want to view. The system displays the **Test suite results record**.

**Related information**

- Identify and resolve client errors
- Allow client errors from test results
- Allow client errors from step results
- Allow client errors from the test logs

**Identify and resolve client errors**
Identify client errors and resolve them in client-side scripts.

**Before you begin**
Role required: admin

**About this task**

When client errors occur, the Automated Test Framework fails the test on the step that was executing when the error occurred. Even though client-side scripts can fail silently on JavaScript errors while procedures are executing, the error may still impact data, and the procedure being executed. The Automated Test Framework considers these errors as validation failures.
Procedure

1. Navigate to Application > Automated Test Framework > Tests and run an Automated Test Framework test that interacts with a form.

2. In the test results for this test, check for a step result with the following summary:

   This step failed because the client error 'DETAILED ERROR MESSAGE' was detected on the page being tested. See failing Test Logs. To ignore these errors in the next test run, use ‘Add all client errors to warning/ignored list’ links.

   This step result appears only on a step that interacts with the UI.

3. To identify and resolve these script errors, open the developer tools browser console on the Client Test Runner page.

   Note: For information about how to open the browser console, see the following article: http://webmasters.stackexchange.com/questions/8525/how-to-open-the-javascript-console-in-different-browsers

4. If you can see the error, try to identify and troubleshoot the problematic client-side script, which may be on any of the following base system tables:
   - ServiceNow Client Script
   - UI Action
   - UI Macro (HTML script)
   - UI Page (HTML script)
   - UI Policy
   - UI Script
   - Tables that extend the preceding base system tables

5. Review the script for errors and once you've fixed them, run your test again.

What to do next
Determine the source of the client error by reviewing the script version history. If you customized a base system script, it's possible that the script has new versions that were skipped during upgrade.

Example client errors
There are several types of common client error.
**Client JavaScript errors**

When a client script causes an error, the browser console displays an error similar to the following example:

```
***************************************************
A script has encountered an error in render events
TypeError: Cannot read property 'id' of undefined
Script ends. Continuing happily
***************************************************
[00:00:00.002] onLoad Modify Comments Label
```

In this example, the client script Modify Comments Label caused the error.

**Other client script errors**

Any other type of script error reports directly to the browser console with any formatting upon occurrence.

`TypeError: callbacks(id) is undefined`

**Script resource links by Sys ID**

In some cases, the console error provides a link to the script resource file using its Sys ID. Following this link may give context to which script had executed it.

```
Uncaught ReferenceError: myobj is not defined
  at incident.do?sys_id=12345678901234567890123456789012
(anonymous) @ incident.do?sys_id=12345678901234567890123456789012  <----- LINK
```

**Script access permissions**

While identifying problematic scripts, be sure the script has permission to access data. Check:

- Access control rule permissions for tables and fields.
- Application access permissions if the script accesses applications in a private scope.
- Domain separation permissions if domain separation is configured.

**Related information**

- [Allowed client errors](#)

**UI test steps**

Test user interfaces by mimicking user actions and interacting with the visible components of a page.
Client test runner dependency
UI test steps require an active client test runner to act directly on the visible components of a page. A tester must manually start one or more client test runners for UI testing. Test designers can schedule selecting an open client test runner from a test suite. See Working with client test runners.

Intelligent wait mechanism
UI test steps have an intelligent wait mechanism triggered by UI changes such as clicking a component or setting a value. The wait mechanism requires the UI change to complete before the next UI test step can proceed. Test designers do not need to manually add wait mechanisms between UI test steps.

Custom UI test steps
Test customized user interfaces such as UI pages and UI macros by retrieving their HTML and JavaScript page components and identifying the test actions they support.

Custom UI test steps require the Automated Test Framework to retrieve and identify the testable components from a target web page.

Testable components
Testable page components consist of standard HTML and JavaScript with these characteristics.

Are set or clicked by user interaction
Testable page components allow users to set a value or click them.

Are accessible from the Document Object Model (DOM)
Testable page components are accessible from the DOM and support JavaScript manipulation of the DOM. Custom UI test steps cannot access page components in the shadow DOM.

Are accessible to JavaScript
Testable page components are accessible to JavaScript. Custom UI test steps cannot access page components that interact directly with the operating system such as file fields or display non-HTML content such as Excel or PDF files.

New browser tabs or windows are not supported by Custom UI test steps.

Are not excluded from custom UI testing
Testable page components are not excluded from custom UI testing. Automated Test Framework excludes page components
that are already testable by other test step categories and also excludes page components associated with Now Platform features.

**Are accessible to the Page Inspector**

Testable page components must return results when viewed from the Page Inspector. Test designers can use the Page Inspector to identify the testable components of a page.

Examples of testable page components include these UI elements.

- Buttons
- Links
- Page text
- UI controls
- UI macros
  - ui_date
  - ui_date_time
  - ui_reference
- UI pages
- Wizards

Examples of untestable page components include these UI elements.

### Example untestable page components

<table>
<thead>
<tr>
<th>Reason untestable</th>
<th>Untestable page components</th>
</tr>
</thead>
<tbody>
<tr>
<td>Are not settable or clickable</td>
<td>Hidden controls</td>
</tr>
<tr>
<td></td>
<td>HTML comments</td>
</tr>
<tr>
<td></td>
<td>HTML layout elements such as div, section, and span.</td>
</tr>
<tr>
<td></td>
<td>HTML script elements</td>
</tr>
<tr>
<td>Are inaccessible from DOM</td>
<td>Dashboards</td>
</tr>
<tr>
<td></td>
<td>Images</td>
</tr>
<tr>
<td></td>
<td>Lists</td>
</tr>
<tr>
<td></td>
<td>Reports</td>
</tr>
<tr>
<td></td>
<td>Shadow DOM</td>
</tr>
</tbody>
</table>
Example untestable page components (continued)

<table>
<thead>
<tr>
<th>Reason untestable</th>
<th>Untestable page components</th>
</tr>
</thead>
<tbody>
<tr>
<td>Are inaccessible to JavaScript</td>
<td>Excel files</td>
</tr>
<tr>
<td></td>
<td>File fields</td>
</tr>
<tr>
<td></td>
<td>PDF files</td>
</tr>
<tr>
<td>Are Now Platform features</td>
<td>Flow Designer</td>
</tr>
<tr>
<td></td>
<td>Studio</td>
</tr>
<tr>
<td></td>
<td>Upgrade Monitor</td>
</tr>
<tr>
<td>Are testable by other test step categories</td>
<td>Form field labels</td>
</tr>
<tr>
<td></td>
<td>Form field values</td>
</tr>
<tr>
<td></td>
<td>Service Catalog</td>
</tr>
<tr>
<td></td>
<td>Workspaces</td>
</tr>
</tbody>
</table>

Settable page components

A _settable_ component is a UI element that has a dynamic value such as a text input field. Settable components support these test actions and test steps.

Test options for settable components

<table>
<thead>
<tr>
<th>Page Inspector actions</th>
<th>Custom UI test steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Component Value</td>
<td>Set Component Values (Custom UI)</td>
</tr>
<tr>
<td>Get Component Value</td>
<td>Assert Text on Page (Custom UI)</td>
</tr>
<tr>
<td></td>
<td>Component Value Validation (Custom UI)</td>
</tr>
<tr>
<td>Is Component Disabled</td>
<td>Component State Validation (Custom UI)</td>
</tr>
</tbody>
</table>

Settable components have a _data type_ that determines what values a Custom UI test step can set. For example, a page component intended to display a reference to a particular record can have a reference data type to only display Sys ID values.

Automated Test Framework allows UI developers to specify a data type to use during custom UI testing. UI developers can assign page components a data type to ensure that a test step sets a valid value. These data types are supported.
• Date
• Date Time
• Reference

See Override component data type for more information.

**Clickable page components**

A clickable component is a UI element that users can interact with by clicking, such as inputs of type check box or radio. Clickable components support these test actions.

<table>
<thead>
<tr>
<th>Test options for clickable components</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Page Inspector actions</strong></td>
</tr>
<tr>
<td>Click On Component</td>
</tr>
<tr>
<td>Get Component Value</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Is Component Disabled</td>
</tr>
</tbody>
</table>

Clickable components do not have a data type since they do not have dynamic values.

**Retrieved page components**

Automated Test Framework stores a list of the retrieved page components for each custom UI page you test. Custom UI test steps display the list of retrieved components from the **Component** and **Component values** fields.

By default, the list of page components is static and is only updated when Test designers manually click **Retrieve Components**. Administrators can enable the system property **sn_atf.page_data_capture.enabled** to refresh the list of page components every time a Custom UI test step is run. Enabling this property during test design ensures that your test designers always have access to the most current list of page components. Disabling this property after test design is complete allows your tests to run faster because test steps can use the previously retrieved list of page components.

The Now Platform treats the list of retrieved page components as data and does not include them in update sets or applications files. When transferring tests from one instance to another, test designers must manually retrieve page components again.

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Design considerations

Follow these design considerations when testing custom UI pages and page components.

Use the page inspector to identify testable page components

The page inspector determines which page components are available for custom UI testing. Page components that are unavailable to the page inspector are unavailable to custom UI testing.

Navigate to the custom UI you want to test

Use existing test steps to navigate to the target custom UI. For example, to test a Knowledge Base article, use the existing test steps to navigate to a module or to open an existing record. Most custom UI testing requires using existing test step categories as part of the test.

Use the component area to identify page components

The component area describes the HTML layout element containing the component such as a `<div>` or `<section>` element. The area helps test designers distinguish between components by providing the location in the page layout.

Test your custom UI rather than Now Platform UI

The Automated Test Framework prevents custom UI testing of Now Platform features. For example, you cannot test dashboards or graphical designers. Instead, build tests to validate your custom UI pages and elements since you have direct control over these user interfaces.

Use HTML attributes to override page component testing properties

Change the testing properties of a particular page component using Automated Test Framework-specific HTML attributes. See Override component test actions.

Retrieve page components again when you move tests to another instance

Custom UI test steps do not store UI components as metadata. Testers must manually retrieve page components again when moving tests between instances.

Example custom UI testing

You can use the list of retrieved components to design custom UI test steps. For example, suppose that you want to test reviewing and commenting on
a Knowledge Base article. A Knowledge Base article contains several page components that require custom UI steps to test.

For example, these page components require custom UI test steps.

1. The number of article views.
2. The buttons to mark the article as Helpful.
3. The text area to Leave a comment.

These steps demonstrate custom UI testing on a Knowledge Base article. The example test consists of these existing and custom UI test steps.

1. **Navigate to Module.** Navigate to the 'Published' module in the 'Knowledge' application.
2. **Open an Existing Record.** Open the 'Knowledge' form with id 'Knowledge: KB0011110'.
3. **Click a UI Action.** Click UI Action 'View Article' on 'Knowledge' form.
4. **Assert Text on Page (Custom UI).** Assert that the text 'developers' is on the page.
5. **Set Component Values (Custom UI).** Set the components on the page as follows: `Textarea <textarea> [article_comments]` = Update with actual article rather than URL to article elsewhere.

6. **Click Component (Custom UI).** Click the component: 'Button <button>: Comment'.

7. **Assert Text on Page (Custom UI).** Assert that the text 'Update with actual article rather than URL to article elsewhere.' is on the page.

### Example test steps for a knowledge base article test

![Example test steps for a knowledge base article test](image)

**Custom UI component version and order**

When you select a component in any of the custom UI test steps, the Custom UI version might show up. If there are multiple
duplicate components, the order of the components show up.

⚠️ **Note:** The Custom UI version shows up only if there are at least two different components with different Custom UI versions.

⚠️ **Note:** The Order shows up only if there are duplicate components in the component drop-down menu. You can disambiguate them according to the displayed order.

⚠️ **Note:** Both Custom UI version and Order show up if there are multiple duplicate components from different versions.

**Identifying components**
Implement an alternative way to identify your component by using the sn-atf-id attribute. This is useful if the name or id attribute of your component is dynamic and changes every time a test runs. Add the sn-atf-id attribute with a consistent value to allow ATF identify your component when running a test. It also allows you to identify your component when building a test. For example, in a button component

```html
<button sn-atf-id="consistentValue">Test</button>
```

⚠️ **Note:** Starting with the Rome release, if you have exactly one component on the page that has sn-atf-id attribute, ATF finds that component irrespective of any other attributes on that component.
**Improve ATF component identification**

Identify your component using the label path included in the **Page area** column. Starting with the Rome release, the `sn_atf.element.use_label_path` property has been set to true by default.

If a component has the `sn-atf-area` attribute, the **Page area** column displays the `sn-atf-area` value. If the `sn-atf-area` attribute is not present, the label path for that component is shown in the **Page area** column.

⚠️ **Note:** If you have multiple, identically named Label values, the label path helps you identify the correct component.

⚠️ **Note:** If a component doesn't have an `sn-atf-area` attribute or a label path, the value is displayed as default.

When you select the required component from the list and click **Submit**, the description of the test step also gets updated with more details.
Page Inspector

Identify the HTML and JavaScript page components in your user interfaces that are available for custom UI testing. Enable automated testing by ensuring that your user interfaces only contain testable page components.

The Page Inspector is a developer setting that opens a new pane beside any currently displayed Now Platform page.

The Page Inspector provides these features.

1. A highlighted frame to identify the currently inspected page.
2. A banner icon to display or hide the Inspector pane.
3. A button to refresh the list of page components.
4. A button to hide the Inspector pane.
5. An inspector button to identify specific components on a page.
6. A list of page components available for custom UI testing.
7. A page component search filter.
8. A row to click to see more information about a page component.
9. A highlighted row and tooltip to preview information about the currently inspected component.
10. A highlighted page component to identify the currently inspected component on the page.
11. A component label and back button to return to the list of page components.
12. A list of actions to take on the current page component.
13. A list of attribute information for the current page component.
14. An attribute name and value pair for a page component.

**Page Inspector view of component details**

The page inspector launcher helps you select and launch a specific page within the page inspector. See *Inspect different page types* for more information.
Testable page components

The Page Inspector retrieves the list of testable page components when you first load a page. Testable components consist of standard HTML and JavaScript that are accessible to the Automated Test Framework. Test designers can use these components as part of custom UI testing.

Untestable page components

The Page Inspector identifies these types of untestable page components.

Exclusion listed

Exclusion listed page components can't be tested using any Custom UI test steps. Exclusion listed page components typically include specialized Now Platform user interfaces and components already testable by other means. Test designers can't create custom UI tests for exclusion listed page components. To test these components, they must use another supported test category such as Forms or Service Catalog.

In the Page Inspector, the exclusion listed interfaces are indicated by a grey background color when the inspect icon is dragged over them. A detailed message about the step category that needs to be used to test these components is displayed by dropping the inspect icon on an exclusion listed component.

- Lists can't be tested using custom UI test steps. Use List and Related List test step category to test lists.

Sample exclusion listed list message

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
• Forms can't be tested using custom UI test steps. To test forms, use Form category test steps.

**Sample exclusion listed form message**

1. Forms are not supported by Custom UI steps. Use the Form category to test forms

**Note:** The UI formatters within forms can be accessed and tested using custom UI test steps. See Create a custom UI test for more information.

• Service Catalog items can't be tested using custom UI test steps. To test catalog items, use Service Catalog category test steps.

**Sample exclusion listed catalog item message**

1. Service Catalog is not supported by Custom UI steps. Use the Service Catalog step category

• Certain components are excluded from custom UI pages and are forbidden from being tested.

**Sample exclusion listed component message**

1. This component is not accessible

• Workspaces can't be tested using custom UI test steps. To test forms in an available workspace, use Form category test steps.

**Sample exclusion listed workspace message**

1. Workspaces are not supported by Custom UI steps. Use the Form Step category to test forms in workspaces

---

**Inaccessible**

**Inaccessible** page components are elements that Automated Test Framework either cannot identify or does not have permission to test because of some configuration issue. Inaccessible page components typically include third-party JavaScript libraries or elements with a Shadow DOM. Test designers cannot create custom UI tests for inaccessible page components, but may be able to redesign the page to use components accessible to Automated Test Framework.

• Certain components are not accessible in the Automated Test Framework

**Sample inaccessible component message**

1. This component is not accessible

• Team development is not supported in the Automated Test Framework
Inspect different page types

Inspect and troubleshoot the functionality of different page types like UI Pages, Service Portal, Standard UI, and Custom URL using the Page Inspector.

Before you begin
Role required: admin, atf_test_designer, atf_test_admin

Procedure
2. Select the Page Type you want to inspect.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UI Pages</td>
<td>Existing UI pages. Select a starting page from the available list.</td>
</tr>
<tr>
<td>Standard UI</td>
<td>Standard platform forms, lists, and some UI pages. Select a form or a list or a UI page in the Starting Page field. Note: The optional Record field shows up only if you select a form in the Starting Page field.</td>
</tr>
<tr>
<td>Service Portal</td>
<td>Any available portal in the instance. Select an available portal and a starting page to be inspected. Note: Portal field shows up only if you select Service Portal as the Page Type.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>Custom</td>
<td>Custom Platform URL to be inspected.</td>
</tr>
</tbody>
</table>

⚠️ **Note:** Don’t copy-paste the complete URL onto the Starting Page URL field. For example, include only `/home.do` as the input if you want to inspect `https://instance.service-now.com/home.do`

3. Click **Inspect**.

### Enable and use the page inspector

Enable a developer setting to inspect UI pages that open within the platform. Use the Manual Page Inspector to inspect pages that open in a new tab, such as Service Portal pages.

**Before you begin**

Role required: admin

**About this task**

Identify the HTML and JavaScript page components in your user interfaces that are available for custom UI testing. Enable automated testing by ensuring that your user interfaces only contain testable page components.

**Procedure**

1. Enable the Page Inspector depending on the type of custom UI page you need to inspect. Some custom UI pages open in a new tab and can only be inspected manually.

<table>
<thead>
<tr>
<th>Option</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inspect page opening in the Now Platform.</td>
<td><strong>a.</strong> Navigate to the record or URL for the page you would like to inspect.</td>
</tr>
<tr>
<td></td>
<td><strong>b.</strong> Open System Settings and select the Developer tab.</td>
</tr>
<tr>
<td></td>
<td><strong>c.</strong> Enable Automated Test Framework Page Inspector.</td>
</tr>
<tr>
<td>Option</td>
<td>Procedure</td>
</tr>
<tr>
<td>--------</td>
<td>-----------</td>
</tr>
<tr>
<td>Inspect page opening in a new tab, such as a Service Portal page.</td>
<td>The page opens in the Page Inspector and the inspect icon appears in the banner frame.</td>
</tr>
</tbody>
</table>

b. Set the Page Type and select the page. If the page type is Service Portal, select the portal in which the page opens.  
c. Click Inspect.  
The page opens in the page inspector.

2. Select a component to inspect.  
   - Drag the inspect icon (✠) from the Page Inspector pane to a component on the page.  
   - Select an available component from the Page Inspector pane.  
The Page Inspector retrieves the list of testable page components when you first load a page. Testable components consist of standard HTML and JavaScript that are accessible to the Automated Test Framework. Test designers can use these components as part of custom UI testing.

3. View component information and perform actions available in the Action field.  
Performing available actions helps you manually confirm which test steps are available to a component. For example, when you select Click On Component, the Page Inspector selects the component and displays the resulting page.

What to do next  
Create a custom UI test

Create a custom UI test
Test components in custom UI pages.

Before you begin  
- Use the Page Inspector to identify testable custom UI components. See Enable and use the page inspector.  
- Role required: admin
Procedure
1. Navigate to Automated Test Framework > Tests.
2. Click New.
3. Enter a name and description for your test.
4. Click Save.
5. In the Test Steps related list, click Add Test Step.
6. Add test steps to navigate to the target custom UI page.

Example
Use the Navigate to Module step to open a page that has an application navigator module. Use the Open Service Portal Page step to open a portal page.
To open a UI page, use these test steps:

a. Open an Existing Record: Open the record for the page.

b. Click UI Action: Click the Try It UI action to open the page.

7. Add test steps from the Custom UI category to validate the behavior of custom UI components. For a list of available Custom UI test steps, see Custom UI category.

8. Retrieve UI components when adding a test step.

Identify the testable page components on a custom UI page by retrieving a list of UI components for test steps. You can collect the components of the current page by selecting Retrieve Components.

The system retrieves the components using an existing or a new client test runner. Ensure that the client test runner tabs used in the retrieval process are running in the foreground. For tests that don’t yield any components after selecting Retrieve Components, the following warning message is displayed and the Next button is disabled.
If you have already retrieved a list of page components, click **Next** to use the previously retrieved list. You can also click **Retrieve Components** to rerun the current steps and refresh the list of testable page components. If the test fails while retrieving components, the following warning notification is displayed. You can review the test results by clicking the link in the notification.

Automated Test Framework stores the list of page components and displays the list in the **Component values** or **Component** field on the test step form. The field displays this information about each component:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>HTML element description and tag. For example, <code>Text&lt;input&gt;</code>.</td>
</tr>
<tr>
<td>Label</td>
<td>HTML component label.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Value</td>
<td>Value of the component. For example, if the component is a search input, the value might be <strong>Search</strong>.</td>
</tr>
<tr>
<td>Page area</td>
<td>HTML layout element that contains the component such as a <code>&lt;div&gt;</code> or <code>&lt;section&gt;</code> element.</td>
</tr>
</tbody>
</table>

If you create a parameterized test that includes Custom UI test steps, the system only uses the first data set to retrieve components.

**What to do next**
Consider adding tests to a test suite. For more information, see Building and running automated test suites.

**Override component test actions**
Change the testing properties of a particular page component using Automated Test Framework-specific HTML attributes.

**Before you begin**
Role required: admin

**About this task**
When Automated Test Framework retrieves a component, it determines which interactions it supports, such as whether it is a settable or clickable component. If the component is settable, Automated Test Framework determines the field type that can be set. If Automated Test Framework incorrectly determines your custom component's actions or field types, or your component contains multiple DOM elements that should be treated as one entity, explicitly set them using Automated Test Framework-specific HTML attributes.

**Using sn-atf-clickable and sn-atf-settable attributes**
Use `sn-atf-clickable` and `sn-atf-settable` attributes to specify that an element and its child elements should be treated as a custom clickable or custom settable component.

**Before you begin**
Role required: admin

**Procedure**
1. Open the custom UI page you would like to test.
2. Add the `sn-atf-clickable` or `sn-atf-settable` attribute to the element being tested.
Example

```html
<div sn-atf-clickable="true" sn-atf-disabled id="customClickable">
    <button id="customButton">Click me</button>
</div>

<div sn-atf-settable="true" id="customSettable" sn-atf-component-value="A default value">
    <input id="customInput" value="A default value"></input>
</div>
```

**Note:** You can use either `sn-atf-clickable` or `sn-atf-settable` attribute to specify if an element should be treated as a custom clickable or custom settable component. You can’t use both attributes on the same element.

3. Use either the `sn-atf-clickable` or `sn-atf-settable` attribute.

- **sn-atf-clickable**: If you added the `sn-atf-clickable` attribute, ATF clicks the component by sending an `sn-atf-click` event to the DOM element with the `sn-atf-clickable` attribute. You should add an event listener (for example, using `addEventListener`) to this DOM element, and implement your custom click logic for the component. You can interact with a custom clickable component using Click Component test step.

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sn-atf-disabled</code></td>
<td>The presence of this optional attribute (regardless of its value) tells ATF that this component is disabled</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If this attribute is missing, ATF considers that this component is enabled by default.</td>
</tr>
<tr>
<td><code>sn-atf-component-value</code></td>
<td>A string or JSON object that tells ATF the current value of this component</td>
</tr>
</tbody>
</table>

- **sn-atf-settable**: If you added the `sn-atf-settable` attribute, ATF sets the component value by sending an `sn-atf-setvalue` event to the DOM element with the `sn-atf-settable` attribute. You should add an event listener (for example, using `addEventListener`) to this DOM element, and implement your custom set value logic for the component. The value that needs to be set can be accessed with `event.detail.newvalue`. The `event` argument is passed to your event handler. You can interact with a custom settable component using Set Component Values test step.
## Settable component attributes

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn-atf-disabled</td>
<td>The presence of this optional attribute (regardless of its value) tells ATF that this component is disabled.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If this attribute is missing, ATF considers that this component is enabled by default.</td>
</tr>
<tr>
<td>sn-atf-component-value</td>
<td>A string or JSON object that tells ATF the current value of this component.</td>
</tr>
<tr>
<td>sn-atf-data-type</td>
<td>Optional type of field to present to user when building a step. It defaults to string. For example, glide_date_time, reference, boolean, etc.</td>
</tr>
<tr>
<td>sn-atf-data-type-params</td>
<td>JSON object with more data type details.</td>
</tr>
</tbody>
</table>

### Example:

```html
//A custom clickable component

<div sn-atf-clickable="true" sn-atf-disabled id="customClickable">
  <button id="customButton">Click me</button>
</div>

<script>
  var customClickableDiv = document.getElementById("customClickable");
  customClickableDiv.addEventListener('sn-atf-click', function() {
    document.getElementById('customButton').click();
  });
</script>

//A custom settable component

<div sn-atf-settable="true" sn-atf-component-value="A default value">
  <input id="customInput" value="A default value"/>
</div>

<script>
  var customSettableDiv = document.getElementById("customSettable");
  customSettableDiv.addEventListener('sn-atf-setvalue', function(event) {
```
Using sn-atf-class attribute

Use the `sn-atf-class` attribute to specify the JavaScript object to use when testing a custom clickable or settable component. Write a custom JavaScript object to specify the test actions available for a custom component.

Before you begin
Role required: admin

About this task
Test designers can manually specify the test actions available for a custom component by writing a custom JavaScript object and assigning the component a `sn-atf-class` attribute. Set the value of the attribute to the name of the JavaScript object containing the component test actions. Testable custom components must be either clickable or settable, and this classification determines the functions and properties your JavaScript object requires. See Custom UI test steps for testable page component requirements.

Procedure
1. Open the custom UI page you would like to test.
2. Add the `sn-atf-class` attribute to the element being tested. Set the value to the name of the JavaScript object embedded in the page that handles `getValue()`, `setValue()`, `click()`, or `isDisabled()` functions.

Example

```html
<div sn-atf-class="MyClickableComponent">
  <label for="a_clickable_checkbox">MyClickableComponent</label>
  <input type="checkbox" id="a_clickable_checkbox" checked="true"/>
</div>
```

3. Create the JavaScript object specified in the `sn-atf-class` attribute.
   Add the functions and attribute needed to identify your custom page component as either a clickable or settable page component.

   **Clickable component functions**

<table>
<thead>
<tr>
<th>Function name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>initialize()</code></td>
<td>Gets the initial values of the component. Enter:</td>
</tr>
<tr>
<td>Function name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>$super(element, area)</td>
<td></td>
</tr>
</tbody>
</table>
| click() | Selects the component. Returns a JSON object with these properties:  
| • success: true if the component can be clicked.  
| [Note: Triggers the UI test step intelligent wait mechanism.](#) |
| getValue() | Gets the value of the element. Returns a JSON object with these properties:  
| • success: true if the value is retrieved.  
| • result: the value of the component. |
| isDisabled() | Whether the component is disabled. Returns a JSON object with these properties:  
| • success: true if the component is disabled.  
| • result: true if the component is disabled. |

### Settable component functions

<table>
<thead>
<tr>
<th>Function name</th>
<th>Description</th>
</tr>
</thead>
</table>
| initialize() | Gets the initial values of the component. Enter:  
| $super(element, area) |
| setValue(newValue) | Sets the value of the component. See the second example below. Returns a JSON object with these properties:  
| • success: true if the value is set.  
| [Note: Triggers the UI test step intelligent wait mechanism.](#) |
| getValue() | Gets the value of the element. Returns a JSON object with these properties:  
| • success: true if the value is retrieved.  
<p>| • result: the value of the component. |
| isDisabled() | Whether the component is disabled. Returns a JSON object with these properties: |</p>
<table>
<thead>
<tr>
<th>Function name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• success: true if the component is disabled.</td>
<td></td>
</tr>
<tr>
<td>• result: true if the component is disabled.</td>
<td></td>
</tr>
</tbody>
</table>

**Settable component attribute**

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isSettable: true</td>
<td>Identifies the component as a settable page component.</td>
</tr>
</tbody>
</table>

4. When creating your custom component in Jelly, add `<g2:atf_only>` tags around the JavaScript object specified in the `sn-atf-class` attribute. These tags ensure the system only runs the JavaScript object during automated testing.

**Example:**

```javascript
//A custom clickable component

<form>
  <div sn-atf-class="MyClickableComponent">
    <label for="a_clickable_checkbox">MyClickableComponent</label>
    <input type="checkbox" id="a_clickable_checkbox" checked="true"/>
  </div>
</form>

<script>
var MyClickableComponent = {
  // The constructor must have this signature, but you can perform additional setup after
  // the $super(element, area) call
  initialize: function($super, element, area) {
    $super(element, area);
  },

  click: function() {
    document.getElementById('a_clickable_checkbox').click();
    return {success: true};
  },

  getValue: function() {
    var isChecked = document.getElementById('a_clickable_checkbox').checked ? "true" : "false";
    return {success: true, result: isChecked};
  }
};
```

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
// The function returns an object with a result attribute of type Boolean
isVisible: function() {
  if (document.getElementById('a_clickable_checkbox').disabled)
    return {success: true, result: true};

  return {success: true, result: false};
},

};
</script>

// A custom settable component

<form>
  <div sn-atf-class="MySettableComponent">
    <label for="a_settable_checkbox">MySettableComponent</label>
    <input type="checkbox" id="a_settable_checkbox" checked="true"/>
  </div>
</form>

<script>
var MySettableComponent = {

  // This attribute is required for settable components
  isSettable: true,

  // The constructor must have this signature, but you can perform additional setup after
  // the $super(element, area) call
  initialize: function($super, element, area) {
    $super(element, area);
  },

  // The value parameter is a string
  setValue: function(value) {
    document.getElementById('a_settable_checkbox').checked = (value == "true");
    return {success: true};
  },

  // The function returns an object with a result attribute of type String
  getValue: function() {
    var isChecked = document.getElementById('a_settable_checkbox').checked ? "true" : "false";
    return {success: true, result: isChecked};
  }
};
</script>
Reference and record picker

Use custom UI steps to manipulate the values of the `sn-reference-picker` and `sn-record-picker` angular directives. The value on a reference picker returns the `sys_id` of the chosen record. The value on a record picker returns the value field chosen for that record picker. Both elements can be set by selecting a record to set as their value.

Override component data type

Use the `sn-atf-data-type` and `sn-atf-data-type-params` attributes to override the type of field displayed in a Set Component Value test step.

Before you begin
Role required: admin

About this task
Settable components have a data type that determines what values a Custom UI test step can set. For example, a page component intended to display a reference to a particular record can have a reference data type to only display Sys ID values.

Procedure
1. Open the custom UI page you would like to test.
2. Add the `sn-atf-data-type` attribute to the settable component. Set the value to the field type you would like displayed in the Set Component Value test step.
   
   This attribute contains a string of the testable data type. Available values include:
<table>
<thead>
<tr>
<th>Attribute value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>glide_date</td>
<td>Contains a string specifying a particular day.</td>
</tr>
<tr>
<td>glide_date_time</td>
<td>Contains a string specifying a particular day and time of day.</td>
</tr>
<tr>
<td>reference</td>
<td>Contains a Sys ID to a related record. This data type requires specifying additional information in the <code>sn-atf-data-type-params</code> attribute.</td>
</tr>
</tbody>
</table>

3. Add the `sn-atf-data-type-params` attribute to provide additional information when the value of `sn-atf-data-type` is `reference`.

This attribute contains a string of JSON formatted key-value pairs. Available key-value pairs include:

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>reference</td>
<td>Name of the table that contains the records you want the reference field to display. For example, add &quot;reference&quot;:&quot;incident&quot; to display records from the Incident table.</td>
</tr>
<tr>
<td>reference_qual</td>
<td>Filter to apply to the query. For example, add &quot;reference_qual&quot;:&quot;active=true&quot; to display only active Incident records. See Reference qualifiers</td>
</tr>
</tbody>
</table>

**Example:**

```html
<input id="someTextField" value="someSysId" sn-atf-data-type="reference" sn-atf-data-type-params='{"reference":"incident","reference_qual":"active=true"}'/>
```

**Select2 functionalities in ATF**

Use the Select2 component to search and select your option from a drop-down menu easily.

You can set the value of select2 in a test. The test then searches for the value in the search bar of Select2. The first valid result in the component drop-down gets selected.

Drag and drop the page inspector on the required drop-down menu. The right panel for the selected component opens. You can specify the text you want to search in the set component value step within the page inspector.
Select the **Set Component Value** option under **Action** to search for a component. Type in the text in the right panel and click **Submit**. This enables the code to open the select2 to search for that component and pick the first option that shows up.

<details>
<summary>Note:</summary>
If multiple, similarly named component options show up on searching a term, only the first option is selected.

</details>

### Limitations of the select2 support
- Only certain versions (3.5.1-3.5.4 and 4.0.0-4.0.13) of the select2 library are supported.
- The select2 support depends on the jquery library. So, if you try to set a value of select2 and the name of the jquery library is changed, it causes the test to fail.
- Searching through Select2 won’t work when there is no search bar.

### Design considerations
Search and select your options efficiently with the following design considerations:
- It is recommended to use uniquely searchable test data
- Use a Component Value Validation step to validate the value set for select2
- Prevent failing of tests by avoiding jquery library name change
- Select2 Adapter and Decorator features are not supported
Browser recommendations for Automated Test Framework

Configure client test runner browsers to run automated tests and avoid performance degradations.

Periodic browser restarts
These browsers have memory-management limitations that make it necessary to occasionally close and restart the browser when running the client test runner.

- Internet Explorer
- Edge
- Older versions of Firefox

How often you should close the browser depends on the memory allocation in the browser application.

Browser CPU throttling
Some browsers throttle CPU usage for windows that are out of focus. Follow these guidelines to avoid CPU throttling issues.

- Run each client test runner in its own browser window.
- Ensure that the client test runner browser window is always partially visible on the screen.
- Ensure that the system screen is not locked or shut off.

Browser zoom level
Client test runners take screen shots as they run tests. For best results with screen shots, leave the browser zoom level set to 100%.

OS X CPU throttling
On OS X with the client test runner on Chrome or Safari: If the screen is locked or the client test runner tab is occluded when the system attempts to run the test suite, tests run significantly slower and may time out. For best performance, run client test runners for scheduled suites in a virtual machine (VM) environment in which the screen does not become locked or disabled.

Rollback in browser sessions
The session cookies roll back all the changes made during a test. When a test is running, everything performed in that session is recorded for rollback. Don't modify your instance when a test is running in the same browser session. For example, if you modify records while a test is running in the same session, the changes are rolled back after the test completes. If you navigate around in
other tabs in the same session, your work may be rolled back and interfere with
tests that rely on implicit navigation.

Parallel testing

Follow these guidelines to avoid issues when running multiple tests in parallel.

- **Run each client test runner in an incognito or private window**
  Because parallel tests roll back all changes tied to the same
  browser session, it's possible for legitimate changes made in another
  browser tab to be rolled back during parallel testing. To prevent
  unwanted rollback of changes, always run client test runners in their
  own browser session. Opening client test runners in an incognito or
  private window ensures that they always have their own browser
  session.

- **Close client test runner windows when testing is complete**
  To prevent unwanted rollback of changes, always close client
  test runners after testing is complete. Closing the browser window
  ensures that test rollback doesn’t revert any legitimate changes
  made in another browser tab.

Working with client test runners

If an automated test includes steps that involve a form or any other user-
interface (UI) element, it runs those steps in a browser tab or window called a
test runner or client test runner. The Automated Test Framework supports two
types of client test runners: Client Test Runners for manually started tests and
Scheduled Client Test Runners for tests started by a schedule.

When test execution is enabled, clicking the Client Test Runner module opens
the client test runner in the current browser session. If tests are waiting to be
run, the Client Test Runner runs a waiting test. If no test is running, the message
Waiting for a test to run displays in the client test runner.

While the client test runner is idle, it checks every five seconds for waiting tests to
start. This ensures that the system runs any tests it may have been unable to start
because no client with the proper configuration was available.

Note: The client test runner monitors for tests from the current session and
runs those tests as the logged-in user (unless it executes an Impersonate
User step). If you start a client test runner, log out from the current session,
and then log in again, the client test runner executes using the new session.

When the client runner is active, it displays the activity of the currently running
test in the **Execution Frame**.
Note: To prevent conflicts, the system allows only one test to run at a given time. This is true even if you have multiple client test runner windows open. If you submit tests to run when another test is already running, the system holds the new tests to run later. If a test remains waiting for more than ten minutes, the system cancels the test.

Test execution property
To work with the client test runner module, the test execution property must be enabled.

Note: By default, the system property that is used to run automated tests is disabled to prevent you from accidentally running these tests on a production system. To avoid data corruption or an outage, run tests only on development, test, and other non-production instances.

If the test execution property is disabled when you select this module, the system displays a message and a link to the automated test framework properties page where you can enable it.

Additional debugging functionality
If you have enabled additional debugging functionality, the client test runner module displays two tabs: Execution Frame and Debug Info. The Execution Frame displays the information normally shown by the client test runner and the Debug Info displays additional debugging information.

The system takes screen shots from the tests in the Execution Frame tab and records them to the test result record.

Browser recommendations for all tests and suites
• Some browsers have memory-management limitations that make it necessary to occasionally close and restart the browser when running the client test runner. These browsers include Internet Explorer, Edge, and older versions of Firefox. How often you should close the browser depends on the memory allocation in the browser application.

• Some browsers have features that throttle CPU time. To avoid problems, follow these guidelines:
  ◦ Run the client test runner in its own browser window.
  ◦ Keep the client test runner at least partially visible on the screen.
  ◦ Make certain the screen is not locked or shut off.

• The client test runner takes screen shots as the tests run. For best results with screen shots, leave the browser zoom level set to 100%.
**Browser recommendations for scheduled suites**

The client test runners for scheduled suites have additional browser requirements.

- On OS X with the client test runner on Chrome or Safari: If the screen is locked or the client test runner tab is occluded when the system attempts to run the test suite, tests run significantly slower and may time out. For best performance, run client test runners for scheduled suites in a virtual machine (VM) environment in which the screen does not become locked or disabled.

- The browser must meet the criteria you specified on the *Scheduled suite run record*.

- A client test runner meeting the criteria you specified on the *Scheduled suite run record* must be available to run the test suite at the scheduled time. The system cannot automatically open a client test-runner session.

**Javascript window command intercepts**

The Client Test Runner captures window object commands including `console.log`, `console.error`, `alert`, `confirm`, and `prompt`, with default responses where necessary.

- Any script that calls `window.confirm` receives a boolean response of `true`.

- Any script that calls `window.prompt` receives the string response `test value`.

**Active Test Runners table**

When you start a client test runner, the system registers that runner in the Active Test Runners table. You can view this table in the *Active Manual Test Runners* module and the *Active Scheduled Test Runners* module. These two modules provide views of the same table, filtered to show only manual or only scheduled test runners.

The Active Scheduled Test Runner module is useful when you create a scheduled suite run. For scheduled suite runs, you can specify the browser to use. To determine the name and version of a browser you want to use, start a scheduled test runner with that browser, then inspect that runner's record in the Active Scheduled Test Runners module.

The data in this table is transient. While the runner is active, it reports in to the system at a specified interval. If the runner does not report in at the expected time, the system marks the runner as inactive. After a period of time the system
deletes the runner. You can modify these intervals on the Automated Test Framework properties page.

Related reference
- Client test runner
- Scheduled client test runner

Related information
- Allowed client errors

Pick a browser
If the test or test suite you are running contains steps that work with a form (any step involving a UI), or any other UI test step element (such as Automated Service Catalog test steps), the Pick a Browser dialog appears after you click Run Test or Run Test Suite. The dialog asks you to choose among any currently-running test clients or start a new test runner.

Before you begin
You must have created the test you want to run.

The test execution property must be enabled. You must have an admin or atf_test_admin role to do so.

Note: The test execution property is disabled by default to prevent running tests on a production system. Run tests only on development, test, and other sub-production instances.

Role required: atf_test_admin, atf_test_designer, or admin

Procedure
1. All registered client test runners that are currently active appear in the Pick a Browser dialog. Choose the browser in which the test or test suite should run (or re-rerun, in the case of failed tests).
   - (Current session) indicates that the browser is currently running. See Working with client test runners.

2. Click Manage your test runners here as needed to view all client test runners registered for the current user.
   - See Active manual test runners.

3. When client test runners are only available in other browsers for the current user, Start a new test runner appears. Click it if you want to open a client test runner in this browser session.

4. Click the appropriate button to run the test. The button that appears is dependent on the type of test you are running:
a. Click **Run Test** if running a single test.

b. Click **Run Test Suite** if running a test suite.

c. Click **Re-run failed tests** if re-running failed tests.

**Results**
The test, test suite or failed tests run in the selected browser or client test runner. The Progress viewer appears for monitoring of the progress of the test run.

**Server test steps**
Test business logic and background processes by performing operations on the server.

Server test steps mimic non-interactive actions such as impersonating users, submitting or saving records, running server-side script, or making REST calls. Since server test steps run directly on the server, they do not require a client test runner. For more information about the server test steps, see **Server category**.

**REST test steps**
Test custom inbound web services and backwards compatibility by making REST calls.

REST requests can only be sent to the current instance. You cannot send a request to another instance or third-party at a remote address.

The REST test configuration only supports the XML and JSON response formats. Binary formats are not supported.

You can create tests that include steps from each of the test step configuration categories. The REST test configuration category contains the Send REST Request - Inbound and assert test configurations. Assert steps must immediately follow a **Send REST Request - Inbound** step. You can have multiple REST assert steps following a **Send REST Request - Inbound** step, but the assert steps cannot be separated from the **Send REST Request - Inbound** step by steps from other test categories. For more information about the REST test steps, see **REST category**.

**Authentication**
As part of the Automated Test Framework, there are two situations when you send REST requests:

- When you use the REST API Explorer to create and test a request
- When you run a test that contains a **Send REST Request - Inbound** step
When you use the REST API Explorer to create and test a request, and the request requires authentication, the REST API Explorer uses your credentials. When the ATF runs the test, the REST API Explorer uses the credentials of the user who scheduled the test. This means that a test might fail unintentionally because of the difference in privileges between the user who created the test and the user who runs the test.

To address the issue of user credentials, you can create a basic-authentication profile for a test user and then on the **Send REST Request - Inbound** form, specify that the profile be used when the test is run.

**Attachment test steps**

Test an attachment-dependent business rule by uploading an attachment either from a form or from a server-side API call. For example, you can have a business rule that doesn’t let you close an incident without an attachment such as a screenshot.

**Upload from form**

As a UI test step, the upload attachment step requires navigation to a form, which you can open using either **Open a New Form** or **Open an Existing Record**. Use **Upload Attachments** to select from the attachments that the test step adds to the form. When you select attachments to add to a form, the system waits to load the attachments before proceeding to the next test step. For more information on UI test dependency and wait mechanism, see **UI test steps**.

**Upload from Server API**

As a Server test step, the upload attachment step has no UI dependencies. Use **Upload Attachments** to select from the attachments that the test step adds to the record. When you select attachments to add to a form, the system waits for the attachments to be loaded before proceeding to the next test step. For more information, see **Server test steps**.

**Design considerations**

Follow these design considerations for attachment test steps:

- All attachment steps require adding one or more attachments.
- The system rolls back any attachments by the step after the test completes.
- The system cannot roll back any existing attachments after the test completes.
- Avoid testing records with existing attachments to eliminate data dependency.
• If UI testing is involved, add the attachment to a form.
• When no UI is involved, add the attachment to the Server API.

List UI actions test steps
Select a UI action from a list to perform different actions on a list or a related list. You can create a new UI action of the following types. See Create a UI action for more information.

• **List banner button**: Creates a button on the banner of a list.

![List banner button example](image)

• **List bottom button**: Creates a button at the bottom of the list.

![List bottom button example](image)

• **List context menu**: Adds an option to the context menu of the list.

![List context menu example](image)

• **List choice**: Adds an option to the list choice at the bottom of the list. You need to select one or more tests to enable the recently added list choice.
• **List link**: Adds a link to the **Related Links** list.

**Related Links**

- **Test_choice_&_link**
- **Test_list banner button**

**Design considerations**

- To use the **Click a List UI Action** test step, the test first needs to navigate to the list or the form with the related list containing the UI action.
- The **Related list** field appears only when you select Related list as the **List type**.
- **Action type** is mostly auto-filtered depending on the **List action** chosen.
- The **Record** field appears only when you select **Single record** for the UI action to be applied.
- Identify the specific record if you have selected **Single record** to apply the UI action.
- For the **Timeout** field to appear, select **Page reloaded or redirected** as the **Assert** type.

**Parameterized tests**

Run a test multiple times with different test data for each run. Create parameters to store test data for each test run.

Parameterized testing offers test designers these benefits.
• Eliminates the need to duplicate test steps just to change test data.
• Increases test reuse by separating test actions from test data.
• Produces a separate test result for each data set.

When the test runs, Automated Test Framework replaces the parameters with data set values. For example, you can create a test of the incident form that uses parameter values for the subcategory and priority fields. You can use one data set to test that the Antivirus category produces a high priority incident, and another data set to test that the Email category produces a low priority incident.

Note: The Run Server Side Script test step is not supported in parameterized tests.

Parameterized test components
Parameterized tests consist of these components.

Parameter

A parameter is a variable that stores a particular type of test data. Each parameter has a unique label and a data type. For example, you can create a parameter to store the Sys ID of a reference field or the integer value of a choice field. Define parameters during test design.

Parameters can be shared or exclusive. Shared parameters can be used in any parameterized test. Exclusive parameters can only be used with the test for which they were created. Each shared parameter is a column in the Test Run Data Sets [sys_atf_parameter_set] table. Each exclusive parameter is a record in the Parameter Variables [sys_atf_parameter_variable] table.

Data set

A data set, also known as a test run data set, includes runtime data used when the test runs. You can set a value for every parameter available to the current test. Data sets specify the parameter value during test runs. You can manually create data sets for a test, or import data from a file. Each data set is a record in the Test Run Data Sets [sys_atf_parameter_set] table.

Parameterized tests fail if data sets are not defined.

Design Considerations

Follow these design considerations when creating parameterized tests.
• Parameterized tests support standard Automated Test Framework features, such as reports, test suites, and data rollback. Copying a parameterized test copies all parameters, test run data sets, and test steps.

• If you create a parameterized test that includes Custom UI test steps, the system only uses the first data set to retrieve components.

Parameterized test runs
Automated Test Framework runs each parameterized test once per data set, using the same test steps and execution order. For example, if a parameterized test has five data sets, Automated Test Framework runs the test five times, once for each data set.

Parameterized test results
Parameterized tests display test results by the execution order of the data sets. Open each test result record to view the test details.

The Parameterized Test Result record Description field lists the parameters and data sets used in the test run.

Create a parameterized test
Build a test that uses variables to store test data.
Before you begin
Role required: atf_test_admin, atf_test_designer, or admin

Procedure
1. Navigate to Automated Test Framework > Tests.
2. Click New.
   A blank Test record opens.
3. Complete the form.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a name for the test.</td>
</tr>
<tr>
<td>Active</td>
<td>Enable</td>
</tr>
<tr>
<td>Enable parameterized testing</td>
<td>Enable</td>
</tr>
<tr>
<td>Description</td>
<td>Enter a description for the test.</td>
</tr>
</tbody>
</table>

4. Click Save.
   Parameterized testing related lists appear.

5. Create a parameter to hold test run data.
   A parameter is a variable that stores a particular type of test data. Each parameter has a unique label and a data type. For example, you can create a parameter to store the Sys ID of a reference field or the integer value of a choice field.

   a. In the Parameter Definitions related list, add a parameter.
      
      • Add Exclusive Parameters: Adds a parameter available to this test only.
      
      • Add Shared Parameters: Adds a parameter available to any parameterized test.

   b. Define the name of the variable and the data type. When creating parameters for a form, the parameter data type must match the field data type. For example, if you are creating a parameter to test a reference field on a form, you must create a parameter of type Reference. For more information on Now Platform data types, see Field types.

   c. Click Submit.

What to do next
Add a parameter to a test step
Add a parameter to a test step

Add a variable to a test step to hold a particular type of data when the test runs.

Before you begin

• Create a parameterized test
• Role required: atf_test_admin, atf_test_designer, or admin

About this task

⚠️ Note: The Run Server Side Script test step is not supported in parameterized tests.

Procedure

1. Navigate to Automated Test Framework > Tests.
2. Open a parameterized test.
3. In the Test Steps related list, create a test step or open an existing step.
4. Click to open a list of available parameters. Select a parameter to add it to a field. The parameter displays in the field.

Results

When the test runs, Automated Test Framework replaces the parameter with test run data.

What to do next

Add parameterized data sets.

Add parameterized data sets

Add or import test data to specify parameter runtime values.
Before you begin

- Create a parameterized test
- Add a parameter to a test step
- Role required: atf_test_admin, atf_test_designer, or admin

About this task

A data set, also known as a test run data set, includes runtime data used when the test runs. You can set a value for every parameter available to the current test. Data sets specify the parameter value during test runs.

Test designers can add data sets manually, or import data from a file.

Note: If you create a parameterized test that includes Custom UI test steps, the system only uses the first data set to retrieve components.

Procedure

1. Navigate to Automated Test Framework > Tests.
2. Open a parameterized test.
3. In the Test Run Data Sets related list, add or import data sets.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
</table>
| Add    | The New Parameter Value Set form opens. This form includes all parameters available to the current test.  
  
a. Define the Order field to set the order in which data sets run.  
b. Specify the parameter value when the test runs.  
c. Click Submit. |
| Import | The Import Test Run Data Sets form opens.  
a. Define the import behavior.  
  
  • Add: Adds the data in the file to existing test run data sets in the Test Run Data Sets table. For more information, see Add a record in the template.  
  
  • Replace: Downloads a template that includes any existing data sets in the Test Run Data Sets table. Replaces all existing da-
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a.</td>
<td>Data sets that have matching Sys IDs in the Test Run Data Sets table with data from the uploaded file. For more information, see Update a record in the template.</td>
</tr>
<tr>
<td>b.</td>
<td>In the Upload Test Run Data File section, select <strong>Yes</strong> to download an Excel template with the required fields, then upload a completed file. Alternately, select <strong>No</strong> to upload a completed file. Each row in the file becomes an entry in the Test Run Data Sets table.</td>
</tr>
</tbody>
</table>

**What to do next**

Click **Run Test**. Automated Test Framework runs each parameterized test once per data set, using the same test steps and execution order. For example, if a parameterized test has five data sets, Automated Test Framework runs the test five times, once for each data set.

Parameterized tests display test results by the execution order of the data sets. Open each test result record to view the test details.

**Allowed client errors**

Add known client errors to the allowed client errors list to allow tests and steps to continue running when a specific error occurs. Set the report level to specify what the Automated Test Framework does when the error occurs in future tests.

**Scenarios for allowing client errors**

Test designers and developers typically allow client-side JavaScript errors to prevent certain types of known failures from impacting test design and results. Scenarios to allow client errors include:

- **Timing constraints**
  
  Temporarily allow a client error until your developers have time to investigate and resolve the issue. For example, when testing an old form containing a longstanding bug.

- **Minimizing the impact of old libraries**
  
  Ignore client errors that cannot be fixed or are unimportant to your operations to eliminate their impact on future test runs. For example, when you find a bug in an old library.

- **Test design time**
  
  Temporarily allow client errors until you finish writing tests and have time to investigate the error. For example, one of your developers modifies a UI policy and the change generates an error.
Possible platform bug

Temporarily allow client errors until a fix is available. Prior to reporting a platform bug to ServiceNow Technical Support, investigate the error, verify it is not a customization error, and identify the type of platform bug involved. For example, a UI policy generates an error during a test. Your investigation verifies that the issue is not a customization error and identifies a platform issue with the UI policy.

Note: While adding client errors to the allowed client errors list allows the test framework to continue testing, it does not guarantee that your tests pass. Test designers and developers should always investigate client errors to determine if there are issues with your business process. For more details, see Identify and resolve client errors.

Report levels for allowed client errors

The report level indicates whether the test framework reports future occurrences of the error as a warning or ignores them altogether. You can change the report level of an allowed error at any time. For example, if you originally add an error as a warning, you can later change the report level to ignored.

<table>
<thead>
<tr>
<th>Report level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Warning</td>
<td>Test steps containing the allowed client error report a status of <strong>Success with warning(s)</strong>. The error message appears in the test result output, and is recorded in the test logs with the status <strong>Warning</strong>.</td>
</tr>
<tr>
<td>Ignored</td>
<td>Test steps containing the allowed client error report a status of <strong>Success</strong>. The error is recorded in the test logs with an <strong>Ignored</strong> status.</td>
</tr>
</tbody>
</table>

Matching process

The Automated Test Framework identifies allowed client errors using a contains search rather than an exact string match. A match occurs when a client error contains a message from an Allowed Client Error [sys_atf_whitelist] record. For example, if you create an Allowed Client Error record for the error message "Test message" with a report level of Ignored, then any client error containing this string is ignored.

Note: When you create or modify an Allowed Client Error record, the Client Test Runner automatically gets the update.
Identifying and resolving client errors

When client errors occur, the Automated Test Framework fails the test on the step that was executing when the error occurred.

Allow client errors from test results

Allow client errors as you review test results.

Before you begin
Role required: atf_test_admin, atf_test_designer, or admin

About this task
You can allow multiple or individual client errors. For each client error, you must decide how to report future instances of the client error. Report level options include:

• Warning: Test steps containing the allowed client error report a status of Success with warning(s). The error message appears in the test result output, and is recorded in the test logs with the status Warning.

• Ignored: Test steps containing the allowed client error report a status of Success. The error is recorded in the test logs with an Ignored status.

Procedure
1. Navigate to Automated Test Framework > Test Results.
2. Select the test result for a specific test.
   The system displays the Test Result record.
3. From the Related links, select one of the following options:
   • Add all client errors to warning list: Allow all client errors in this test with a report level of Warning.
   • Add all client errors to ignored list: Allow all client errors in this test with a report level of Ignored.
   The Automated Test Framework allows the selected client errors and displays a status message at the top of the form.

Related information

Allowed client errors

Allow client errors from step results
Allow client errors as you review step results.

Before you begin
Role required: atf_test_admin, atf_test_designer, or admin
About this task
You can allow multiple or individual client errors. For each client error, you must decide how to report future instances of the client error. Report level options include:

- **Warning**: Test steps containing the allowed client error report a status of Success with warning(s). The error message appears in the test result output, and is recorded in the test logs with the status Warning.

- **Ignored**: Test steps containing the allowed client error report a status of Success. The error is recorded in the test logs with an Ignored status.

Procedure
1. Navigate to **Automated Test Framework > Test Results**.
2. Select the test result for a specific test. The system displays the Test Result record.
3. Select the client errors to be allowed.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Specific step result from Step Results related list** | a. In the Step Results related list, right-click the step result containing client errors you want to allow.  
b. From the context menu, select one of the following options:  
  - **Add all client errors to warning list**: Allow all client errors in this step with a report level of Warning.  
  - **Add all client errors to ignored list**: Allow all client errors in this step with a report level of Ignored. |
| **Multiple step results from Step Results related list** | a. In the Step Results related list, select the check box in the first column for each step result containing client errors you want to allow.  
b. From the **Actions on selected rows** list, select one of the following options:  
  - **Add all client errors to warning list**: Allow all client errors in this step with a report level of Warning.  
  - **Add all client errors to ignored list**: Allow all client errors in this step with a report level of Ignored. |
| **Specific step result from Step Result record** | a. In the Step Results related list, select the Step Result record containing a client error you want to allow.  
b. From the Related links, select one of the following options: |
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Add all client errors to warning list</td>
<td>Allow all client errors in this step with a report level of Warning.</td>
</tr>
<tr>
<td>• Add all client errors to ignored list</td>
<td>Allow all client errors in this step with a report level of Ignored.</td>
</tr>
</tbody>
</table>

The Automated Test Framework allows the selected client errors and displays a status message at the top of the form.

**Related reference**

**Allowed client error records**

**Allow client errors from the test logs**

Allow client errors as you review test logs.

**Before you begin**

Role required: atf_test_admin, atf_test_designer, or admin

**About this task**

You can allow multiple or individual client errors. For each client error, you must decide how to report future instances of the client error. Report level options include:

• **Warning**: Test steps containing the allowed client error report a status of Success with warning(s). The error message appears in the test result output, and is recorded in the test logs with the status Warning.

• **Ignored**: Test steps containing the allowed client error report a status of Success. The error is recorded in the test logs with an Ignored status.

**Procedure**

1. Navigate to **Automated Test Framework > Test Results**.

2. Select the test result for a specific test. The system displays the Test Result record.

3. Select the client errors to be allowed.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specific test log from the Test Log related list</td>
<td>a. In the Test Log related list, right-click the test log client error you want to allow.</td>
</tr>
<tr>
<td></td>
<td>b. From the context menu, select one of the following options:</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>• Add client error to ignored list</td>
<td>Allow the client error with a report level of Ignored.</td>
</tr>
<tr>
<td>• Add client error to warning list</td>
<td>Allow the client error with a report level of Warning.</td>
</tr>
</tbody>
</table>

**Multiple test logs from the Test Log related list**

a. In the Test Log related list, select the check box in the first column for each test log client error you want to allow.

b. From the Actions on selected rows list, select one of the following options:
   - • Add client error to ignored list: Allow the client error with a report level of Ignored.
   - • Add client error to warning list: Allow the client error with a report level of Warning.

**Specific test log from Test Result Item record**

a. In the Test Log related list, select the test log (Test Result Item record) containing a client error you want to allow.

b. From the Related links, select one of the following options:
   - • Add all client errors to warning list: Allow all client errors in this step with a report level of Warning.
   - • Add all client errors to ignored list: Allow all client errors in this step with a report level of Ignored.

The Automated Test Framework allows the selected client errors and displays a status message at the top of the form.

**Related reference**

- Allowed client error records

**Manually allow client errors**

Manually create allowed client error entries as needed in the Allowed Client Errors table.

**Before you begin**

Role required: atf_admin, atf_test_designer, or admin

**About this task**

You normally allow client errors directly from a test result, step result, or test log, but you can manually allow them.
Procedure
1. Navigate to Automated Test Framework > Run > Allowed Client Errors.
2. Click New.
3. On the form, fill in the fields.

### Allowed Client Error form

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report level</td>
<td>Report action to take when the client error is encountered. Options include:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Warning – Step &amp; Test will report Success with Warning(s):</strong> Test steps containing the allowed client error report a status of Success with warning(s). The error message appears in the test result output, and is recorded in the test logs with the status Warning.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Ignored – Step &amp; Test will report Success:</strong> Test steps containing the allowed client error report a status of Success. The error is recorded in the test logs with an Ignored status.</td>
</tr>
<tr>
<td>Active</td>
<td>Check box to enable or disable allowing a client error.</td>
</tr>
<tr>
<td>Error message</td>
<td>Message of the client error you want to allow.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the error you want to allow. If this client error was allowed from a test result, step result, or a test log, the test log description is copied into this field.</td>
</tr>
</tbody>
</table>

4. Click Submit.

**Related reference**

Allowed client error records

**View the progress of automated tests**

When an automated test is running, view its progress in the Run Test progress dialog.

**Before you begin**

The system must be running a test.
Role required: atf_test_admin or atf_test_designer
About this task
When you execute a test or test suite, the system automatically displays the Run Test progress dialog. If you close this dialog, you can re-display it from either the results list or the results page for the currently-running test or test suite.

Procedure
1. If necessary, navigate to the Test Results list or Test Results page for the currently-running test or test suite.
   - If the system is currently running a test suite, navigate to Automated Test Framework > Suite Results. If desired, you can click the row for the running test suite to view the Suite Results page for that test suite.
   - If the system is currently running a test that’s not part of a test suite, navigate to Automated Test Framework > Test Results. If desired, you can click the row for the running test to view the Test Results page for that test.

2. Display the Run Test dialog.
   - If you are currently viewing the Test Results page or the Suite Results page, click Show Progress under Related Links.
   - If you are currently viewing the Test Results list or Suite Results list, right-click the row for the running test or test suite, then click Show Progress.

Passing data from one automated test step to another
Some automated test steps create data that you can use as an input to a subsequent step.

You can pass data from one test step to another using input variables and output variables.

The term input variables is another name for the field values associated with a step. These values are input variables because they provide the input the step needs to accomplish its task. For example, the Open Form step has these input variables: Table, Record, and View.
Example test step showing input variables

<table>
<thead>
<tr>
<th>Description</th>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open the &quot;incident&quot; form with id &quot;[step 1: Record insert:Record]&quot;</td>
<td>Table: Incident[incident]</td>
</tr>
<tr>
<td>Record: Step 1: Record insert:Record</td>
<td>Document_id</td>
</tr>
</tbody>
</table>

Some step types also have output variables. These are the values that later steps in the same test can use as input. For example, the Record Insert step has an output variable called Record which contains the sys_id of the newly created record.

**Note:** Input data can also be passed to an automated test case from an external source such as a .csv file.

The test step form doesn’t indicate if a test step has output variables or not. You can easily tell if any output variables are available to provide a value to any given input variable. If you can map the value of one step’s output variable to the current step’s input variable, the system displays the mapping icon (🔗) to the right of that input field. When you click the variable mapping icon, the system displays a tree giving you access to any available output variables from previous steps.
Example test step showing output variables

For step-by-step instructions on how to assign the value of an output variable to another step's input variable, see Pass values from one automated test step to another.

For an example of a test that passes variables using input and output variables, see Automated Test Framework use case: reference a value from a previous step.

Pass values from one automated test step to another

Assign a form field the value of an output variable returned from a previous step.

Before you begin
You must have a previous test step that returns an appropriate output variable.

Role required: admin, atf_test_admin and atf_test_designer

Procedure
1. To the right of the field whose value you want to assign, click the input value icon.

   The input value mapping control lists previous steps that create output variables. If no previous steps create output variables, the control displays the message: There are no elements to show.

2. Click the row for the step that contains the output variable you want to use as an input.
3. Click the output variable you want to use.
   If the output variable is an id for a glide record, the control displays a tree
   picker providing access to fields for this record.

4. Navigate through the tree picker hierarchy until you find and select the value
   you want.

Related information

 Automated Test Framework use case: reference a value from a previous step

Building and running automated test suites

Run a group of tests in a specific order to test an application or a group of
related features.

A test suite can contain both individual tests and other test suites. A test suite
that contains another test suite is called a parent, and the test suite contained
within the parent is called a child. While a test suite can have both individual
tests and test suites as children, tests cannot have other tests as children. Tests
can only contain test steps.

Benefits

Grouping tests into test suites offers these benefits.

• Allows testers to run every test in a test suite with one action.
• Allows testers to run all child test suites in a parent test suite.
• Allows testers to see test results for every test in a test suite.
• Allows test designers and testers to schedule when to run test suites.
• Allows test designers and testers to schedule starting client test runners to
  support test runs.

Hierarchies

Automated Test Framework supports building a multi-level hierarchies where a
test suite can be both a parent and child. For example, this figure illustrates Test
Suite 1 as the parent at the top of the hierarchy. Test Suite 1.3 is a child of Test
Suite 1 and also a parent of Test Suite 1.3.1 and Test Suite 1.3.2.
Schedules
To schedule a test suite, you need three components:

- a test suite record
- a schedule record specifying when you want the system to run the test suite
- a scheduled suite run record that associates the test suite to run with the schedule for running it

With this model, you can associate a schedule with many different test suites, and vice versa.

⚠ Note: You can schedule only test suites, not individual tests. Scheduled tests will run only if there is an open Scheduled Client Test Runner page matching the scheduled suite’s browser conditions. Scheduled tests cannot run on a machine that is locked, powered down, or does not already have the browser open.

The watchlist on the test suite run record also allows you to specify users to receive an email when the system finishes executing the test suite run.

If the test suite contains one or more form steps (steps involving a user interface), you must ensure that a scheduled client test runner is actively running in a browser when the schedule triggers the suite run.
Filters
Automate the creation of test suites by using a filter to dynamically add tests to a test suite when they match the filter conditions. Reduce the time that your test designers spend manually creating and maintaining test suites.

Related reference
Step results record
Related information
Suites
Create an automated test suite
Group automated tests into a suite you can execute as a batch.

Before you begin
The tests you want to include in the test suite must exist.
Role required: atf_test_admin or atf_test_designer

Procedure
1. Navigate to Automated Test Framework > Suites.
2. Click New.
   The system displays the Test Suite New Record form.
3. In the Name field, enter a name for this suite.
4. Select the tests to be included in this suite.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter</td>
<td>Add tests dynamically using a filter:</td>
</tr>
<tr>
<td></td>
<td>a. In the Filter field, use the condition builder to create the conditions a test must match for inclusion in the test suite.</td>
</tr>
<tr>
<td></td>
<td>b. Click Save.</td>
</tr>
<tr>
<td></td>
<td>All tests that match the filter conditions appear in the Test Suite Tests related list. Because the suite is dynamic, any</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>new test that matches the conditions is automatically added to the suite.</td>
<td></td>
</tr>
<tr>
<td>Add tests manually:</td>
<td></td>
</tr>
<tr>
<td>a. In the Test Suite Tests related list, click Insert a new row....</td>
<td></td>
</tr>
<tr>
<td>b. In the Test field, enter the name of the test to add to this test suite.</td>
<td></td>
</tr>
</tbody>
</table>

5. In the Description field, enter a description for this test suite.
6. In the Test Suite Tests related list, specify options for a test.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Enter a value to specify the order in which this test should execute within the test suite.</td>
</tr>
<tr>
<td>Abort on Failure</td>
<td>Enter a value that indicates whether you want the test suite to stop or continue if this test fails.</td>
</tr>
<tr>
<td></td>
<td>• By default, the system assigns the value false to this field. If this test fails, the system still executes any further tests in the test suite.</td>
</tr>
<tr>
<td></td>
<td>• Set the value to true to stop the test suite if this test fails.</td>
</tr>
</tbody>
</table>

7. Repeat steps 4 - 6 for every test you want to include in this test suite.
8. Click Submit.

**Copy an automated test suite**

Reduce time in creating tests by copying an entire test suite. Rename and modify the test suite after copying. The Copy Test Suite button on the Test Suite form copies all the nested tests and child test suites within the test suite.

**Before you begin**
Role required: atf_test_admin or atf_test_designer

**Procedure**
1. Navigate to Automated Test Framework > Suites.
2. Select the row containing the test suite that you want to copy.
   The Test Suite form opens.
3. Near the top-right corner of the Test Suite form, click Copy Test Suite.
A progress tracker appears to confirm the status. If you click **Go to Copy**, the system displays a new test suite record identical to the copied record, except for the **Name**.

**Note:** If you apply filter conditions, copied tests are added to the test suite if the conditions are met.

If any tests or child test suites fail to copy, they are skipped and copying of the remaining tests and child test suites continues. A warning message showing a partial success appears below the progress bar.

**Note:** You can copy a test suite only when its in the same scope as the current scope. The **Copy Test Suite** button appears only when the test suite is in the same scope. The scopes of the tests in a suite are preserved when copying a test suite. See **Application Scope** for more information.

If the copy operation fails, an error message appears below the progress bar.

4. In the **Name** field, enter the name that you want to assign to this new test suite.

5. Edit the tests and test suites within the copied test suite.

6. When you are finished making changes, click **Update**.

**Results**

A copy of the test suite along with all the nested tests and child test suites is created.
Add tests to a suite with a filter
Automate the creation of test suites by using a filter to dynamically add tests to a test suite when they match the filter conditions. Reduce the time that your test designers spend manually creating and maintaining test suites.

Before you begin
The tests you want to include in the test suite must exist.
Role required: atf_test_admin or atf_test_designer

Procedure
1. Navigate to Automated Test Framework > Suites.
2. Click New.
   The system displays the Test Suite New Record form.
3. In the Name field, enter a name for this suite.
4. For Filter, use the condition builder to specify the conditions a test must meet to be added to the test suite.
5. Click Save.

Results
All tests that match the filter conditions appear in the Test Suite Tests related list. Because the suite is dynamic, any new test that matches the conditions is automatically added to the suite.

Add test to an existing automated test suite
Add a test to a test suite that already exists.

Before you begin
The tests you want to include and the test suite must exist.
Role required: atf_test_admin or atf_test_designer

Procedure
1. Navigate to Automated Test Framework > Suites.
2. Click the row containing the test suite you want.
   The system displays the Test Suite form.
3. In the Test Suite Tests related list, click Insert a new row....
4. In the Test field, enter the name of the test to add to this test suite.
5. In the **Order** field for this row, enter a value to determine the order in which this test should execute within the test suite. By default, the system assigns a value to this field according to the order in which you add the tests.

6. In the **Abort on Failure** field for this row, enter a value that indicates whether you want the test suite to stop or continue if this individual test fails. By default, the system assigns the value **false** to this field. **False** means that if this test fails, the system still executes any further tests in the test suite.

7. Repeat steps 3 - 6 for every test you want to include in this test suite.

8. Click **Submit**.

**Add child test suite to parent test suite**

Add to a multi-level test suite by including a child test suite within a parent test suite.

**Before you begin**
The parent test suite must exist.
Role required: atf_test_admin or atf_test_designer

**Procedure**

1. Navigate to **Automated Test Framework > Suites**.

2. If the child test suite exists, open the child test suite form for editing. If the child test suite does not yet exist, create it. In both cases, leave the test suite form open.

3. In the **Parent suite** field, enter the name of the test suite you want to act as the parent to this child.

4. If desired, add one or more tests to the child test suite.

5. Click **Submit**.

**Run an automated test suite**

After creating an automated test suite, run it in a non-production instance.

**Before you begin**
You must have created the test suite you want to run.
The **test execution property** must be enabled. You must have an admin or atf_test_admin role to do so.
**Note:** The test execution property is disabled by default to prevent running tests on a production system. Run tests only on development, test, and other sub-production instances.

Role required: atf_test_admin, atf_test_designer, or admin

**About this task**
This procedure outlines how to start a test suite manually. You can also schedule test suites to run at a later time. For more information, see Working with scheduled test suites.

**Procedure**
1. Navigate to **Automated Test Framework > Suites**.
2. If necessary to view the Test Suites list, click **Test Suites**.
3. Click the row containing the test suite you want to run. The system displays the **Test Suite** form.
4. Click **Run Test Suite**.

   **Note:** If the test execution property is not enabled, the **Run Suite** button does not appear. In this case, see the annotation at the top of the form, and click the link to enable running tests.

5. If the tests associated with the test suite include a form step (any step involving a UI), or other kinds of UI test steps, the **Pick a Browser** dialog appears before executing the tests. Use it to choose among any currently-running test clients, or start a new runner. For more information, review **Browser recommendations for all tests and suites**.
   If the tests associated with the test suite only include server test steps, the system executes the tests without displaying the Pick a Browser dialog.

**What to do next**
Monitor the progress of the tests. When complete, click **Go to results** on the progress dialog window to display the **Test Results** list, where you can view and analyze the results.

**Schedule an automated test suite**
Schedule one or more test suites to run at a specific date and time.

**Before you begin**
You must have created the test suites you want to schedule.
Role required: atf_test_admin or atf_test_designer
About this task
To schedule a test suite, you need three components:

- a test suite record
- a schedule record specifying when you want the system to run the test suite
- a scheduled suite run record that associates the test suite to run with the schedule for running it

For more information about the capabilities of and requirements for scheduled test suites, see Working with scheduled test suites.

Procedure

1. Navigate to Automated Test Framework > Schedules.
   The system displays the list of existing test suite schedules.

2. To create a new schedule, click New.
   The system displays the Suite Schedule record form.

3. On the Suite Schedule record form, enter the name of the schedule, the frequency with which to run associated suites, the time at which to run associated suites, and the timezone for this schedule.

4. Optional: To specify a condition that must be met for running associated test suites, check Conditional, then fill in the Condition text box with the appropriate script.

5. To add a test suite to run, navigate to the Scheduled Suites related list, then click New.
   The system displays Scheduled Suite Run record form.

6. On the Scheduled Suite Run record form enter the appropriate data.
   a. Enter the test suite to run.
   b. If the suite contains UI steps, enter any client constraints you wish to apply (such as browser to use). For more information on client constraints, see Scheduled Suite Run.
   c. Add to the record's watchlist users you want the system to inform (by email) when the scheduled suite has finished.
   d. Click Submit.

7. Optional: To add more test suites to this schedule, repeat steps 5 and 6.

8. Click Update.

9. If your suite includes any test steps that work with a form – or any other element on the client side – follow these steps to open a browser window for running the client portion of the scheduled tests.
a. Review **Browser recommendations and requirements** for all tests and suites, as well as those that apply only to scheduled suite runs.

b. In the **Navigator**, right-click **Scheduled Client Test Runner**, and then click the option to open in a separate tab or window, as you prefer.

c. Leave open the browser window that’s running the client test runner and return to the browser window that contains the **Navigator**.

### Run a scheduled test suite using a script

Execute a scheduled UI test suite immediately using a script without having to wait for the scheduled time. You can use this method while trying to automate the process of running a test.

**Before you begin**

You’ve created and scheduled the test suites that you want to run. See [Create an automated test suite](#) and [Schedule an automated test suite](#), for more information.

Role required: **atf_test_admin** or **atf_test_designer**

**About this task**

The following steps might not be in line with your software configurations.

**Procedure**

1. Spin up a virtual machine (VM) on an operating system with the necessary browsers.

2. Open a browser on the instance and navigate to the Scheduled Client Test Runner.

3. Call the scriptable method `new sn_atf.ScheduledRunsExecutor().setScheduleSysId("SYS_ATF_SCHEDULE_SYS_ID_HERE").start();` to run the scheduled suite immediately.

   **Note:** The `start()` method returns `sys_progress_worker.sys_id` of the progress worker.

   To run a test only when the script is called, set the Run field to **On Demand** in the Schedule form.

**Re-run failed tests in an automated test suite**

Re-run failed tests within a test suite without rerunning the entire suite.
Before you begin
Role required: atf_test_admin, atf_test_designer, or admin

About this task
The **Re-run failed tests** button appears on the Suite Result form and on the Suite Execution Progress Viewer after a suite with failed tests completes. It does not appear if test execution is disabled, the suite is deactivated, the suite passed, or the user does not have one of the required roles.

**Re-run failed tests** button actually re-runs all non-passing tests. This includes tests with the following status: canceled, skipped, failure, and error. It does not include the test results with a *Success with warning(s)* status.

Procedure
1. Navigate to either the suite result form or suite execution progress viewer for the completed suite that had failed tests.
2. Click **Re-run failed tests**.
3. If the re-run tests include a form step (any step involving a UI), or other kinds of **UI test steps**, the **Pick a Browser** dialog appears before executing the tests. Use it to choose among any currently running test clients, or start a new runner. For more information, review **Browser recommendations for all tests and suites**. If the re-run tests include only **server test steps**, the system executes the tests without displaying the Pick a Browser dialog.

Results
The system re-runs the failed tests:

- The system creates a new suite result hierarchy for the re-run tests. The Progress Workers, Test Result, and Test Suite Result forms show the same suite hierarchy as the previous test suite. They do not include the tests or suites that passed in a previous run.
- If you delete or deactivate a child suite or test that failed, and then re-run it, the system does not execute that suite or test in the re-run.
- If you add a child suite or test to the suite to a failed test, and then re-run it, the system does not execute the added suite or test in the re-run.

What to do next
To view the results from the previous run of a test or suite, click **Previous test result** on the test result form or **Previous suite result** on the test suite result form. These fields only appear for tests and suites that have been re-run.
Parallel testing
Reduce test design time by running multiple tests and test suites in parallel. Design tests to run in parallel by avoiding resource conflicts and data dependencies.

⚠️ Note: If two or more users are developing tests simultaneously, parallel testing reduces test design time. After test design completes, it is recommended to organize tests into a single hierarchical suite structure, and run the tests as a single base suite.

Parallel testing limit
Parallel testing enables users to run multiple automated tests simultaneously. This process continues until the number of parallel running tests is as per the following formula.

\[
\text{Number of parallel tests} = \max(1, \text{number of worker threads} - 2)
\]

⚠️ Note: If your instance has 2 or less worker threads, configuration improvements review is recommended.

The actual number of parallel tests that a non-production instance can support depends on the system resources that the instance has when it is provisioned. The parallel testing limit ensures that an instance always has system resources available for other non-testing tasks.

Test waiting queue
When the system reaches the parallel testing limit, it reschedules tests to run later. It automatically places the tests back in sys_trigger until a worker thread is available to pick them up. Each test in the waiting queue has a schedule the next time the test runs.

Design considerations
Run multiple tests and test suites in parallel to reduce test design time. Avoid resource conflicts and data dependencies by designing parallel running tests. Avoid resource conflicts and data dependencies by designing tests that rely only on newly generated or self-created data, or have mutual exclusion rules defined between tests that share resources.

 Prevent resource conflicts between parallel tests
Prevent resource conflicts by running tests that create their own data. Tests that run with existing data prevent other tests that need the same data from running in parallel.
Note: If you have two or more tests with resource conflicts, see Mark tests as mutually exclusive to create a mutual exclusion rule that prevents the tests from running in parallel.

Mutually exclusive tests

Prevent conflicting tests from running in parallel by marking them as mutually exclusive. For example, when the system identifies tests that modify the same record, the system makes these tests mutually exclusive. You can also manually mark tests as mutually exclusive.

The system marks tests as mutually exclusive when there is a potential resource conflict. When the system can't detect resource conflicts automatically, you can create your own mutual exclusion rules that can prevent conflicting tests from running in parallel. For example, if a test changes a sys_properties record, the record shows up under Records Modified of that test. If the validation path of another test depends on the same sys_properties record without any change, that test fails. This can occur if the previous test runs at the same time.

You can view mutually exclusive tests on the Mutually Exclusive Tests related list in the test form. This related list shows all tests that don't run in parallel with the current test and the reason. The same test might appear more than once in the list if there are multiple reasons.

Mutually exclusive tests reasons
Tests are marked mutually exclusive for the following reasons.
• Two or more tests modify the same record.
• A test that runs in parallel with itself.
• You can create your own mutual exclusion rules when the system can’t detect resource conflicts automatically.

Mark tests as mutually exclusive
You can mark tests as mutually exclusive using any of the following methods.

Mutually Exclusive Tests tab
Select a test from the Tests list and navigate to Mutually Exclusive Tests > Add Mutual Exclusion to make the selected test mutually exclusive with another test.

Tests list
Select one or more tests from the Tests list and choose Add mutually exclusive test from the Action on selected rows context menu. Enter one test in the Add mutually exclusive test dialog box to make the selected tests mutually exclusive with the current test.

Parallel Test Runs tab
When two or more tests run in parallel, navigate to Test Results > Parallel Test Runs. Select one or more tests and choose Add mutually exclusive test from the Action on selected rows context menu to mark the selected tests as mutually exclusive.

Note: The Parallel Test Runs tab is visible only if the test runs in parallel with one or more tests.

Automated Test Framework design considerations
Create reliable, scalable, and efficient tests by following these design considerations.

General testing
Avoid modifying ServiceNow system tables or tables extending the Application File [sys_metadata] that can potentially change the behavior of the system. Avoid using or modifying any existing records to prevent unexpected results between tests. The following are some of the common examples of system data changes that can cause unexpected results.
• Impersonate an existing account
• Delete an existing record.
• Run a test that disables a business rule or system property
• Validate with an existing record

Parallel testing
Reduce test design time by running multiple tests and test suites in parallel. Design tests to run in parallel by avoiding resource conflicts and data dependencies.

Prevent resource conflicts between parallel tests
Prevent resource conflicts by running tests that create their own data. Tests that run with existing data prevent other tests that need the same data from running in parallel.

Note: If you have two or more resource conflicting tests, see Mark tests as mutually exclusive to create a mutual exclusion rule that prevents them from running in parallel.

Parameterized testing
Run a test multiple times with different test data for each run. Create parameters to store test data for each test run. See Parameterized test components for more information.

• Create parameters to store test data for each test run.
• Ensure that the parameterized tests support standard Automated Test Framework (ATF) features, such as reports, test suites, and data rollback. Copying a parameterized test copies all parameters, test run data sets, and test steps.

Note: If a parameterized test including Custom UI test steps is created, the system only uses the first data set to retrieve components.

Custom UI testing
Test customized user interfaces such as UI pages and UI macros by retrieving their HTML and JavaScript page components and identifying the test actions they support.

Use the page inspector to identify testable page components
The page inspector determines which page components are available for custom UI testing. Page components that are unavailable to the page inspector are unavailable to custom UI testing.

Navigate to the custom UI you want to test
Use existing test steps to navigate to the target custom UI. For example, to test a Knowledge Base article, use the existing test steps to navigate to a module or to open an existing record. Most custom UI testing requires using existing test step categories as part of the test.

**Use the component area to identify page components**

The component area describes the HTML layout element containing the component such as a `<div>` or `<section>` element. The area helps test designers distinguish between components by providing the location in the page layout.

**Test your custom UI rather than Now Platform UI**

The Automated Test Framework prevents custom UI testing of Now Platform features. For example, you cannot test dashboards or graphical designers. Instead, build tests to validate your custom UI pages and elements because you have direct control over these user interfaces.

**Use HTML attributes to override page component testing properties**

Change the testing properties of a particular page component using Automated Test Framework-specific HTML attributes. See [Override component test actions](#).

**Retrieve page components again when you move tests to another instance**

Custom UI test steps don’t store UI components as metadata. Testers must manually retrieve page components again when moving tests between instances.

**Clone tests from production system**

Move your tests to the production system to clone the most updated instances for testing. Speed up the testing time by directly copying or cloning a test from the production system to a subproduction instance.

**Note:** By default, the system property that is used to run automated tests is disabled to prevent you from accidentally running these tests on a production system. To avoid data corruption or an outage, run tests only on development, test, and other non-production instances.
## Warning messages for all testing

<table>
<thead>
<tr>
<th>Warning messages</th>
<th>Design considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Impersonating an existing user may cause unexpected behavior for this test. Avoid potential issues by adding a 'Create a User' step instead. See the documentation for Test Design Considerations.</td>
<td>Create a new user to ensure proper roles and groups and avoid using existing records. See General testing for more information.</td>
</tr>
<tr>
<td>Using a table that extends Application File [sys_metadata] may cause unexpected behavior for other tests running in parallel. See the documentation for Test Design Considerations.</td>
<td>Avoid running a test with a table that extends the Application File because it might affect other tests. See Parallel testing for more information.</td>
</tr>
<tr>
<td>Using a system table may cause unexpected behavior for other tests running in parallel. See the documentation for Test Design Considerations.</td>
<td>Avoid using a system table because it might affect other tests running in parallel. See Parallel testing for more information.</td>
</tr>
<tr>
<td>Using an existing record may cause unexpected behavior for this test. See the documentation for Test Design Considerations.</td>
<td>Avoid using existing records because these records might not have the state and values as expected by the test. Use records created during the test to ensure proper state and values. See General testing for more information.</td>
</tr>
<tr>
<td>Modifying an existing record may cause unexpected behavior for other tests running in parallel. See the documentation for Test Design Considerations.</td>
<td>Avoid using existing records because it might affect other tests. Use records created during the test. See General testing for more information.</td>
</tr>
<tr>
<td>Using assert type '--None--' may cause unexpected behavior for server UI actions. Avoid potential issues by setting the assert type and using a timeout. See the documentation for Test Design Considerations.</td>
<td>Server UI actions cause the current form to submit and the page to reload. Select an assert type other than None to avoid any unexpected behavior for server UI actions. Set a timeout to ensure that your test waits for the form to be submitted or not submitted before moving on to the next step. When testing server</td>
</tr>
</tbody>
</table>
### Cancelling automated tests and test suites

You can cancel automated tests and automated test suites that are running or are queued to run.

How you cancel an automated test or automated test suite depends on whether the test or test suite is currently running or is queued to run.

#### Cancel queued automated test suite

You can cancel an automated test suite that is queued but has not yet run.

**Before you begin**
Role required: atf_test_admin or atf_test_designer

**Procedure**

1. Navigate to **Automated Test Framework > Run > Waiting/Running Test Runs**.
2. Select the action check boxes for the test suites to cancel.
3. From the action choice list, select **Delete**.

#### Cancel running and pending tests in an automated test suite

Cancel running and pending tests in a running suite test.

**Before you begin**
Role required: atf_test_admin or atf_test_designer

**Procedure**

1. If necessary, display the Run Test progress dialog.
2. Click **Cancel Pending Steps**.
   
   The system displays a dialog asking you to confirm that you want to cancel this test execution.
3. In the Confirmation dialog, click **Yes**.

---

**Warning messages**

<table>
<thead>
<tr>
<th>Design considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>UI actions, the <strong>None</strong> assert type configures automatically to <strong>Form submitted to server</strong>.</td>
</tr>
</tbody>
</table>

---

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**Results**
The system cancels any running and pending tests in this test suite and rolls back any data that tests in the test suite may have changed.

**Cancel running automated test**
You can cancel a running test from the Run Test progress dialog.

**Before you begin**
The system must be running a test.
Role required: atf_test_admin or atf_test_designer

**Procedure**
1. If necessary, display the Run Test progress dialog.
2. Click **Cancel Pending Steps**. The system displays a dialog asking you to confirm that you want to cancel this test execution.
3. In the Confirmation dialog, click **Yes**.

**Results**
The system cancels the running test and rolls back any data changes the test made.

**Cancel waiting automated test**
You can cancel a waiting automated test from the Waiting/Running Test Runs module.

**Before you begin**
Role required: atf_test_admin or atf_test_designer

**Procedure**
1. Navigate to **Automated Test Framework > Run > Waiting/Running Test Runs**.
2. Select the action check boxes for the tests to cancel.
3. From the action choice list, select **Delete**.

**Administering the Automated Test Framework**
Enable or disable the Automated Test Framework, modify retention policies, move tests between instances, control user access to the Automated Test Framework, and create custom test step configurations and step environments.
For details about individual properties that control how the Automated Test Framework works, see Properties.

Creating custom test step configurations

Step configuration records (or step configs) define how each step type behaves. You can create new step configurations that define custom steps that run on the server.

When you add a step to an automated test, that step has a defined type such as Set Field Values, or Record Insert. Each step configuration has a number of characteristics that affect how steps of that type behave, including the inputs required, the actions performed, and so on.

These characteristics are defined in the step’s step configuration record.
The Step execution script field determines the actions the system executes when a step with this config runs.

Related information

- Step configurations
- Step execution scripts

Create custom step configuration

Create a custom step configuration that can form the basis of new steps that run on the server.

Before you begin

Role required: atf_test_admin

About this task

The Automated Test Framework includes specific types of steps such as Open Form, Set Value, Assert Value and so forth. With the Step Configurations module, you can create steps that perform actions you specify. You can only create configurations for steps that run on the Server. You cannot create configurations for steps that run on the browser.

Procedure

1. Navigate to Automated Test Framework > Administration > Step Configurations.
2. Click New.
   The system displays the Test Step Config form.
3. In the Name field, enter a name for your step type.
4. Leave Active checked.
5. Leave Step environment set to Server-Independent. You can only define step configurations that run on the server; you cannot define step configurations that run in the browser.
6. Optional: In the Category field, select the category to which you want to assign this step. Categories are used for filtering the step list in the Add Step dialog. For more information, see Category field example.
7. Optional: In the Batch Order Constraint field, choose one of the following values.
   - None: A step based on this configuration can appear at any point in a test.
   - Start Batch Execution: If this test includes a batch with this step, this step must be the first in the batch.
• **Run in the middle of an execution**: If a test includes a batch with this step, this step must appear after the first and before the last step of the batch.

• **Stop Execution**: If a test includes a batch with this step, this step must be the last step in the batch.

8. In the **Order** field, enter an integer specifying where steps with this configuration appear in the step list on the Add Test Step dialog. For more information, see the example using the Order field.

9. In the **Template reminder** field, enter the instructions you want to appear when this step is included in a test as part of a template. For more information, see the example of using the Template reminder field.

10. In the **HTML description** field, enter the text you want to appear when the cursor highlights this step on the Create New Step dialog. For more information, see the example using the HTML description field.

**Note:** You may find it convenient to skip the next two steps, both of which involve writing scripts, until after you have added any input and output variables.

11. In the **Description generation script** field, add code to the provided template to generate the description assigned to a Test Step record when a step of this type is included in a test. For more information about writing this script, see Step description generation script. To see an example of where the system displays this description, see Description generation script example.

12. In the **Step Execution Script** field, add code to the provided template to define the script that executes when a step of this type runs. The script template provides instructions and examples for working with step inputs, outputs, and step results. For more details on the step execution script, see Step execution scripts.

13. Click **Submit**. The system creates a new test step configuration and returns to the list of test configurations.

14. **Optional**: To add input or output variables, re-open the step config record.

15. **Optional**: To add input variables to this step config, scroll to the Input Variables tab, then click **New**. Fill out the required fields for the new variable, then click **Submit**. Repeat until you have added all the input variables needed.
16. Optional: To control the order in which input variables appear on the New Step form, edit the values in the Order column for the Input Variables related list.

17. To add output variables to this step config, scroll to the Output Variables tab, then click New. Fill out the required fields for the new variable, then click Submit. Repeat until you have added all the output variables needed.

Related reference
Automated Test Framework Step Config record

Related information
Step execution scripts
Step configurations
Add a predefined list of steps (template) to an automated test

Create a custom step configuration category
Create a custom step config category.

Before you begin
Role required: att_test_admin

About this task
Categories are used for filtering the step list in the Add Step dialog. For more information, see Category field example.

Procedure
1. Navigate to Automated Test Framework > Administration > Step Configuration Categories.
2. Click New.
   The system shows the Test Step Config Category form.
3. In the Name field, enter a name for your step category.
4. In the Step Environment field, enter the step environment in which steps under this category execute:
   - Server - Independent, if you want this category to contain steps that execute on the Server.
   - UI, if you want this category to contain steps that execute on browser.
   - Server - REST, if you want this category to contain steps that send Inbound REST messages to the instance.
5. In the **Display name** field, enter the category name you want to appear in the middle column of the Add Test Step dialog when this category is selected.

**Example**

![Add Test Step dialog](image)

6. Click **Submit**. The system creates a test step category and returns to the list of test step environments.

**Working with test step templates**

Test step templates contain a list of steps to be added all at once to an automated test.

**Related information**

- Add a predefined list of steps (template) to an automated test

**Create an automated test steps template**

Reduce testing time by creating a template containing a list of steps to add all at once to an automated test.

**Before you begin**

Role required: atf_test_admin

**About this task**

Many tests follow similar patterns. One common pattern, for example, is to open a form, set some field values, validate some field values, click a UI action,
open a record producer, open a catalog item, and submit the current form. If a template exists containing these steps, you can add them to a test all at once. The Automated Test Framework comes with default templates in the base system. With this procedure, you create your own templates.

Procedure

1. Navigate to **Automated Test Framework > Administration > Test Templates**. The system displays the Test Templates list.

2. Click **New**. The system displays the Test Template form.

3. In the **Name** field, enter a name for your template.

4. In the **Test Template** field, click the lock icon (🔒). The Test Template field unlocks and expands to allow editing.

5. In the **Test Template** field, enter the name of the first test step to add to this template. The system adds the test step to the Test template list.

6. Continue adding test steps – in the order that you want them to appear – until you’ve added all the steps that you want to include in the list.

   - **Note:** The test template supports tables, catalog items, and the record producer.

7. In the **Description** field, enter a description for this template.

8. Click **Submit**.

Related information

Add a predefined list of steps (template) to an automated test

Edit automated test steps template

Edit an existing test template.

Before you begin

Role required: atf_test_admin

Procedure

1. Navigate to **Automated Test Framework > Administration > Test Templates**. The system displays the **Test Templates** list.

2. Click the row for the test template you want to edit. The system displays the Test Template form.
3. In the **Test Template** field, click the lock icon (잠금 아이콘). The Test Template field unlocks and expands to allow editing.

4. **Note:** You can delete steps anywhere in the list, but you can add steps only to the end of the list. You can re-order steps in a test after you add them using the template, but you cannot re-order steps in the template itself.

Add or delete steps to the template.

- To delete a step, select that step, then click the X icon.
- To add a step, in the **Test Template** field, enter the name of the step to add.

5. Continue adding and deleting test steps until it contains the steps you want.

6. Click **Submit**.

**Related information**

Add a predefined list of steps (template) to an automated test

**Enable or disable executing Automated Test Framework tests**

Allow or prevent tests and test suites from executing on this instance.

**Before you begin**

Role required: att_test_admin

**About this task**

By default, the system property that is used to run automated tests is disabled to prevent you from accidentally running these tests on a production system. To avoid data corruption or an outage, run tests only on development, test, and other non-production instances.

**Procedure**

1. Navigate to **Automated Test Framework > Administration > Properties**.
2. Set the test execution property.
   - To enable test and test suite execution, check **Enable test/test suite execution**.
   - To disable test and test suite execution, uncheck **Enable test/test suite execution**.
3. Click **Save**.
Modify data retention policy for ATF test results

Modify the Auto Flush data retention policy, which designates how long the system retains data, and referencing data, for test and test suite results. You can change the frequency of flushing for the sys_atf_test_result or sys_atf_test_suite_result base tables. This setting controls how far back in time test result data is available.

Before you begin
Role required: atf_test_admin

About this task
The system regularly flushes data in the sys_atf_test_result and sys_atf_test_suite_result base tables, (and optionally, referencing data). By default, the system deletes test and test suite results data 30 days after creation. This task enables you to modify the Auto Flush retention policy for data stored in a specific base table (sys_atf_test_result or sys_atf_test_suite_result).

Procedure
1. Navigate to Automated Test Framework > Administration > Table Cleanup. The system displays a list of the retention policies (Auto Flushes) it maintains for automated testing results tables.
2. Select the retention policy (sys_atf_test_result or sys_atf_test_suite_result) to modify. The system displays the record for this retention policy.
3. The Tablename field displays the name of the table to which the selected Auto Flush retention policy applies. Skip this field to accept the default.

Note: Selecting another tablename in this field compromises the integrity of the Auto Flush record. Leave the tablename on existing ATF policies at the base system (default) value so it does not adversely affect ATF data retention behavior.

4. Specify how the system should determine the length of time for retention of data, and referencing data.
   a. In the Matchfield field, enter the field you want the system to use to monitor duration. For example, to specify that you want to delete data x amount of time after the system created it, leave Matchfield set to its default value of sys_created_on.
   b. In the Age in seconds field, enter the amount of time (in seconds) the system must wait before deleting the associated data and referencing data.
5. If you want to apply the policy to the specified data (for example, sys_atf_test_result), AND any data that references it, select Cascade delete (default value).
Affected referencing data is stored in the following tables: sys_atf_test_result_item, sys_atf_test_result_step, and sys_attachment (when table_name = sys_atf_test_result). If you want the policy to simply flush data in the selected table (for example, sys_atf_test_result), and skip flushing of the referencing data, then clear Cascade delete.

6. In the Conditions field, specify the filter conditions to use for selection of data (and optionally, referencing data) for this Auto Flush retention policy. The default is Retain indefinitely is false, because the Test results record also contains a Retain indefinitely check box that allows opting out of the auto flushes for specific test results.

7. Click Update.

Manage status and retention policies for automated test client runners
Modify how often active client test runners report in to the system and how long the system retains records for inactive client test runners.

Before you begin
Role required: atf_test_admin

About this task
When you start a client test runner, the system registers that runner as active, meaning that it is either running a test or is available to run a test. While the runner is active, it reports in to the system at a specified interval. If the runner does not report in at the expected time, the system marks the runner as inactive. After a period of time the system deletes the runner. This task enables you to modify the Automated Test Framework properties that control these intervals.

Procedure
1. Navigate to Automated Test Framework > Administration > Properties.
2. Navigate to the Test Runner Properties section.
3. To set the interval at which active client test runners report in to the system, enter the reporting interval in seconds in the Test runner heartbeat interval field.
4. To set the period of time a test runner can remain inactive before the system deletes it, enter the period of time in seconds in the Test runner timeout field.
5. Click Save.
Related reference
- Active manual test runners
- Active scheduled test runners

Related information
- Working with client test runners

Move automated tests from one instance to another
Move automated tests from one instance to another using the normal process for update sets.

Before you begin
Role required: atf_test_admin

About this task
You can move automated tests, automated test suites, and related data using update sets. For more information, see System update sets.

Compare results and execution times for different automated test and suite results
You can compare execution times for different runs of an automated test or automated test suite. You can also compare results over time for a single automated test suite.

Before you begin
Role required: atf_test_admin or atf_test_designer

Compare execution times for different runs of an automated test
Compare how long it took the system to execute each test step across different runs of the same test.

Procedure
1. Navigate to Automated Test Framework > Tests.
2. Select the row for the test whose results you want to compare.
   The system displays the Test form.
3. Navigate to the Test Results related list.
4. Check the rows for the test results you want to compare.
5. From the Actions on selected rows control, click Compare test step results.
   The system displays the Compare test result execution times bar graph.
Compare execution times for different runs of the same automated test suite

Compare how long it took the system to execute each test across different runs of the same test suite.

Procedure
1. Navigate to Automated Test Framework > Suites.
2. Select the row for the test suite whose results you want to compare. The system displays the Test Suite form.
3. Navigate to the Test Suite Results related list.
4. Check the rows for the test suite results you want to compare.
5. From the Actions on selected rows control, click Compare test results. The system displays the Compare test result execution times bar graph.

Compare results for automated test suite runs (aging report)

Compare how many tests passed versus failed across different runs of the same test suite.

Procedure
1. Navigate to Automated Test Framework > Suites.
2. Select the row for the test suite whose results you want to compare. The system displays the Test Suite form.
3. Under Related Links, click Display aging report. The system displays the Test suite aging report.

Administering REST test step configurations

Set request and response payload sizes, filter request and response headers, and create basic auth profiles.

Create a basic auth profile using the Automated Test Framework

Create basic auth profiles to specify basic authentication credentials for Send Request - Inbound test steps.

Before you begin
Role required: web_service_admin

About this task
The user name and password must be valid credentials on the instance where the tests using the profile are run.
Procedure

1. Navigate to Automated Test Framework > Tests.
2. Select a test that uses a Send Request - Inbound step.
3. Select a Send Request - Inbound step.
4. In the Basic authentication field, select the hourglass to look up the available profiles.
5. On the Basic Auth Configurations form, select New.
6. In the Name field, enter a name for the profile.
7. In the Username field, enter a user name.
8. In the Password field, enter a password.
9. Click Submit.

Filter REST request and response headers

You can add a list of REST request and response headers that are not to be saved in step-result records. You can filter headers that might contain authentication credentials or other sensitive information. The phrase "Header redacted for security" is saved instead.

To specify headers to be filtered, create a system property glide.atf.rest.log.header_blacklist with a comma-separated list of header names to be filtered. For information on adding properties, see Add a system property.

Automated Test Framework REST properties

These properties are installed with ATF REST.

ℹ️ Note: To open the System Property [sys_properties] table, enter sys_properties.list in the navigation filter.

<table>
<thead>
<tr>
<th>Property</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>glide.atf.rest.log.header_blacklist</td>
<td>A list of headers whose content is not to be added to the log, or shown on a form. The phrase 'Header redacted for security' is saved instead.</td>
</tr>
<tr>
<td></td>
<td>• Type: String</td>
</tr>
<tr>
<td></td>
<td>• Default value: empty</td>
</tr>
<tr>
<td>Property</td>
<td>Usage</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>glide.atf.rest.request_payload_max_size</td>
<td>The maximum size of the request payload.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Type</strong>: String</td>
</tr>
<tr>
<td></td>
<td>• <strong>Default value</strong>: 100 Kb</td>
</tr>
<tr>
<td></td>
<td>• <strong>Maximum value</strong>: 1024 Kb</td>
</tr>
<tr>
<td></td>
<td>• <strong>Location</strong>: System Property [sys_properties] table</td>
</tr>
<tr>
<td>glide.atf.rest.response_payload_max_size</td>
<td>The maximum size of the response payload.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Type</strong>: String</td>
</tr>
<tr>
<td></td>
<td>• <strong>Default value</strong>: 100 Kb</td>
</tr>
<tr>
<td></td>
<td>• <strong>Maximum value</strong>: 5120 Kb</td>
</tr>
<tr>
<td></td>
<td>• <strong>Maximum value</strong>: System Property [sys_properties] table</td>
</tr>
</tbody>
</table>

**Optimizing automatic test performance**

You can troubleshoot automatic test performance by inspecting system transaction log records and potentially shorten execution time by adjusting how often automatic tests capture screenshots.

**Managing automatic test screenshot settings**

Capturing many screenshots can impair test performance. You can control which types of screenshots the system captures to minimize this effect.

By default, the system captures a screenshot every time it executes a form test step. This information can be useful for understanding test results, but may slow down how fast the system executes the test.

You can change automatic test framework settings so that the system captures all screenshots (as it does by default), no screenshots, or just screenshots for failed test steps.

You can change these settings to affect all tests run on this instance, or to affect just the current client test runner session. To affect all tests run on this instance,
set the automatic test framework property from the automatic test framework properties page. To affect just the current client test runner session, set the screenshot mode from client test runner browser window.

**Set the system property to control when the Automated Test Framework captures screenshots.**

To control how often this instance captures screenshots for form test-steps, set the screenshot capture mode on the automatic test framework properties page.

**Before you begin**
Role required: atf_test_admin or admin

**About this task**
Setting the screenshot mode from the automatic test framework properties page affects any new client test runners started on this instance. This setting does not affect currently-running test-runners.

**Procedure**

1. Navigate to **Automated Test Framework > Administration > Properties**.
2. Set the **Enable or disable screenshot capture** property. From the drop-down menu
   - To capture screenshots for all steps, select **Enable for all steps**.
   - To capture screenshots only for failed steps, select **Enable for all failed steps**.
   - To capture no screenshots, select **Disable for all steps**.
3. Set the **Screenshot timeout** time interval, in seconds. The Client Test Run does not take a screenshot capture if the length of the attempt exceeds this value. Users should review performance settings and browser caches on affected client systems before increasing this value.
4. Click **Save**.

**Set the client test-runner property to control when the Automated Test Framework captures screenshots**

To control how often the current client test runner captures screenshots for form test-steps, set the screenshot capture mode on the client test runner browser window.

**Before you begin**
Role required: atf_test_admin or admin
About this task
Setting the screenshot mode from the client test-runner browser window affects only this client test-runner and persists until this test-runner session is closed. This setting does not affect any other running test-runners or any future test runners.

Procedure
1. From the client test-runner browser window, click the form preferences icon ( ).
2. Click the **Screenshot mode** option.
   • To capture screenshots for all steps, select **Enable for all steps**.
   • To capture screenshots only for failed steps, select **Enable for all failed steps**.
   • To capture no screenshots, select **Disable for all steps**.
3. Click **Save**.

View transaction data for automated test results
To help troubleshoot performance issues with automatic tests, you can inspect related records from the transactions log entry [syslog_transaction] table.

Before you begin
Role required: test_admin or admin

About this task
⚠️ **Note:** The system may not be able to log some transactions with short durations.

Procedure
Navigate to related transaction record in the system transaction log.
• From the **Step results record**, view transactions in the Step transactions related list.
• From the **Test Results record**, view transactions in the Test Transactions related list.

Working with scheduled test suites
You can schedule a test suite to run at a specified date and time.
To schedule a test suite, you need three components:
- a test suite record
- a schedule record specifying when you want the system to run the test suite
- a scheduled suite run record that associates the test suite to run with the schedule for running it

With this model, you can associate a schedule with many different test suites and vice-versa.

**: Note:** You can schedule only test suites, not individual tests. Scheduled tests will only run if there is an open Scheduled Client Test Runner page matching the scheduled suite’s browser conditions. Scheduled tests cannot run on a machine that is locked, powered down, or does not already have the browser open.

The watchlist on the test suite run record also allows you to specify users to receive an email when the system finishes executing the test suite run.

If the test suite contains one or more form steps (steps involving a user interface), you must ensure that a scheduled client test runner is actively running in a browser when the schedule triggers the suite run.

**: Note:** See Browser recommendations and requirements for recommendations and requirements for running the client test runner.

For step-by-step instructions on how to schedule a test suite, see *Schedule an automated test suite*.

**Designate users to receive email when system finishes running a scheduled test suite**

You can designate users to be notified when a scheduled test suite finishes executing.

**Before you begin**

Role required: atf_test_admin

**Procedure**

1. Navigate to the scheduled test suite run record.
2. Add to the record’s **watchlist** all users you want the system to notify when this scheduled test suite run completes.
Automated Test Framework scheduled test suite completed email

When the system completes executing a scheduled test suite, it sends an email to users on the Scheduled Suite Run record watchlist. This email contains information and links to further information about the Scheduled Suite Run and its results.

⚠ Note: Email functionality must be enabled in order to send an email.

Suite Stats

The Suite Stats section of the email reports the number of suites and individual tests broken down by result status: Failed (F), Error (E), Skipped (S), Canceled (C), or Passed (P). For a description of what each status means, see Test suite results record.

Test Suite Results

The Test Suite Results section of the email reports test suite results over time.

⚠ Note: By default, this report includes only suites with failed tests, but you can change this setting with the Email properties field on the Properties page. If set to default, only the failed tests within a test suite are reported.
Note: ATF reports only recent tests results on the email report. All the previous test results still exist if it's within the set retention time limit. See Table cleanup for more information on retention policy for ATF test results.

### Test Results Key

<table>
<thead>
<tr>
<th>Letter</th>
<th>Color</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>red</td>
<td>Test failed</td>
</tr>
<tr>
<td>P</td>
<td>green</td>
<td>Test passed</td>
</tr>
<tr>
<td>S</td>
<td>white</td>
<td>Test skipped</td>
</tr>
<tr>
<td>C</td>
<td>white</td>
<td>Test canceled</td>
</tr>
<tr>
<td>E</td>
<td>orange</td>
<td>Test has error</td>
</tr>
</tbody>
</table>
Each entry acts as a link to the result record for that run. If you point the mouse to any of these entries, the system displays the parent suite number, the parent suite end time, and the test result output.

**Properties affecting email content**
On the Automated Test Framework Properties form, you can set options affecting the format and content of the email.

**Automated Test Framework use case examples**
Use cases can help you construct tests for common scenarios.

**Automated Test Framework use case: test basic form operations**
This use case illustrates testing basic form operations with the Automated Test Framework.

**About this task**

**Steps in test**
**Procedure**

1. Impersonate a user with the permissions needed to perform these steps, in this example ATF.User.

**Example**

**Test step 1 - Impersonate**

2. Open a form, in this example a Catalog Task form.

**Example**

**Test step 2 - Open a new Form**

3. On the open form, set field values, including for any mandatory fields. This example sets field values for **Assigned to**, **Short description**, and **Description**.
### Example

**Test step 3 - Set Field Values**

<table>
<thead>
<tr>
<th>Execution order</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Global</td>
</tr>
</tbody>
</table>

**Table**: Catalog Task [xc_task]

**Field values**:
- **Assigned to**: MFT User
- **Short description**: Deliver PC to lab.
- **Description**: Deliver PC to IT Lab from receiving or move PC from stockroom

4. Submit the open form.

### Example

**Test Step 4 - Submit a Form**

<table>
<thead>
<tr>
<th>Execution order</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Global</td>
</tr>
</tbody>
</table>

**Description**: Submit the current form and confirm form submitted to server

**Assert type**: Form submitted to server
Automated Test Framework use case: reference a value from a previous step

This use case illustrates assigning a form field the value of an output variable from a previous step.

About this task

In this example, the second step references an output value from the first step.

Automated Test Framework: Pass values from one step to another example

Procedure

1. Insert a record into the incident table. This example inserts a record into the Incident table.

Example

Step 1 - Record Insert

2. Open the record just inserted. Specify the record to open by assigning to the Record field, the output variable from Step 1.
3. Validate that fields on the open record have the values you expect.

**Example**

**Step 3 - Field Values Validation**

**Related information**

Pass values from one automated test step to another
Automated Test Framework use case: test a business rule

This use case illustrates testing a business rule with the Automated Test Framework.

About this task

This example tests a business rule that sets the value of Locked out to true when active is set to false.

Automated Test Framework: Business rule example
Procedure

1. Impersonate a user with the necessary permissions. In this example, the step impersonates the admin user.

Example

Step 1 - Impersonate

2. Open a form for the table to which this business rule applies. This example opens a new User form.

Example

Step 2 - Open a New Form

3. Set values on the form that meet the requirements for submitting the form and for triggering the business rule. This examples sets values for the Active, Last name, and First name fields.
Example

**Step 3 - Set Field Values**

<table>
<thead>
<tr>
<th>Execution order</th>
<th>Application</th>
<th>Active</th>
<th>Test</th>
<th>Step config</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Global</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Description**
Set the values on the form as follows:
Active = false
Last name = Testperson
First name = Jane

**Table**
User [sys_user]

**Field values**
- Active
- Last name
- First name

4. Submit the form.

Example

**Step 4 - Submit Form**

<table>
<thead>
<tr>
<th>Execution order</th>
<th>Application</th>
<th>Active</th>
<th>Test</th>
<th>Step config</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Global</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Description**
Submit the current form and confirm form submitted to server

**Assert type**
Form submitted to server
5. Validate that the business rule ran. In this example the business rule tested sets **Locked out** to **true** if **Active** is set to **false**.

**Example**

**Step 5 - Field Values Validation**

Automated Test Framework use case: test a data policy

This use case illustrates testing a data policy with the Automated Test Framework.

**About this task**

This example tests the data policy that sets the field **Assignment Group** to **mandatory** if **impact** is **high**.
Automated Test Framework: Data policy being tested
Automated Test Framework: Data policy example

Procedure

1. Impersonate a user with the necessary permissions. In this example, the step impersonates an admin user.

Example

Step 1 - Impersonate

2. Open a form for the table to which this data policy applies. This example opens a new incident form.
Example

Step 2 - Open a New Form

3. Check that the data policy has not yet been triggered. In this example, the step checks to confirm that **Assignment group** is not mandatory.

Example

Step 3 - Field State Validation
4. If applicable, set the conditions that trigger the data policy. This example sets **Impact** to **High**.

**Example**

**Step 4 - Set Field Values**

- **Execution order**: 1
- **Application**: Global
- **Step config**: Data Policy Test Example
- **Table**: Incident (incident)
- **Field values**:
  - **impact**: 1 - High
  - **~ value~**

Update Delete

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
5. Validate that the data policy is enforced. In this example, the test confirms that the data policy set **Assignment group** to **High** after the previous step set **Impact** to **High**.

Example

**Step 5 - Field State Validation**

Automated Test Framework use case: test a script include

This use case illustrates testing a script include with the Automated Test Framework.

To test a script include with the Automated Test Framework, create a test that performs these steps:

- a step that causes the system to execute the script include. Examples:
  - Run Server Side Script test step that calls the script include.
  - Form action step script – such as open a form, submit or form, or set a field value – that invokes the script include.

- A step that validates that the script include took the expected actions. The specific test step for this validation depends on what the script include is designed to do. Examples:
- Field values validation, if the script sets a field value
- Field state validation, if the script changes a field state
- Record Query, if the script generates a new record

This example shows a test with one test step: Run Server Side Script. The script associated with this test step calls a script include that returns the value of its argument plus three. If the value returned from the script include is 8, the script include has worked as intended and the test passes.

Automated Test to test a script include
Automated test step for testing script include

```
62 //
63 // assertEqual: A function used to compare that assertion.value == assertion.value;
64 // In case of failure it throws an error and logs that the assertion by
65 // name has failed
66 //
67 // Example:
68 //
69 // var testAssertion = {};
70 // name: "my test assertion",
71 // shouldbe: "expected value"
72 // value: "actual value"
73 //
74 // assertEqual(testAssertion); // throws error, logs message to test step output
75 //
76 // function testoutput(step, stepResult, assertEqual)
77 // var input = assertEqual(s);
78 // if (input === true)
79 // return true;
80 // else
81 // return false;
82 //)
83 //)
84 //)
85 //)
86 //)
87 //)
88 //)
89 //)
```

Script include to test with Automated Test Framework

```
1 - function addThree() {
  2   return x + 3;
  3 }
```

Automated Test Framework use case: test a Service Catalog request

This use case illustrates testing a service catalog request with the Automated Test Framework.
About this task
With the Replay Request Item test step, you can test the service catalog ordering process once a request exists and has a record in the request item table. In the Rome release, you cannot create an automated test for the process by which the user creates a new request.

Automated Test Framework: Service catalog example
Procedure

1. Replay an existing service catalog request item. This test step inserts a new record in the [sc_request] table for the catalog request item RITM0010001.

Example

Service Catalog test step 1 details: Replay Request Item

This insertion triggers the Service Catalog Request workflow, which checks the price of the item, determines that it exceeds $1000.00, and therefore generates approval records for users belonging to the Catalog Request Approvals group. In this example, only one user–Eric Schroeder–belongs to this group.

2. Impersonate Eric Schroeder, the user who needs to approve this Service Catalog Request.
3. Verify that the system created an approval record for Eric Schroeder and this request. Note that for the Approval for field, you assign the output value from Step 1: Replay Request item > Request.

**Example**

**Step 3 details: Record Query for Approval record**

4. Set the state of this approval record to Approved.
The Service Catalog Request workflow sees that all required approval records have the state of **Approved** and transitions to the Approval Action which marks the request record \[\text{sc\_request}\] as **Approved**.

**Step 4: Triggered workflow marks request record as approved**

When the record in \[\text{sc\_request}\] changes to the Approved state, an associated business rule generates request items \[\text{sc\_request\_item}\] for each item in the request. In this example, the request contains only one item, so the business rule inserts one record into the \[\text{sc\_request\_item}\] table. This insertion triggers the Service Catalog Item Request workflow.
The first activity in the Service Catalog Item Request workflow generates an approval record for the head of the department in which the requesting user works. In this example, the department head is Natasha Ingram.

5. The workflow does not continue until the department head approves it, so the next test step impersonates Natasha Ingram.
Example

Step 5 - Impersonate User

6. Obtain the sys_id for the new approval record with the Record Query step. Note that Record Query creates an output variable with the sys_id of the first record returned from the query.

Example

Step 6 - Record Query test step

7. Set the approval record to Approved.
Example

**Step 7 - Approval User test step**

Note how Step 7 refers to the **First record** output variable from Step 6 to specify which record to approve. When the record is approved, the workflow transitions to the next Approval - User activity, which generates an approval record for the CIO. In this example, the CIO is Bow Ruggeri.

**Step 7 details - Service Catalog Item workflow**

8. Impersonate Bow Ruggeri.
9. Obtain the sys_id for the approval record for Bow Ruggeri.
10. Set the approval record to **Approved**.
When the record is approved, the workflow transitions to the Approval Action activity which sets the record for this item in the [sc_request_item] table to **Approved**. The workflow transitions to the Catalog Task activity labelled *Asset Mgmt. Fulfills Order*. This Catalog Task activity generates a new record in the [sc_task] table that instructs a user in the Fulfillment group to order the item.

**Step 10 - Service Catalog Item workflow**

11. Impersonate a user in the Fulfillment group, in this example ATF.User.

**Example**

**Step 11 - Impersonate User test step**

12. Obtain the sys_id for the new catalog task with the Record Query step. Note that Record Query creates an output variable with the sys_id of the first record returned from the query.
Example

Step 12 - Record Query test step

13. Mark the [sc_task] record as Closed Complete.

Example

Step 13 - Record Update test step

Note how Step 13 uses the First record output variable from Step 12 to specify which record to mark as Closed Complete.

When the record is marked Closed Complete, the workflow exits the Catalog Task activity along the In Stock exit path.
Step 13 - Service Catalog Item Request workflow

The workflow transitions to the Notification activity, then to the Catalog Task activity labelled Deploy Item to User. The Deploy Item to User Catalog Task activity inserts a new record into the [sc_task] table that instructs a user in the Deployment group to deliver the item.

14. Obtain the sys_id for the new catalog task with the Record Query step. Note that Record Query returns an output variable with the sys_id of the first record returned from the query.

Example

Step 14 - Record Query test step
15. Mark the [sc_task] record as *Closed Complete*. Note how Step 15 uses the *First record* output variable from Step 14 to specify which record to mark as *Closed Complete*.

**Example**

**Step 15 - Record Update test step**

When the record is marked *Closed Complete*, the workflow exits the Catalog Task activity, logs a message, and exits.

**Step 15 - Service Catalog Item Request workflow**

16. Verify that the request item in [sc_request_item] has the state *Closed Complete*. 
17. Verify that the request \[sc_request\] has the state Closed Complete.

**Example**

**Step 16 - Record Validation test step**
Automated Test Framework use case: Retrieve an incident using REST-Inbound

The Get Newly Created Resource via REST API Test test is provided with the Automated Test Framework, and uses the REST - Inbound and assert steps.

Before you begin
Review the REST API and the Send REST Request- Inbound- REST API Explorer configuration step information before creating this test.

Role required: atf_ws_designer

About this task
This test creates an incident record, uses a REST- Inbound step to retrieve the record, and then uses assert steps to determine whether the request was successful.

You can use the Send REST Request- Inbound REST API Explorer step to build and test the request, or you can manually create the request using the Send REST Request- Inbound step. In either case, you must specify the basic authentication information on the Send REST Request- Inbound step form.

When creating your test, start with your REST API and determine what behavior you want to validate. You can then determine what test data to create. You can use other test step configuration categories to create, update, or delete records, and then use a Send REST Inbound step to retrieve, update, or delete the test data. Conversely, you can use the Send REST Inbound step to create records, and then use other test step configuration categories to validate that the records were created correctly.

This task steps you through creating the Get Newly Created Resource via REST API Test test.

Procedure

1. Create a test.

   a. Navigate to Automated Test Framework > Tests, click New. The Test new record form is shown.

   b. Enter a test name and a description, and click Submit.

   c. Click the test created in the previous step. The Test form is shown.

2. Create test data.

   a. Click the Add Test Step button. The Add Test Step form is shown.
b. Click a test configuration category, and then click a test configuration.
   To replicate the **Get Newly Created Resource via REST API Test** test, click **Server > Record Insert**, and then **Next**.
   The test configuration form you selected is shown.

c. Fill in the information needed for the test configuration you selected.
   To replicate the **Get Newly Created Resource via REST API Test** test, on the **Record Insert** form, specify the incident table, and add a value for the Short description field, and click **Submit**.

3. Create the REST request.

   a. Click the **Add Test Step** button.
      The **Add Test Step** form is shown.

   b. Click a test configuration category, and then click a test configuration.
      To replicate the **Get Newly Created Resource via REST API Test** test, click **REST > Send REST Request - Inbound REST API Explorer**, and then **Next**.
      The **REST API Explorer** is shown.

   c. Fill in the information needed for the REST request, and click **Send**.
      To replicate the **Get Newly Created Resource via REST API Test** test, fill in the fields.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace</td>
<td>now</td>
</tr>
<tr>
<td>API Name</td>
<td>Table API</td>
</tr>
<tr>
<td>API Version</td>
<td>latest</td>
</tr>
<tr>
<td>tableName</td>
<td>incident</td>
</tr>
</tbody>
</table>

When you do not specify the **Basic authentication**, the REST API Explorer uses your credentials.

The **Create Automated Test Step** is shown after you click **Send**.

d. When ready, click **Create Automated Test Step**.
   The Send REST Request - Inbound test step is created.

e. Click the Send REST Request - Inbound step, and in the **Basic authentication** field, specify a basic auth configuration.
If no configurations are available, you can create a basic auth configuration by clicking New on the Basic Auth Configurations form.

f. To replicate the Get Newly Created Resource via REST API Test test, in the Path field, click the contextual search button, and then click Record Insert > Record.

g. Click Update.

4. Create assert steps to verify the REST response.

   a. Click the Add Test Step button.
      The Add Test Step form is shown.

   b. Click a test configuration category, and then click a test configuration.
      To replicate the Get Newly Created Resource via REST API Test test, click REST > Assert Status Code, and then Next.
      The Assert Status Code form is shown.

   c. Enter the information needed for the form, and click Submit.
      To replicate the Get Newly Created Resource via REST API Test test, in the Operation field, select is and in the Status Code field, enter 200.
      To replicate the Get Newly Created Resource via REST API Test test, repeat this step to create Assert Response JSON Payload is Valid, and Assert JSON Response Payload Element test steps.

Quick start tests

Copy and customize quick start tests provided by the Now Platform® to validate that your instance works after you make any configuration changes. For example, if you apply an upgrade or develop an application.

The tests can only produce a pass result when you run them with the default demo data that’s provided with the application or feature plugin.

⚠️ Note: If your QST fails, it can be that the test doesn’t match your instance customizations. Use the Notes field for each test step to update the test to pass with your data. If you have customized the business process associated with a test, revise the test to match your customizations.

To apply a quick start test to your instance-specific data, copy the quick start test and add your custom data.

⚠️ Note: You can copy either individual quick start test or the entire quick start tests suite.
Activation

Each application or feature has its own plugin activation requirements for enabling quick start tests. See Available quick start tests by application or feature for activation information.

Managing copies

When you copy a test, Automated Test Framework populates the Copied from field with the name of the copied test. When an upgrade changes a quick start test Automated Test Framework notifies test designers about the change in a notification on the test form. Test designers can revert the copied test to the upgraded version with the Revert Copy to Base System UI action.

Tip: If a QST is updated, the previously copied versions of the QST won’t have the recent changes. Review the old copies using Copies to Review for all the tests that have an associated warning message.

Available quick start tests by application or feature

Validate that your instance still works after you make any configuration change such as apply an upgrade or develop an application. Copy and customize the ServiceNow-provided quick start tests to pass when using your instance-specific data.

DANGER: By default, the system property that is used to run automated tests is disabled to prevent you from accidentally running these tests on a production system. To avoid data corruption or an outage, run tests only on development, test, and other non-production instances. See Enable or disable executing Automated Test Framework tests.

Agile Development 2.0

Agile Development 2.0 quick start tests require activating the Agile Development 2.0 plugin (com.snc.sdlc.agile.2.0) and the Agile Development 2.0 - ATF Tests plugin (com.snc.sdlc.agile.2.0.atf).

Agile Development 2.0 quick start test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verify that global rank is populated when a story is created</td>
<td>Verify the global rank of a story after creation.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>
### Agile Development 2.0 quick start test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verify that closing a sprint with active stories is prevented</td>
<td>Verify that a sprint with active stories cannot be closed.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Verify that sprints cannot overlap in the same group</td>
<td>Verify that sprints in the same group do not overlap.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Verify that sprint points are updated</td>
<td>Verify that changes to stories produce accurate sprint point totals.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Verify that only one sprint in a group can have the current state</td>
<td>Verify sprint statuses.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Verify sprint end date is after the sprint start date</td>
<td>Verify sprint start and end dates.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Verify that any update on story rolls up to Epic</td>
<td>Verify that adding, estimating, removing, deleting, updating, or cancelling a story updates the epic-level roll-ups correctly.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Verify changes to the scope of a current sprint do not alter the value of the Total Committed Points</td>
<td>Verify that the value of Total Committed Points does not change with change in the scope of a sprint after its state is changed to Current.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Verify active flag is set false when Agile Story state is changed to Completed/Cancelled</td>
<td>Verify that active flag of an Agile story is set to the following:</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td>• False, if the state is changed to Completed or Cancelled</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• True, for all other states</td>
<td></td>
</tr>
</tbody>
</table>
### Agile Development 2.0 quick start test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verify that updating the team/group capacity overrides the capacity on all the future sprints</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
</table>
| Verify that any update to the **Group capacity** field of the assignment group results in the following changes to the **Group capacity** field of the various sprints associated with this assignment group:  
  - For the sprints that are in the Draft and Planning state:  
    - The group capacity is updated to the new value.  
    - The **Group capacity** field is editable.  
  - For the sprints in the Current, Complete, or Cancelled state:  
    - The group capacity remains the old value.  
    - The **Group capacity** field is read-only. |

<table>
<thead>
<tr>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paris</td>
</tr>
</tbody>
</table>

For the sprints in the Draft or Planning state, you can individually edit the group capacity of the sprint anytime later. This would not change the group capacity of the assignment group associated with this sprint.
## Agile Development 2.0 quick start test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
</table>
| Verify create and edit functionality of an epic backlog on the scrum program board | • Verify that you can create an epic backlog for scrum programs from the Backlog tab of Agile Board.  
• Verify that you can update an existing epic backlog for scrum programs from the Backlog tab of Agile Board.  
• The epics listed in the backlog must belong to the selected epic backlog. | Quebec          |
| Verify a Demand is converted to a scrum story                      | If the PPM Standard plugin (com.snc.financial_planning_pmo) is active, verify that a Demand can be converted to an Agile 2.0 story using the **Create Story** related link on the Demand form.  
For more information on how to create an Agile 2.0 story from a demand, see **Create an artifact from a demand**. | Quebec          |
| Verify a Demand is converted to a scrum epic                       | If the PPM Standard plugin (com.snc.financial_planning_pmo) is active, verify that a Demand can be converted to an Agile 2.0 epic using the **Create** | Quebec          |
### Agile Development 2.0 quick start test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Epic related link on the Demand form. For more information on how to create an Agile 2.0 epic from a demand, see Create an artifact from a demand.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

To learn more about Agile Development 2.0, see Agile Development 2.0.

### Application Portfolio Management

Application Portfolio Management quick start tests require enabling the Application Portfolio Management – ATF Tests plugin (com.snc.apm.atf).

#### APM: Create Business application and capability test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>APM: Create Business Application</td>
<td>Verify the creation of an application category and then the creation of a business application with users having apm_user role.</td>
<td>Madrid</td>
</tr>
<tr>
<td>APM: Create Business Capability</td>
<td>Verify the creation of a parent and child business capability and verify its field values.</td>
<td>Madrid</td>
</tr>
<tr>
<td>APM: Test relating Business Service, Business Application, and Software Models</td>
<td>Verify the creation of a business application, business service, using the existing software model, and a relationship between them.</td>
<td>Orlando</td>
</tr>
<tr>
<td>APM: Test for Indicator Score and Application Score generation</td>
<td>Verify the creation of indicator, scoring profile, and generation</td>
<td>Paris</td>
</tr>
</tbody>
</table>
### APM: Create Business application and capability test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>of indicator scores and application scores.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>APM: Business Application with Information Object and Data Domain</td>
<td>Verify the creation of business application, information object, and addition of the CRUD operations in relation attributes.</td>
<td>Quebec</td>
</tr>
</tbody>
</table>

To learn more about Application Portfolio Management, see [Application Portfolio Management](#).

### Assessments and Surveys

Assessments and Surveys quick start tests require activating the Automated Test Framework for Survey plugin (com.glide.automated_testing_impl.Survey).

<table>
<thead>
<tr>
<th>Test Suite for Survey</th>
<th>Name</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Survey: Basic Platform Based Test</td>
<td>Create a survey using Platform UI actions.</td>
<td>Madrid</td>
</tr>
<tr>
<td></td>
<td>Survey: Platform test for Dynamic Validation</td>
<td>Validate a survey dynamically.</td>
<td>Madrid</td>
</tr>
<tr>
<td></td>
<td>Survey: Clone Action</td>
<td>Clone a survey and validate the records of the original survey and the cloned survey.</td>
<td>New York</td>
</tr>
<tr>
<td></td>
<td>Survey: Question Bank Flow</td>
<td>Verify the addition of a question bank to a survey.</td>
<td>New York</td>
</tr>
<tr>
<td></td>
<td>Survey: Survey Creator Work Flow</td>
<td>Survey creator can create a survey and assign to the user who can take the survey and submit it.</td>
<td>New York</td>
</tr>
</tbody>
</table>
### Test Suite for Survey (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assessment: Assign assessment to assessor</td>
<td>Assign an assessment to an assessor and verify that the instance is created for the assessor.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Assessment: Create assessment as survey creator</td>
<td>Create an assignment as a survey creator.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Assessment: Basic test flow</td>
<td>Verify the basic flow of an assessment.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

To learn more about Assessments and Surveys, see Service administration.

### Change Management

Change Management quick start tests require activating the Change Management - ATF Tests plugin (com.snc.change_management.atf).

**CHG: Emergency Type Change Request test suite**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emergency Type Change Request workflow</td>
<td>Process an emergency change request from new to closed.</td>
<td>Madrid</td>
</tr>
<tr>
<td>On Hold for Emergency type Change Request</td>
<td>Validate the approval state of an on-hold emergency change request.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Copy Change For Emergency type Change Request</td>
<td>Validate the state of a copied emergency change request.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Reject By Approver for Emergency type</td>
<td>Validate the state of a rejected emergency change request.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Revert to new for emergency type</td>
<td>Validate the state of an emergency change</td>
<td>Madrid</td>
</tr>
</tbody>
</table>
### CHG: Emergency Type Change Request test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convert Emergency to Normal type</td>
<td>Validate the conversion of an emergency change request to a normal change request.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Cancel Change Request For Emergency Type</td>
<td>Validate the state of a canceled emergency change request.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>

### CHG: Normal Type Change Request test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal Type Change Request Workflow</td>
<td>Process a normal change request from new to closed.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Convert Normal to Emergency type</td>
<td>Validate the conversion of a normal change request to an emergency change request.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Copy change on Normal Change Request</td>
<td>Validate the state of a copied normal change request.</td>
<td>Madrid</td>
</tr>
<tr>
<td>On hold for Normal type Change Request</td>
<td>Validate the approval state of an on-hold normal change request.</td>
<td>Madrid</td>
</tr>
<tr>
<td>State validation when Reject Normal type Change request by Approver.</td>
<td>Validate the state of a rejected normal change request.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Revert to New Functionality for Normal Type Change Request</td>
<td>Validate the state of a normal change request after using the Revert to new UI action.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Cancel Change Request For Normal type</td>
<td>Validate the state of a canceled normal change request.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>
### CHG: Standard Change Proposal test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard Change Proposal</td>
<td>Determine whether a user can successfully perform standard change proposal creation, approval, and template publishing processes.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>

### CHG: Standard Type Change Request test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard Change Request Workflow</td>
<td>Process a standard change request from new to closed.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Convert Standard to Emergency Change Request</td>
<td>Validate the conversion of a standard change request to an emergency change request.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Convert Standard to Normal Change Request</td>
<td>Validate the conversion of a standard change request to a normal change request.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>

### CHG: Unauthorized Change Request and Outage test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unauthorized change request Workflow</td>
<td>Process a unauthorized change request from new to closed.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Create Outage of type planned outage from change request</td>
<td>Validate the creation of an outage of type planned outage from a change request.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Create Outage of type outage from change request</td>
<td>Validate the creation of an outage of type outage from a change request.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
**CHG: Risk Conditions with Best practice plugin test suite**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Moderate Risk with UI Action Property</td>
<td>Process a moderate risk with UI action property.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Low Risk with UI Action Property</td>
<td>Process a low risk with UI action property.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Leave Alone Risk with UI Action Property</td>
<td>Process a leave alone risk with UI action property.</td>
<td>Orlando</td>
</tr>
<tr>
<td>High Risk with UI Action Property</td>
<td>Process a high risk with UI action property.</td>
<td>Orlando</td>
</tr>
<tr>
<td>High Risk with Business Rule Property</td>
<td>Process a high risk with Business rule property.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Moderate Risk with Business Rule property</td>
<td>Process a moderate risk with Business rule property.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

**CHG: Change Request against Conflict Sources test suite**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change against Blackout window</td>
<td>Process a change request against a blackout window.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Check conflicts for CI Already Scheduled</td>
<td>Validate the conflicts for CI already scheduled.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Change against Conflict Sources</td>
<td>Validate the change request against conflict sources.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

**CHG: Change Schedule Definition and Sharing test suite**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change Schedules Definitions on New</td>
<td>Process the creation of change schedules definitions from New button on Change Schedules landing page.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
CHG: Change Schedule Definition and Sharing test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Share Panel On Change Schedules Definition</td>
<td>Validate the share panel on change schedules definition.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Share Change Schedule Definition</td>
<td>Validate the sharing of change schedules definitions.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Create Standard Change via Service Portal</td>
<td>Create Standard Change from Service Portal</td>
<td>Rome</td>
</tr>
</tbody>
</table>

To learn more about Change Management, see Change Management.

Cloud Provisioning and Governance
Cloud Provisioning and Governance quick start tests require activating the following Cloud Provisioning and Governance plugins:

- Cloud Provisioning and Governance (com.snc.cloud.mgmt)
- Domain Support- Domain Extensions Installer plugin (com.glide.domain.msp_extensions.installer)
- Service Catalog- Domain Separation plugin (glideapp.servicecatalog.domain_separation)

Cloud Provisioning and Governance: Azure test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMP: Add Azure Credentials</td>
<td>Add Azure credentials by inserting the fields into the Credentials table.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Azure Service Account</td>
<td>Add Azure service account by inserting the fields into the service account table.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Update datacenter type</td>
<td>Update the Cloud Service account with the datacenter type.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>CMP: Creating Resource Group</td>
<td>Creating Resource Group to provision Azure stack via ARM Template.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Create Azure Logical datacenter.</td>
<td>Create Azure Logical datacenter for provisioning resources.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Create 'Hosted on' relationship.</td>
<td>Create hosted on relationship between logical datacenter and Cloud Service account.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Create Cloud Account</td>
<td>Create cloud account by inserting the fields into the cloud account table.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Cloud account and Logical datacenter association</td>
<td>Create an association between CMP cloud account and Logical datacenter.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Create Catalog Item</td>
<td>Create Cloud Catalog Item with ARM Template.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Create template version.</td>
<td>Create Cloud Template Version for Cloud Catalog Item.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: ARM template body</td>
<td>Update Cloud Template Version with ARM template body.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Open the 'ServiceNow Cloud Template Versions'</td>
<td>Open the 'ServiceNow Cloud Template Versions' GUI page.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
Cloud Provisioning and Governance: Azure test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMP: Activate Cloud Template Version</td>
<td>Activate Cloud Template Version to create resource block, blueprint, and catalog.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Verify the Cloud Template Version.</td>
<td>Validate whether status is Success and state is active in the Cloud Template Version after Activation operation.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Activate Cloud Catalog Item</td>
<td>Activate Cloud Catalog Item to display catalog order form to the cloud user.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Add stack and resource group name in Catalog item</td>
<td>Set the default value for stack name and resource group in the catalog item.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Verify whether stack is present</td>
<td>Validate whether stack is available with name &quot;ATFARMStack&quot;.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Open Cloud Catalog Item page</td>
<td>Open Cloud Catalog Item page in the Service Portal.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Submit Catalog form</td>
<td>Submit the Catalog Form to start provisioning ARM stack.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Verify whether stack status is active</td>
<td>Validate the status of a stack after provisioning the stack.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Verify stack status after Day2 - Stop operation</td>
<td>Validate the status of the stack status changed from 'On' to 'Off' after the Stop operation.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
### Cloud Provisioning and Governance: Azure test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMP: Verify stack status after Day2 - Start operation</td>
<td>Validate the status of the stack status changed from 'Off' to 'On' after the Start operation.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Verify stack status after Day2 - Deprovision operation</td>
<td>Validate the status of the stack status changed from 'On' to 'Terminate' after the de-provisioning operation.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

### Cloud Provisioning and Governance: AWS test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMP: Add AWS Credentials</td>
<td>Add AWS credentials by inserting the fields into the credential table.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: AWS Service Account</td>
<td>Add AWS service account by inserting the fields into the service account table.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Update datacenter type</td>
<td>Update the cloud service account with the datacenter type.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Create logical datacenter</td>
<td>Create Logical datacenter for provisioning the resources.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Create hosted on relationship</td>
<td>Create hosted on relationship between logical datacenter and Cloud Service account.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Create Cloud Account</td>
<td>Create cloud account by inserting the fields into the cloud account table.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
### Cloud Provisioning and Governance: AWS test suite (continued)

<table>
<thead>
<tr>
<th>Test Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create an association between CMP cloud account and logical datacenter.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Create Cloud Catalog Item with the CFT Template.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Created Cloud Template Version for a Cloud Catalog Item.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Update Cloud Template Version with the CFT template body.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Open the 'ServiceNow Cloud Template Versions' GUI page.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Activate Cloud Template Version to create resource block, blueprint, and catalog.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Validate the Cloud Template Version whether status is Success and state is active after Activation operation.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Activate Cloud Catalog Item to display the catalog order form to the cloud user.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Set the default value for stack name in the catalog item.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Validate whether stack is available with name &quot;ATFCFTStack&quot;.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
Cloud Provisioning and Governance: AWS test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMP: Open Cloud Catalog Item page</td>
<td>Open the Cloud Catalog Item page in the Service Portal.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Submit Catalog form</td>
<td>Submit the Catalog Form to start provisioning CFT Stack.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Verify whether stack status is active</td>
<td>Validate the status of the stack after provisioning the stack.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Verify stack status after Day2 - Stop operation</td>
<td>Validate the status of the stack status changed from 'On' to 'Off' after the Stop operation.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Verify stack status after Day2 - Start operation</td>
<td>Validate the status of the stack status changed from 'Off' to 'On' after the Start operation.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CMP: Verify stack status after Day2 - De-provision operation</td>
<td>Validate the status of the stack status changed from 'On' to 'terminate' after the de-provisioning operation.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

To learn more about Cloud Provisioning and Governance, see Cloud Provisioning and Governance.

Coaching

Coaching quick start tests requires activation of the Coaching plugin (com.sn_coaching).

Coaching test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coaching: Create an assessment manually</td>
<td>As a coach, verify that you can create an</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
## Coaching test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>when logged in as a coach.</td>
<td>assessment and assign it to trainees.</td>
<td></td>
</tr>
<tr>
<td>Coaching: Add skills to an opportunity and verify those skills awarded to trainee.</td>
<td>Add skills to a coaching opportunity and verify that those skills have been awarded to trainees after they complete an assessment.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Coaching: Complete an assessment as a virtual coach.</td>
<td>Verify that the virtual coach completes an assessment and provides feedback to the trainee when a virtual coach is attached to an opportunity.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Coaching: Add skills to an assessment and verify those skills awarded to trainee.</td>
<td>Add skills to a coaching assessment and verify that those skills have been awarded to trainees after they complete an assessment.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Coaching: Add skills to a recommendation and verify those skills awarded to trainee.</td>
<td>Add skills to a coaching recommendation and verify that those skills have been awarded to trainees after they complete an assessment.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Coaching: Move assessments from one state to another when logged in as a coach.</td>
<td>As a coach, verify that you can move an assessment from one state to another.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Coaching: Attach a recommendation</td>
<td>Verify that a recommendation learning on an</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.  
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Coaching test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>learning to an assessment.</td>
<td>opportunity gets attached to an assessment when an assessment is generated.</td>
<td></td>
</tr>
<tr>
<td>Coaching: When a coaching opportunity is inactive, assessments are not generated.</td>
<td>Verify that when a coaching opportunity is in inactive state, it does not generate assessments.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Coaching: Submit Coaching survey as a Coach user.</td>
<td>Verify that coach can submit survey for a trainee.</td>
<td>Quebec</td>
</tr>
<tr>
<td>Coaching: Submit Coaching survey as a Trainee user.</td>
<td>Verify that trainee can submit survey for a coach.</td>
<td>Quebec</td>
</tr>
</tbody>
</table>

To learn more about coaching, see Coaching.

Communities

Communities quick start tests require activating the Customer Communities plugin (com.sn_customer_communities) and the Communities Demo Data plugin (com.sn_communities_demo).

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Post question and validate feed</td>
<td>Verify that a question is posted in the community and validate whether it appears in the content feed.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Post video</td>
<td>Verify that a video is posted in the community.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Forum membership approval</td>
<td>Verify that a membership request to a forum is approved.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Forum membership reject</td>
<td>Verify that a membership request to a forum is rejected.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Approve a moderation task</td>
<td>Verify that a moderation task is approved.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Reject a moderation task</td>
<td>Verify that a moderation task is rejected.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Topic subscription and activity feed validation</td>
<td>Verify that a topic is subscribed to and the topic activities appear in the activity feed.</td>
<td>New York</td>
</tr>
<tr>
<td>Approve content approval workflow task on question</td>
<td>Verify the content approval workflow of a question.</td>
<td>New York</td>
</tr>
<tr>
<td>Question auto-subscription and activity feed validation</td>
<td>Verify that the author is automatically subscribed to the question and the question activities appear in the activity feed.</td>
<td>New York</td>
</tr>
<tr>
<td>Follow a user</td>
<td>Verify that a community user is able to follow another community user.</td>
<td>New York</td>
</tr>
<tr>
<td>Reject content approval workflow task on question</td>
<td>Verify the content approval workflow of a question when content is rejected.</td>
<td>New York</td>
</tr>
<tr>
<td>Gamification on video posting</td>
<td>Verify the gamification points gained when posting a video.</td>
<td>New York</td>
</tr>
</tbody>
</table>
### Communities: Community test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forum subscription and activity feed validation</td>
<td>Verify that a forum is subscribed to and the forum activities appear in the activity feed.</td>
<td>New York</td>
</tr>
<tr>
<td>Question subscription and activity feed validation</td>
<td>Verify that a question is subscribed to and the question activities appear in the activity feed.</td>
<td>New York</td>
</tr>
<tr>
<td>Post a video with 'Disable comment' option</td>
<td>Verify a video is posted with Disable comments option as selected.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Pin a video as Featured</td>
<td>Verify that a video is marked as featured by community administrator in the forum and community home page.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

To learn more about Communities, see [Communities](#).

### Configuration Compliance

Configuration Compliance quick start tests require activating the Configuration Compliance application (sn_vulc) and loading the demo data.

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Compliance Assignment Rules</td>
<td>Create an assignment rule.</td>
<td>Rome</td>
</tr>
<tr>
<td>Configuration Compliance - Reapply Group Rule</td>
<td>Reapply a test result group rule.</td>
<td>Rome</td>
</tr>
<tr>
<td>Configuration Compliance - Delete Group Rule</td>
<td>Delete a test result group rule.</td>
<td>Rome</td>
</tr>
</tbody>
</table>

To learn more about Configuration Compliance, see [Configuration Compliance](#).
Configuration Management Database (CMDB)

Configuration Management Database (CMDB) quick start tests require activating the Configuration Management (CMDB) plugin (com.snc.cmdb) and the CMDB - ATF Tests plugin (com.snc.cmdb.atf).

CMDB BSM: Dependency Views test suite
Test suite to check functionality of Dependency Views APIs.

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMDB BSM: Dependency Views</td>
<td>Test functionality of Dependency Views APIs. These APIs retrieve Dependency Views map and associated map items such as context menu items, for a given CI sys_id and using itil user role.</td>
<td>New York</td>
</tr>
</tbody>
</table>

CMDB HEALTH: CI Health Dashboard test suite
Test suite to check whether CMDB CI Health Dashboard is functional at a basic level.

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMDB HEALTH: Health Job Status</td>
<td>Checks whether any CMDB Health dashboard jobs, which were started 30 or more days ago, are still in progress.</td>
<td>New York</td>
</tr>
<tr>
<td>CMDB HEALTH: CMDB Health Completeness/Recommended</td>
<td>Checks whether the CI dashboard is functional for the recommended metric (included in the CMDB Health completeness KPI). This test validates:</td>
<td>New York</td>
</tr>
</tbody>
</table>
**CMDB HEALTH: CI Health Dashboard test suite**

Test suite to check whether CMDB CI Health Dashboard is functional at a basic level.

(continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Creation of a health inclusion rule for the recommend metric.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Creation of a recommended field that satisfies the health inclusion rule.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Validate that the health inclusion rule is correctly applied to a test record with missing data in the recommended field.</td>
<td></td>
</tr>
</tbody>
</table>

**CMDB IRE: Identification Reconciliation Engine test suite**

Test suite to check Identification and Reconciliation Engine (IRE) functionality.

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMDB IRE: IRE Validation</td>
<td>Validate CI identifiers and reconciliation definitions and check indexes for CI identifiers.</td>
<td>Madrid</td>
</tr>
<tr>
<td>CMDB IRE: Reconciliation Rule</td>
<td>Check operations on a reconciliation rule, in CI Class Manager, using itil and itil_admin roles. Operations include create, edit, and delete a reconciliation rule. Also, check for active and not active setting, and derived rules.</td>
<td>Paris</td>
</tr>
</tbody>
</table>
**CMDB IRE: Identification Reconciliation Engine test suite**

Test suite to check Identification and Reconciliation Engine (IRE) functionality.

(continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMDB IRE: Identification Rule</td>
<td>Check operations on an identification rule, in CI Class Manager, using itil and itil_admin roles. Operations include create, edit, and delete an identification rule. Also, check for active and not active setting, and derived rules.</td>
<td>Paris</td>
</tr>
</tbody>
</table>

**CMDB QB: Query Builder test suite**

Test suite to verify CMDB Query Builder functions such as create query, read query, and execute query using two related user roles - cmdb_query_builder and cmdb_query_builder_read.

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMDB QB: Query Builder - cmdb_query_builder Role</td>
<td>Verify that cmdb_query_builder user role can save queries, and access and run all saved queries, in CMDB Query Builder.</td>
<td>New York</td>
</tr>
<tr>
<td>CMDB QB: Query Builder - cmdb_query_builder_read Role</td>
<td>Verify that cmdb_query_builder_read user role can access and run all saved queries, and cannot save any query, in CMDB Query Builder.</td>
<td>New York</td>
</tr>
</tbody>
</table>
### CMDB REL: Relationship Editor and Formatter test suite

Test suite to verify functionality of Relationship Editor and Relationship Formatter.

<table>
<thead>
<tr>
<th>Test Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMDB REL EDITOR:Relationship Editor</td>
<td>Check addition of relations to a CI and deletion of relations from a CI using itil user role.</td>
</tr>
<tr>
<td>CMDB REL FORMATTER:Relationship Formatter</td>
<td>Check accuracy of CI information, relationship types, relationships, associated records such as change tickets, and settings such as CMDB views (relationship filters), displayed for a specific CI in relationship formatter using itil user role.</td>
</tr>
</tbody>
</table>

### CMDB Workspace: Agent Workspace test suite

Test suite to verify functionality of CMDB Agent Workspace.

<table>
<thead>
<tr>
<th>Test Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMDB Workspace: Workspace Validation</td>
<td>Test adding, editing, and deleting of a list item, and editing of a filtered list, using itil and admin roles.</td>
</tr>
</tbody>
</table>

### CMDB SDK: SDK REST API test suite

Test suite to verify functionality of CMDB SDK Rest APIs.

<table>
<thead>
<tr>
<th>Test Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMDB SDK: Query CMDB Metadata</td>
<td>Test querying CMDB metadata.</td>
</tr>
</tbody>
</table>
CMDB SDK: SDK REST API test suite

Test suite to verify functionality of CMDB SDK Rest APIs.

(continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMDB SDK: Create a relation for a CI using REST APIs</td>
<td>Test creation of a relationship for a CI in the CMDB using the CMDB REST APIs.</td>
<td>New York</td>
</tr>
<tr>
<td>CMDB SDK: Delete a relation for a CI using REST APIs</td>
<td>Test deletion of a relationship for a CI using CMDB REST APIs.</td>
<td>New York</td>
</tr>
<tr>
<td>CMDB SDK: Create a CI using REST API</td>
<td>Test creation of a CI using CMDB REST APIs.</td>
<td>New York</td>
</tr>
<tr>
<td>CMDB SDK: Query CMDB using REST APIs</td>
<td>Test querying the CMDB using CMDB REST APIs.</td>
<td>New York</td>
</tr>
<tr>
<td>CMDB SDK: Update a CI using REST APIs</td>
<td>Test updating of a CI using CMDB REST APIs.</td>
<td>New York</td>
</tr>
<tr>
<td>CMDB SDK: Query for a CI using REST APIs</td>
<td>Test querying a CI using CMDB REST APIs.</td>
<td>New York</td>
</tr>
</tbody>
</table>

To learn more about Configuration Management Database, see [Configuration Management Database](#).

**Continual Improvement Management**

Continual Improvement Management quick start tests require activating the Continual Improvement Management Automated Tests plugin (com.sn_cim.aff)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIM: Inbound integration from Assessments</td>
<td>Create an improvement initiative from an assessment.</td>
<td>Paris</td>
</tr>
<tr>
<td>CIM: Inbound integration from Incident Management</td>
<td>Create an improvement initiative from an incident.</td>
<td>Paris</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>--------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>CIM: Inbound integration from Problem Management</td>
<td>Create an improvement initiative from a problem.</td>
<td>Paris</td>
</tr>
<tr>
<td>CIM: Inbound integration from Survey Management</td>
<td>Create an improvement initiative from a survey.</td>
<td>Paris</td>
</tr>
</tbody>
</table>

To learn more about Continual Improvement Management, see Continual Improvement Management.

**Customer Service Management**

All Customer Service Management quick start tests require activating the Customer Service Management Demo Data plugin (com.snc.customerservice.demo). Some quick start tests also require activating the following plugins:

- Consumer Service Portal (com.glide.service-portal.consumer-portal)
- Customer Service Management for Orders (com.snc.csm.order)
- Customer Service Portal (com.glide.service-portal.customer-portal)
- Customer Service with Request Management (com.sn_cs_sm_request)
- Customer Service with Service Management (com.sn_cs_sm)
- Customer Service Household (com.snc.household)
- Business Location (com.snc.business_location)
- CSM Extension for Proxy Contacts (com.snc.csm_proxy_contacts)
- Case Playbook for Complaints (sn_complaint)
- Case Playbook for Onboarding (sn_onboarding)
- Major Issue Management (com.sn_majorissue_mgt)
- Proactive Customer Service Operations with Event Management (com.snc.proactive_cs_itom)
- Skill Determination (com.snc.skill_determination)

**CSM: Case Management test suite**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSM: Create Product Case</td>
<td>Create a case for a product.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>CSM: Assign Case to an Agent</td>
<td>Create a case and assign it to a customer service agent.</td>
<td>Madrid</td>
</tr>
<tr>
<td>CSM: Assign Asset on Case</td>
<td>Assign an asset to a case.</td>
<td>Madrid</td>
</tr>
<tr>
<td>CSM: Assign Entitlement</td>
<td>Assign an entitlement to a case.</td>
<td>Madrid</td>
</tr>
<tr>
<td>CSM: Escalate an Account</td>
<td>Escalate an account.</td>
<td>Madrid</td>
</tr>
<tr>
<td>CSM: Escalate a Case</td>
<td>Escalate a case.</td>
<td>Madrid</td>
</tr>
<tr>
<td>CSM: Create Special Handling Notes</td>
<td>Create special handling notes for a case.</td>
<td>Madrid</td>
</tr>
<tr>
<td>CSM: Close a Case</td>
<td>Close a case.</td>
<td>Madrid</td>
</tr>
<tr>
<td>CSM: Time Recording</td>
<td>Record the time worked on a case.</td>
<td>Madrid</td>
</tr>
<tr>
<td>CSM: Create CHG from Case</td>
<td>Create a change record from a case.</td>
<td>Madrid</td>
</tr>
<tr>
<td>CSM: Create Incident from Case</td>
<td>Create an incident record from a case.</td>
<td>Madrid</td>
</tr>
<tr>
<td>CSM: Create Order Case</td>
<td>Create a case for an order.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Note: Requires Customer Service Management for Orders.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSM - Create Order Case as Customer from CSM Portal</td>
<td>Create an order case as a customer from the Customer Service Portal.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>
### CSM: Case Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong> Requires Customer Service Portal. Also requires that the test be run as admin.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSM: Create Problem from Case</td>
<td>Create a problem record from a case.</td>
<td>Madrid</td>
</tr>
<tr>
<td>CSM - Create Proactive Case by NOC Operator</td>
<td>Verify whether a proactive case is created.</td>
<td>New York</td>
</tr>
<tr>
<td><strong>Note:</strong> Requires Proactive Customer Service Operations with Event Management.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSM - Employee creating case OBO customer</td>
<td>As an employee with the proxy contact role (sn_customerservice_proxy_contact), create a case from the self-service portal on behalf of a customer.</td>
<td>New York</td>
</tr>
<tr>
<td><strong>Note:</strong> Requires the CSM Extension for Proxy Contacts.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSM - Lookup Type Skill Determination Rule Test</td>
<td>Create a lookup type skill determination rule.</td>
<td>New York</td>
</tr>
<tr>
<td><strong>Note:</strong> Requires Skill Determination.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSM - Manager creating request OBO customer from CSM portal</td>
<td>As a user with the case manager role, create a case on behalf of a customer from the Customer Service Portal.</td>
<td>New York</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>--------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>CSM - Simple Type Skill Determination Rule Test</td>
<td>Create a simple type skill determination rule.</td>
<td>New York</td>
</tr>
<tr>
<td></td>
<td>Note: Requires Skill Determination.</td>
<td></td>
</tr>
<tr>
<td>CSM: Register New Case Type</td>
<td>Register a case type and verify the record is created.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CSM - Project Manager create Project for an Account</td>
<td>Project Manager creates project for an account.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td>Note: Requires Customer Project Management.</td>
<td></td>
</tr>
<tr>
<td>CSM - Project Manager identify Project Contact</td>
<td>Project manager identifies customers to a project.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td>Note: Requires Customer Project Management.</td>
<td></td>
</tr>
<tr>
<td>CSM - Project Manager create project task and assign to a customer</td>
<td>Project manager creates project task and assigns to a customer.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td>Note: Requires Customer Project Management.</td>
<td></td>
</tr>
<tr>
<td>CSM - Create Case from Project</td>
<td>Create a case from project.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CSM - Create Case from Project Task</td>
<td>Create a case from a project task.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>---------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>CSM: Create Task from Case</td>
<td>Create a case task from a case.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CSM: Agent Creating Request for Customer</td>
<td>As a customer service agent, create a request for a customer.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td>Note: Requires Customer Service Request Integration.</td>
<td></td>
</tr>
<tr>
<td>CSM - Agent Create Cases from a Project</td>
<td>As a customer service agent, create a case from a project.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td>Note: Requires Customer Project Management.</td>
<td></td>
</tr>
<tr>
<td>CSM - Agent Create Cases from a Project task</td>
<td>As a customer service agent, create a case from a project task.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td>Note: Requires Customer Project Management.</td>
<td></td>
</tr>
<tr>
<td>CSM - Agent Create Change Requests for a project</td>
<td>As a customer service agent, create a change request for a project.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CSM: Create Major Case and its Child Cases</td>
<td>Create a major case and the associated child cases for the customer accounts in the recipient list.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td>Note: Requires Major Issue Management.</td>
<td></td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>CSM: Advanced Type Skill Determination Rule Test</td>
<td>Create an advanced type skill determination rule.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires Skill Determination.</td>
<td></td>
</tr>
<tr>
<td>CSM - Service Contracts covered under Sold Product</td>
<td>Create sold products and service contracts and associate service contracts to a sold product. Verify the association between the active contracts and the sold product.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires Customer Service Install Base Management.</td>
<td></td>
</tr>
<tr>
<td>AWA - Create New Service Channel</td>
<td>Create a new service channel in the Advanced Work Assignment application.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires Advanced Work Assignment for CSM.</td>
<td></td>
</tr>
<tr>
<td>CSM-ITOM - Create Child Cases for Proactive Major Case</td>
<td>Create a child case for a proactive major case using recipient list.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires Proactive Customer Service Operations with Event Management.</td>
<td></td>
</tr>
</tbody>
</table>
### CSM: Case Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSM - Create Outage from Case</td>
<td>Validate if newly created outage is linked to a case.</td>
<td>Paris</td>
</tr>
</tbody>
</table>
| CSM: Create Sold Product on Household | Create a sold product on a household and its member.  

**Note:** Requires Customer Service Install Base Management and Customer Service Household with Load demo data enabled. | Quebec |
| CSM: View Health Status of Install Base Item from Account and Case pages | Validates the functionality of the **Refresh Install Base Health** button on the Account and Case record pages. | Quebec |
| CSM: Assign Case Task to Case Task Agent | The customer service agent creates and assigns a case task to a case task agent. | Rome |
| CSM: Case Task Agent views assigned Case Task | Verifies that the case task agent can view an assigned case task. | Rome |
| CSM: Case Task Agent Completes Assigned Task | The case task agent completes an assigned case task. | Rome |
| CSM: Report a knowledge gap from a case in Agent Workspace | Verify that a knowledge gap related to a case was reported in Agent Workspace | Rome |
### CSM: Case Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSM-Create a case using 'Create a case (POST)' API</td>
<td>Create a case using 'Create a case (POST)' API</td>
<td>Rome</td>
</tr>
<tr>
<td>CSM-Query a case using 'Query a case (GET)' API</td>
<td>Query a case using 'Query a case (GET)' API</td>
<td>Rome</td>
</tr>
<tr>
<td>CSM: OCS Manager creating New OSP</td>
<td>Verify that a new Outsourced Service Provider (OSP) is created by OCS internal manager.</td>
<td>Rome</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires Outsourced Customer Service.</td>
<td></td>
</tr>
<tr>
<td>CSM: Case Creation by OCS Agent</td>
<td>Verify that an Outsourced Customer Service agent is able to create a case.</td>
<td>Rome</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires Outsourced Customer Service.</td>
<td></td>
</tr>
</tbody>
</table>

### CSM: Case Types - Complaint

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a Complaint Case</td>
<td>Tests that a user can create a case that is of type complaint in the sn_complaint_case table.</td>
<td>Paris</td>
</tr>
<tr>
<td>Escalate a Complaint Case</td>
<td>Tests that a user can escalate a case that is of type complaint in the sn_complaint_case table.</td>
<td>Paris</td>
</tr>
</tbody>
</table>
### CSM: Case Types - Onboarding

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create an Onboarding Case</td>
<td>Tests that a user can create a case that is of type onboarding in the sn_onboarding_case table.</td>
<td>Paris</td>
</tr>
<tr>
<td>Escalate an Onboarding Case</td>
<td>Tests that a user can escalate a case that is of type onboarding in the sn_onboarding_case table.</td>
<td>Paris</td>
</tr>
</tbody>
</table>

### CSM: Operations Dashboard test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>awa_admin_operations_dashboard</td>
<td>Verify whether user with role awa_admin is able to view Advanced Work Assignment menu under Operations Dashboard and unassigned interactions and unassigned task work items modules.</td>
<td>Orlando</td>
</tr>
<tr>
<td>awa_manager_operations_dashboard</td>
<td>Verify whether user with role awa_manager is able to view Advanced Work Assignment menu under Operations Dashboard and unassigned interactions and unassigned task work items modules.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
# CSM: Portal test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSM - Create Product Case as Customer from CSM Portal</td>
<td>Create a product case as a customer from the Customer Service Portal.</td>
<td>Madrid</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires Customer Service Portal. Also requires that the test be run as admin.</td>
<td></td>
</tr>
<tr>
<td>CSM - Create Product Case as Partner from CSM Portal</td>
<td>Create a product case as a partner from the Customer Service Portal.</td>
<td>Madrid</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires Customer Service Portal. Also requires that the test be run as admin.</td>
<td></td>
</tr>
<tr>
<td>CSM - Search on Homepage</td>
<td>Search for information from the Customer Service Portal. The search includes cases, Knowledge articles, and Community threads.</td>
<td>New York</td>
</tr>
<tr>
<td>CSM - Update Support Profile</td>
<td>Update a contact's profile from the Customer Service Portal.</td>
<td>New York</td>
</tr>
<tr>
<td>CSM - Provide requested info on case</td>
<td>From the Customer Service Portal, the contact can provide information for a case that was requested by the agent.</td>
<td>New York</td>
</tr>
<tr>
<td>CSM - Accept Proposed Solution On Case</td>
<td>Accept a proposed solution for a case from the Customer Service Portal.</td>
<td>New York</td>
</tr>
</tbody>
</table>
### CSM: Portal test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSM - Provide Feedback on Survey</td>
<td>Provide feedback on a survey after a case is closed from the Customer Service Portal.</td>
<td>New York</td>
</tr>
<tr>
<td>CSM - View All Desktop Notifications</td>
<td>View all Customer Service Management specific desktop notifications.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CSM - View Publications on CSM Portal</td>
<td>View publications on Customer Service Management portal.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CSM - Create Contact on CSM Portal</td>
<td>Create contacts on Customer Service Management portal.</td>
<td>Orlando</td>
</tr>
<tr>
<td>CSM: Validate Outage widgets in CSM Portal</td>
<td>Validates various types of outages and the corresponding widgets shown on the Customer Service Portal home page and the Install Base page.</td>
<td>Quebec</td>
</tr>
</tbody>
</table>

### CSP: Portal test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSP - Create Product Case as Consumer from CSP Portal</td>
<td>Create a product case as a consumer from the Consumer Service Portal.</td>
<td>Madrid</td>
</tr>
<tr>
<td><strong>Note:</strong> Requires Consumer Service Portal. Also requires that the test be run as admin.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSP - Search on Homepage</td>
<td>Search for information from the Consumer</td>
<td>New York</td>
</tr>
</tbody>
</table>
### CSP: Portal test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Portal. The search includes cases, Knowledge articles, and Community threads.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSP - Update Support Profile</td>
<td>Update a consumer’s profile from the Consumer Service Portal.</td>
<td>New York</td>
</tr>
<tr>
<td>CSP - Register Your Product</td>
<td>Register a product from the Consumer Service Portal.</td>
<td>New York</td>
</tr>
<tr>
<td>CSP - Provide requested info on case</td>
<td>From the Consumer Service Portal, the consumer can provide information for a case that was requested by the agent.</td>
<td>New York</td>
</tr>
<tr>
<td>CSP - Accept Proposed Solution On Case</td>
<td>Accept a proposed solution for a case from the Consumer Service Portal.</td>
<td>New York</td>
</tr>
<tr>
<td>CSP - Provide Feedback on Survey</td>
<td>Provide feedback on a survey after a case is closed from the Consumer Service Portal.</td>
<td>New York</td>
</tr>
<tr>
<td>CSP - View Publications on CSP Portal</td>
<td>View publications on Customer Service portal.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

### TC: Targeted Communications test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC - Create Recipient List</td>
<td>Create a recipient list with the required parameters. Verify the new recipient list in the related list “Recipients”.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
**TC: Targeted Communications test suite (continued)**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC - Create Publication</td>
<td>Create a publication. The publication is published based on the publication date and verify if an user in the recipient list gets the publication.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

**CSM Agent Workspace test suite**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSM - Create Problem from Case in Workspace</td>
<td>Tests the creation of a problem record from a case by a customer service agent in CSM Agent Workspace.</td>
<td>Quebec</td>
</tr>
<tr>
<td>CSM - Escalate a Case from Agent Workspace</td>
<td>Escalates a customer service case from CSM Agent Workspace.</td>
<td>Quebec</td>
</tr>
<tr>
<td>CSM - Create a Case and Propose Solution</td>
<td>Creates a customer service case and proposes a solution for the case from CSM Agent Workspace.</td>
<td>Quebec</td>
</tr>
<tr>
<td>CSM - Create Case and Assign this Case to Agent from Agent Workspace</td>
<td>Verifies the ability to create a customer service case and assign the case to an agent in CSM Agent Workspace.</td>
<td>Quebec</td>
</tr>
<tr>
<td>CSM - Approve Major Case Candidate from Agent Workspace</td>
<td>Verifies the ability to approve a major case candidate on a case from CSM Agent Workspace.</td>
<td>Quebec</td>
</tr>
</tbody>
</table>
CSM Agent Workspace test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSM - Create Child Case for a Major Case from Agent Workspace</td>
<td>Verifies the ability to create a child case from a major case using the recipient list from CSM Agent Workspace.</td>
<td>Quebec</td>
</tr>
<tr>
<td>CSM - Create a Business Location from Agent Workspace</td>
<td>Creates a new business location in Agent Workspace.</td>
<td>Rome</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires Business Location.</td>
<td></td>
</tr>
<tr>
<td>CSM - Create Household in Workspace</td>
<td>Creates a new household in Agent Workspace.</td>
<td>Rome</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires Customer Service Household.</td>
<td></td>
</tr>
<tr>
<td>CSM - Create Knowledge Gap from Case</td>
<td>Creates a knowledge gap from a case in Agent Workspace.</td>
<td>Rome</td>
</tr>
</tbody>
</table>

To learn more about Customer Service Management, see [Customer Service Management](#).

**Dashboards**

Dashboards quick start tests require activating the Automated Test Framework - Responsive Dashboards plugin (com.glide.automated_testing_impl.dashboards). This plugin is active on zboot of the instance.

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Responsive Dashboard Sharing</td>
<td>Confirm dashboard sharing by impersonating users.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Responsive Dashboard Visibility</td>
<td>Confirm dashboard visibility by impersonating users.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>

To learn more about Dashboards, see [Dashboards](#).

**DevOps**

DevOps quick start tests are available when you install the DevOps application from ServiceNow Store.

**DevOps test suite**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>DevOps Orchestration Flow with CR</td>
<td>Verify the Flow Designer flow for a DevOps orchestration tool that includes a change request.</td>
<td>Madrid</td>
</tr>
<tr>
<td>DevOps Orchestration Tool Flow</td>
<td>Verify the Flow Designer flow for a DevOps orchestration tool.</td>
<td>Madrid</td>
</tr>
<tr>
<td>DevOps Plan Tool Flow</td>
<td>Verify the Flow Designer flow for a DevOps planning tool.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>

To learn more about DevOps, see [DevOps](#).

**Essential SAFe**

Essential SAFe quick start tests require activating the Agile - Scaled Agile Framework - Essential SAFe plugin (com.snc.sdlc.safe) and the Agile - Scaled Agile Framework - Essential SAFe - ATF Tests plugin (com.snc.sdlc.safe.atf).
## Essential SAFe test suites

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Essential SAFe: Feature tests</td>
<td>Verify feature global rank updates.</td>
<td>Madrid</td>
</tr>
<tr>
<td></td>
<td>For a SAFe feature, verify that:</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td>• Actual start date is populated after the state is changed to Implementation, Validation on Staging, or Deployment.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Actual end date is populated after the state is changed to <strong>Released</strong> or <strong>Cancelled</strong>.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Active flag is set to the appropriate value:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ <strong>False</strong>, if the state is changed to <strong>Released</strong> or <strong>Cancelled</strong>.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ <strong>True</strong>, for all other states.</td>
<td>Quebec</td>
</tr>
<tr>
<td></td>
<td>If the PPM Standard plugin (com.snc.financial_planning_pmo) is active, verify that a Demand can be converted to a SAFe feature using the <strong>Create SAFe Feature</strong> related link on the Demand form.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information on how to create an SAFe feature from a demand,</td>
<td></td>
</tr>
</tbody>
</table>
## Essential SAFe test suites (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>see Create an artifact from a demand.</td>
<td>See Create an artifact from a demand.</td>
<td></td>
</tr>
<tr>
<td>Essential SAFe: Program increment tests</td>
<td>Verify program increment date overlapping.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Essential SAFe: Sprint tests</td>
<td>Verify the generation of ART sprints and team sprints as well as updates to sprint points and dates.</td>
<td>Madrid</td>
</tr>
<tr>
<td></td>
<td>Verify that any update to the <strong>Group capacity</strong> field of the assignment group results in the following changes to the <strong>Group capacity</strong> field of the various sprints associated with this assignment group:</td>
<td>Paris</td>
</tr>
<tr>
<td></td>
<td>• For the sprints that are in the <strong>Draft</strong> or <strong>Planning</strong> states:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ The group capacity is updated to the new value.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ The <strong>Group capacity</strong> field is editable.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• For the sprints in the <strong>Current</strong>, <strong>Complete</strong>, or <strong>Cancelled</strong> states:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ The group capacity remains the old value.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ The <strong>Group capacity</strong> field is read-only.</td>
<td></td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Essential SAFe: Story tests</td>
<td>For the sprints in the <strong>Draft</strong> or <strong>Planning</strong> state, you can individually edit the group capacity of the sprint anytime later. This would not change the group capacity of the assignment group associated with this sprint.</td>
<td></td>
</tr>
<tr>
<td>Essential SAFe: Story tests</td>
<td>Verify story global rank updates.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>
| Essential SAFe: Story tests | Verify that active flag of the SAFe story is set to the appropriate value:  
   - **False**, if the state of the state is changed to Completed or Cancelled.  
   - **True**, for all other states. | Orlando |
| Essential SAFe: Story tests | Verify that adding, estimating, removing, deleting, updating, or cancelling a SAFe story updates the SAFe feature-level and then the epic-level roll-ups correctly. | |
| Essential SAFe: Story tests | Verify that adding, updating, or deleting the feature on a SAFe story updates the Epic field on the SAFe story form. | Quebec |
| Essential SAFe: Story tests | If the PPM Standard plugin (com.snc.financial_planning_pmo) | |
## Essential SAFe test suites (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>is active, verify that a Demand can be converted to a SAFe story using the <strong>Create SAFe Story</strong> related link on the Demand form. For more information on how to create a SAFe story from a demand, see <strong>Create an artifact from a demand</strong>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Essential SAFe: Team tests</td>
<td>Verify team association with an ART.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>
| Essential SAFe: Epic tests | For a SAFe epic, verify that:  
  - Actual start date is populated after the state is changed to **Implementation**.  
  - Actual end date is populated after the state is changed to **Complete**.  
  - Active flag is set to the appropriate value:  
    - **False**, if the state is changed to **Released** or **Cancelled** states.  
    - **True**, for all other states. | Orlando |
<p>| If the PPM Standard plugin (com.snc.financial_planning_pmo) is active, verify that a Demand can be | Quebec |</p>
<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>converted to a SAFe epic using the <strong>Create SAFe Epic</strong> related link on the Demand form. For more information on how to create a SAFe epic agile from a demand, see <strong>Create an artifact from a demand.</strong></td>
<td></td>
</tr>
</tbody>
</table>
| Essential SAFe: Program PI Objective tests | Verify the functionality of creating and updating PI objectives  
• Verify that you can create a program increment-level objective  
• Verify that you can update the created PI objective with the planned business value (PBV) and actual business value (ABV)  
• Verify that the percentage of business value achieved is computed as the percentage of ABV complete, for only the committed PI objectives, as compared to the PBV of the program in the PI | Rome |
| Essential SAFe: Team PI Objective tests | Verify the functionality of creating and updating team PI Objectives | Rome |
Essential SAFe test suites (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Verify that you can create a team-level PI objective</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Verify that you can update the created PI objective with the planned business value (PBV) and actual business value (ABV)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Verify that the percentage of business value achieved is computed as the percentage of ABV complete, for only the committed PI objectives, as compared to the PBV of the team in the PI</td>
<td></td>
</tr>
</tbody>
</table>

To learn more about Essential SAFe, see Essential SAFe.

**Event Management**

Event Management quick start tests require activating the Event Management plugin (com.glideapp.itom.snac).

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMSelfMonitoring</td>
<td>Track problems with the Event Management plugin after upgrade. To understand the details of issues found, open the All Alerts list and search for alerts that failed, whose source is EMSelfMonitoring, and that were opened or</td>
<td>Madrid</td>
</tr>
</tbody>
</table>
reopened since the upgrade. The exact time for filtering can be found in the error message.

To learn more about Event Management, see Event Management.

Field Service Management

Field Service Management quick start tests are available when you enable the Field Service Management plugin (com.snc.work_management). Enable the demo data plugin (com.snc.work_management.demo) in a non-production instance to start using the quick start tests available with your application. You can also modify existing data and customize it to run the quick tests.

FSM: Field Service Management test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
</table>
| FSM: Create Initiate Qualify Dispatch and assign Work Order Task | • Create a work order.  
• Initiate and qualify a work order.  
• Dispatch a work order.  
• Assign a work order task to an agent. | Madrid |
| FSM: Part Sourcing | • Source any part to an agent’s stock room from the work order task.  
• Create a transfer order line for the part sourcing.  
• Use this part for any work order to consume it using the **Part Usage** action. | Madrid |
<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
</table>
| FSM: Part Usage               | • Source any part to an agent's stock room from the work order task.  
• Use this part for any work order to consume it using the **Part Usage** action.                                                                                                                                          | Madrid          |
| FSM: Questionnaire            | Create a questionnaire and associate it with a work order task.                                                                                                                                               | Madrid          |
| FSM: Field Service Configuration | Verify that the default configuration is preserved. The default configuration includes the task assignment method, qualification requirement, PDF summary, and agent's ability to accept or reject tasks.                 | New York        |
| FSM: Planned Maintenance      | Create a planned maintenance work order with weekly interval time for printer maintenance.                                                                                                                   | New York        |
| FSM: Appointment Booking Configuration | • Verify that the default configuration for appointment booking is preserved.  
• Verify that the point of sale service and that the catalog item exist in the system.                                                                                 | New York        |
<p>| FSM: Create Work or Personal Schedules | Create personal or work schedule for agents.                                                                                                                                                               | Orlando         |</p>
<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSM: Field Service Property Settings</td>
<td>Verify that the field service system properties preserve expected values.</td>
<td>Orlando</td>
</tr>
<tr>
<td>FSM: Work Groups</td>
<td>Verify that field service agents can be added to work groups.</td>
<td>Orlando</td>
</tr>
<tr>
<td>FSM: Dynamic Scheduling - Preferred Technician assignment with mandatory parts reservation</td>
<td>With Dynamic scheduling:&lt;br&gt;• Preferred technician should be picked for task assignment.&lt;br&gt;• Mandatory parts should be reserved in the agent stock room.</td>
<td>Paris</td>
</tr>
<tr>
<td>Note:</td>
<td>Activate Customer service management demo data plugin.</td>
<td></td>
</tr>
<tr>
<td>FSM: Create Time Card</td>
<td>Verify that a time card is created for an agent in the work order task.</td>
<td>Quebec</td>
</tr>
<tr>
<td>FSM: Create incidental</td>
<td>Verify that an incidental is created for an agent in the work order task.</td>
<td>Quebec</td>
</tr>
<tr>
<td>FSM: Onboarding contractor company</td>
<td>Verify that a contractor company is onboarded with an assignment group, an external manager, and an external agent.</td>
<td>Quebec</td>
</tr>
<tr>
<td>FSM: Onboarding contractor agent by external manager</td>
<td>Verify that an external contractor manager can onboard external agents from the Field</td>
<td>Quebec</td>
</tr>
</tbody>
</table>
### FSM: Field Service Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSM: External contractor manager fulfil the work order task</td>
<td>Verify that a contractor manager or an agent is able to fulfil the assigned work order task.</td>
<td>Quebec</td>
</tr>
<tr>
<td>FSM: Assign work order task to Vendor group</td>
<td>Verify that a work order task is assigned to the external assignment group based on the defined criteria, such as task location and configuration parameters.</td>
<td>Quebec</td>
</tr>
<tr>
<td>FSM: Pause and Resume work order task</td>
<td>Verify that an agent can pause and resume work for a work order tasks.</td>
<td>Rome</td>
</tr>
<tr>
<td>FSM: Dynamic Scheduling - Assign technician with matching skills</td>
<td>With Dynamic Scheduling, verify that a work order task is assigned to a technician who possesses all mandatory skills mentioned in the task.</td>
<td>Rome</td>
</tr>
<tr>
<td>FSM: Off boarding contractor agent by external manager</td>
<td>Verify that an external contractor manager can offboard external agents from the Field Service Contractor Management portal.</td>
<td>Rome</td>
</tr>
</tbody>
</table>

To learn more about Field Service Management, see Field Service Management.

### Finance Close Automation

Finance Close Automation quick start tests are available when you install the Finance Close Automation application from ServiceNow Store.
<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCA: Applicable Months sub-set tests</td>
<td>Verify that the Applicable months can only be sub-set of functional workbook.</td>
<td>New York</td>
</tr>
<tr>
<td>FCA: dates validations</td>
<td>Verify that the date related calculations are correct.</td>
<td>New York</td>
</tr>
<tr>
<td>FCA: task relationships</td>
<td>Verify that dates and states are updated correctly for the tasks having relationships.</td>
<td>New York</td>
</tr>
<tr>
<td>FCA: relationship with parent tasks</td>
<td>Verify that dates and states are updated correctly for tasks that have relationship with parent tasks.</td>
<td>New York</td>
</tr>
<tr>
<td>FCA: Reset Task</td>
<td>Verify the FCA workflow when the task is reset.</td>
<td>New York</td>
</tr>
<tr>
<td>FCA: Reject Task</td>
<td>Verify the FCA workflow when the task is rejected.</td>
<td>New York</td>
</tr>
<tr>
<td>FCA: Create JE Task</td>
<td>Verify that ERP source is available for a Journal Entry task type.</td>
<td>New York</td>
</tr>
<tr>
<td>FCA: Negative values as close day</td>
<td>Verify that both the positive and negative values are acceptable for close day and the planned dates are calculated correctly.</td>
<td>New York</td>
</tr>
<tr>
<td>FCA: Close day checks for Daily close Checklist</td>
<td>Close day verifications in daily close workbook.</td>
<td>New York</td>
</tr>
<tr>
<td>FCA: Daily close planned end date checks</td>
<td>Verify planned end dates in a daily close workbook.</td>
<td>New York</td>
</tr>
</tbody>
</table>
## FCA test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCA: Kickstart errors</td>
<td>Verify that all the kickstart validations run.</td>
<td>New York</td>
</tr>
<tr>
<td>FCA: One final task verification</td>
<td>Verify that there is only one final task in each functional workbook.</td>
<td>New York</td>
</tr>
<tr>
<td>FCA: Confidential task</td>
<td>Verify that only the owner, reviewer, and approver of a confidential task can view the task.</td>
<td>New York</td>
</tr>
<tr>
<td>FCA: Unique checklist Verification</td>
<td>Verify that only one workbook is created for a specific day or period.</td>
<td>New York</td>
</tr>
<tr>
<td>FCA: Milestone and JE Validations</td>
<td>Verify that milestone and journal entry tasks don't have any child tasks.</td>
<td>New York</td>
</tr>
</tbody>
</table>

To learn more about Finance Close Automation, see [Finance Close Automation](#).

### Financial Management

Financial Management quick start tests require activating the Financial Management Core - ATF Tests plugin (com.snc.financial_management.atf)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verify FM Cost Allocation Flow</td>
<td>Verify the cost allocation flow in financial modeling.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>

To learn more about Financial Management, see [Financial Management](#).

### GRC: Audit Management

GRC: Audit Management quick start tests require activating the GRC: Audit Management plugin (com.sn_audit) and loading the demo data.
## GRC: Audit Management test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRC: Create Audit Engagement and Generate Audit Task</td>
<td>Validates audit engagement creation and associates entities to generate controls and test plans. Generates audit task which is associated to a test plan.</td>
<td>Paris</td>
</tr>
<tr>
<td>GRC: Create and process a milestone</td>
<td>Create a milestone in an engagement, notice that the due date cannot be in past and the completion date cannot be in future and for a milestone in open state the percent complete is 0 which changes in accordance with the milestone state change.</td>
<td>Paris</td>
</tr>
<tr>
<td>GRC: Cost and Resource plan rollup</td>
<td>Create an audit plan and associate an engagement to it, on adding cost plan and resource plan to this engagement, notice that these costs are rolled up to the plan. Any edits to these costs in engagement reflects in the plan.</td>
<td>Paris</td>
</tr>
<tr>
<td>GRC: Create Engagement Project Manual and automatic</td>
<td>On an engagement in the validate state, once the Enable advanced planning ui-action is performed notice an engagement project gets created and when the state of an</td>
<td>Paris</td>
</tr>
</tbody>
</table>
### GRC: Audit Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>engagement associated to an audit plan having &quot;Advanced planning capabilities&quot; is changed to validate notice that an engagement project gets created automatically.</td>
<td></td>
</tr>
<tr>
<td>GRC: Auditable Unit with Detailed Risk Assessment</td>
<td>Create an Auditable unit with method as &quot;Detailed Risk Assessment&quot; and request for Assessing the Risk Assessment by adding the Assessor once the assessor responds and Marks Assessment as Complete after performing the control assessment and residual assessment, the risk assessment fields should be auto updated.</td>
<td>Paris</td>
</tr>
</tbody>
</table>

To learn more about Audit Management, see [Audit Management](#).

### GRC: Continuous Authorization and Monitoring

GRC: Continuous Authorization and Monitoring quick start tests require activating the Continuous Authorization and Monitoring plugin (com.sn_compliance) and loading the demo data.

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRC: System Owner create and validate responsibilities and roles for an AB and AP</td>
<td>System Owner creates and validates responsibilities and roles for an Authorization</td>
<td>Quebec (compatible with Paris and Orlando)</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Boundary and Authorization Package. Information Owners and System User are pre-populated when selecting the Authorization Boundary.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| GRC: System Owner validate App Modules visibility | Verifies that the system owner persona is able to view the following application menu:  
• Continuous Authorization & Monitoring and the following modules under that menu:  
• All Authorization Boundaries  
• All Authorization Packages  
• Information Type Library  
• Control Overlays  
• Control Objectives  
• Controls  
• All Engagements | Quebec (compatible with Paris and Orlando) |
| GRC: System Owner Request First approval & My approvals module | System Owner requests an approval. | Quebec (compatible with Paris and Orlando) |
| SO: Create and validate responsibilities and roles for an AB and AP | Verifies if a system owner can create an Authorization Boundary by completing the fields | Quebec (compatible with Paris and Orlando) |
GRC: Continuous Authorization and Monitoring Quick Start Tests test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>on the Authorization Boundary form. Also verify if the same SO can create an Authorization Package from the form view.</td>
<td></td>
</tr>
</tbody>
</table>

To learn more about Continuous Authorization and Monitoring, see Continuous Authorization and Monitoring.

GRC: Policy and Compliance Management

GRC: Policy and Compliance Management quick start tests require activating the Policy and Compliance Management plugin (com.sn_compliance) and loading the demo data.

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRC: Create Controls</td>
<td>Validates control creation.</td>
<td>Madrid</td>
</tr>
<tr>
<td>GRC: Create Policy Exception and Flow</td>
<td>Create a policy exception and navigate through its states.</td>
<td>Paris</td>
</tr>
<tr>
<td>GRC: Policy Lifecycle</td>
<td>Create a policy and navigate through its states.</td>
<td>Paris</td>
</tr>
</tbody>
</table>

To learn more about Policy and Compliance Management, see Policy and Compliance Management.

GRC: Risk Management

GRC: Risk Management quick start tests require activating the Risk Management plugin (com.sn_risk) and loading demo data.
<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRC: Create Profile</td>
<td>Validate profile creation.</td>
<td>Madrid</td>
</tr>
<tr>
<td>GRC: Create Risk</td>
<td>Validates risk creation.</td>
<td>Madrid</td>
</tr>
<tr>
<td>GRC: Create Control</td>
<td>Validates control creation.</td>
<td>New York</td>
</tr>
<tr>
<td>GRC: Create Control Objective and Generate Controls</td>
<td>Validates control objective creation and associates profiles to generate controls.</td>
<td>New York</td>
</tr>
<tr>
<td>GRC: Create Issue</td>
<td>Validates issue creation</td>
<td>New York</td>
</tr>
<tr>
<td>GRC: Accept an Issue</td>
<td>Validates all the states of an issue till it is closed by accepting the issue.</td>
<td>New York</td>
</tr>
<tr>
<td>GRC: Remediate an Issue</td>
<td>Validates issue cannot be closed be with an open remediation task.</td>
<td>New York</td>
</tr>
<tr>
<td>GRC: ATF Flow for Indicator (Manual):</td>
<td>Create an indicator template with type manual and associate a control objective to it. Execute one of the indicators thus formed. Mark the state of the indicator task created as closed and result as failed. The associated control will become non-compliant and an issue will be generated.</td>
<td>New York</td>
</tr>
<tr>
<td>GRC: ATF flow for Indicator (Basic)</td>
<td>Create an indicator template with type basic and result as failed. Associate a control objective to it and give the supporting data. Execute one</td>
<td>New York</td>
</tr>
</tbody>
</table>
### GRC: Risk Management Quick Start Tests test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>of the indicators thus formed. The associated control will become non-compliant and an issue will be generated.</td>
<td>GRC: ATF flow for indicator (Script) Create an indicator template with type script and enter a script and set the value of result.passed and result.value. Associate a control objective to it. Execute one of the indicators. The associated control will become non-compliant and an issue will be generated.</td>
<td>New York</td>
</tr>
</tbody>
</table>

To learn more about Risk Management, see [Risk Management](#).

### Hardware Asset Management

Hardware Asset Management quick start tests are available when you install the HAM: Hardware Asset Management application from the ServiceNow Store.

**Hardware Asset Management test suite**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asset Lifecycle Automation - Asset Disposal workflow</td>
<td>Validates asset disposal work flow.</td>
<td>Paris</td>
</tr>
<tr>
<td>Asset Lifecycle Automation-Deployment workflow</td>
<td>Validates asset deployment work flow.</td>
<td>Paris</td>
</tr>
</tbody>
</table>
## Hardware Asset Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image_url" alt="Image" /> Note: Requires demo data.</td>
<td><img src="image_url" alt="Image" /></td>
<td>Quebec</td>
</tr>
<tr>
<td>Asset Lifecycle Automation - Hardware Asset Refresh workflow</td>
<td>Validates asset refresh workflow.</td>
<td>Quebec</td>
</tr>
<tr>
<td>Asset Lifecycle Automation - Loaner Asset workflow</td>
<td>Validates loaner asset workflow.</td>
<td>Quebec</td>
</tr>
<tr>
<td>Asset Lifecycle Automation - Leased Asset Expiration end to end workflow</td>
<td>Validates leased asset expiration end to end workflow.</td>
<td>Quebec</td>
</tr>
<tr>
<td>Asset Lifecycle Automation - RMA</td>
<td>Validates RMA workflow.</td>
<td>Rome</td>
</tr>
<tr>
<td>Asset Lifecycle Automation - Asset Reservations</td>
<td>Validates loaner asset workflow.</td>
<td>Rome</td>
</tr>
</tbody>
</table>

To learn more about Hardware Asset Management, see [Hardware Asset Management](#).

### HR Service Delivery

#### HR Service Delivery case tests

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>HR: Case updates visible to employees</td>
<td>Cases created and updated by an HR agent are visible to the Opened for or Subject person</td>
<td>Quebec</td>
</tr>
<tr>
<td>HR: Creation of 401k enrollment case from portal</td>
<td>Creates a 401(k) enrollment case from the Employee Service Center portal.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>HR: Creation of Employee Relations case from ESC portal</td>
<td>Creates a Disciplinary Issue case from the Employee Service Center portal. Verifies that the case creates successfully.</td>
<td>New York</td>
</tr>
<tr>
<td>HR: Creation of Employment Verification Letter case from ESC portal</td>
<td>Creates an Employment Verification Letter case from the Employee Service Center. Verifies that the case creates successfully.</td>
<td>Madrid</td>
</tr>
<tr>
<td>HR: Creation of Payroll case from ESC portal</td>
<td>Creates a direct deposit payroll setup case from the Employee Service Center or service portal. Verifies that the case creates successfully.</td>
<td>New York</td>
</tr>
<tr>
<td>HR: Creation of Tuition Reimbursement request from portal</td>
<td>Creates a Tuition Reimbursement Request case from the Employee Service Center portal. Verifies that the case creates successfully.</td>
<td>Madrid</td>
</tr>
<tr>
<td>HR: Fulfillment Instructions</td>
<td>Verifies that fulfillment instructions are created for a specific condition for an opened for user. Also verifies the fulfillment instructions change when the details of the case changes.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
## HR Service Delivery case tests (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>HR: Fulfillment Instructions conditions for COE specific fields</td>
<td>Verifies the conditions for COE specific fulfillment instruction. Fulfillment instructions update on an HR case based on the conditions defined in the fulfillment instructions.</td>
<td>Quebec</td>
</tr>
<tr>
<td>HR: General Benefits Inquiry Case Creation</td>
<td>Creates a General Benefits case using the native UI. It also verifies that the case opens and can be updated after case creation.</td>
<td>Madrid</td>
</tr>
<tr>
<td>HR: Payroll Discrepancy Case Creation</td>
<td>Creates a Payroll Discrepancy case using the native UI. It also verifies that the case opens and can be updated after case creation.</td>
<td>Madrid</td>
</tr>
<tr>
<td>HR: Reclassify Case Transfer</td>
<td>Tests when an HR case transfers from one COE (HR service) to a different COE using the reclassify (HR case number remains the same) method.</td>
<td>Orlando</td>
</tr>
<tr>
<td>HR: Response Template Configuration</td>
<td>Creates a response template for a payroll case and pastes the response in the worknotes.</td>
<td>Quebec</td>
</tr>
<tr>
<td>HR: Search catalog items &amp; KBs in ESC</td>
<td>Verifies the search functionality in the</td>
<td>Orlando</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Employee Service Center.</td>
<td></td>
<td>Orlando</td>
</tr>
<tr>
<td>HR: Standard Case Transfer</td>
<td>Tests an HR case transfers from one COE (HR service) to a different COE using the standard (creates HR case number and deletes old number) case transfer.</td>
<td>Orlando</td>
</tr>
<tr>
<td>HR: Suspend Case Flow</td>
<td>Tests the ability to suspend an HR case and later resume the case.</td>
<td>Rome</td>
</tr>
<tr>
<td>HR: Tuition Reimbursement Case Creation</td>
<td>Creates a Tuition Reimbursement HR case.</td>
<td>Orlando</td>
</tr>
<tr>
<td>HR: Workforce Administration Case Creation</td>
<td>Creates a Workforce Administration case, signs an employee verification letter, and generates the letter.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Create HR Employee Document</td>
<td>Creates an employee document and ensures the Attachment [sys_attachment] table points back to the record in the Employee Document [sn_hr_ef_employee_document] table. Verifies that the employee document is accessible from the Employee Service Center via the employee HR profile.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>
**Note:** Requires plugin activation of:

- Human Resources Scoped App: Core plugin (com.sn_hr_core)
- Employee Service Center [com.sn_hr_service_portal]

### Employee Center test

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESC: Employee can see ticket updates</td>
<td>Employees can view updates to their tickets.</td>
<td>Quebec</td>
</tr>
<tr>
<td>ESC: Post General HR Inquiry questions on ESC portal</td>
<td>Verifies an HR employee can create a General Inquiry case and post general HR inquiry questions on the ESC.</td>
<td>Quebec</td>
</tr>
<tr>
<td>ESC: Submit a Record Producer which creates Universal Request and HR Case</td>
<td>Verifies a user can submit an HR catalog item that creates a Universal Request. Also verifies the Universal Request and HR case are created and linked.</td>
<td>Quebec</td>
</tr>
<tr>
<td>ESC: Verify Standard Ticket page on ESC for HR Case</td>
<td>Creates a general inquiry case from the service portal and verifies it appears on the standard Ticket page.</td>
<td>Quebec</td>
</tr>
<tr>
<td>ESC: Verify widget contents in Catalog items</td>
<td>Verifies the widget content in a Catalog page.</td>
<td>Orlando</td>
</tr>
<tr>
<td>ESC: Verify widget contents in knowledge pages</td>
<td>Verifies the widget content in a Knowledge page.</td>
<td>Orlando</td>
</tr>
<tr>
<td>HR: Search catalog items &amp; KBs in ESC</td>
<td>Verifies the search functionality in the ESC.</td>
<td>Quebec</td>
</tr>
</tbody>
</table>

**Note:** Requires plugin activation of:

- Human Resources Scoped App: Core plugin (com.sn_hr_core)
- Employee Service Center [com.sn_hr_service_portal]
### Content Publishing: Content Publishing tests

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Publishing: Audience</td>
<td>Tests the audience configurations for Content Delivery.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Content Publishing: Schedule Portal Content</td>
<td>Creates and schedules test content for the Content Delivery demo portal.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

**Note:** Requires plugin activation of:
- Content Delivery [com.sn_content_delivery]

### Content Experiences: Content Experience tests

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Experiences: Create and Publish a Campaign</td>
<td>Creates and publishes a campaign with portal content and verifies against the Content Delivery portal.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Content Experiences: Portal preview</td>
<td>Verifies that a campaign manager is able to preview portal content using the portal preview.</td>
<td>Rome</td>
</tr>
</tbody>
</table>

**Note:** Requires plugin activation of:
- Content Automation [com.sn_content_automation]

### Lifecycle Events: Tests for verifying Lifecycle Events

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Employee request and associate it to LE case</td>
<td>Creates and validates that an employee request appears in the Lifecycle Event case action list.</td>
<td>Rome</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>---------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>HR Lifecycle: Access Ticket page for a New Hire Onboarding Case in Employee Service Center</td>
<td>Verifies creation of a new hire and can access the Ticket page of the Employee Service Center.</td>
<td>Orlando</td>
</tr>
<tr>
<td>HR Lifecycle: Access ESC ticket page and todos page and complete assigned todos</td>
<td>Creates a Tuition Reimbursement Request case from Employee Service Center and the Complete NDA to-do from the subject person.</td>
<td>Orlando</td>
</tr>
<tr>
<td>HR LifeCycle: Request Onboarding Case Creation</td>
<td>Creates a Request Onboarding case using the native UI and updates the case after creation.</td>
<td>Madrid</td>
</tr>
<tr>
<td>HR Lifecycle: Trigger Rescind Workflow</td>
<td>Triggers the Rescind workflow for a New Hire Onboarding Lifecycle Event case.</td>
<td>Quebec</td>
</tr>
<tr>
<td>HR Lifecycle: Verify Requests page for Open and Closed cases</td>
<td>Verifies open and closed cases on the Requests page.</td>
<td>Orlando</td>
</tr>
<tr>
<td>HR Lifecycle: Verify search in Requests Page</td>
<td>Verifies the search functionality on the Requests page for open and closed cases.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Configure HR Service for Auto Case Closure</td>
<td>Verifies that Lifecycle event cases are automatically closed.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
Note: Requires plugin activation of:

- Human Resources Scoped App: Core plugin (com.sn_hr_core)
- Human Resources Scoped App: Lifecycle Events [com.sn_hr_lifecycle_events]
- Human Resources Scoped App: Lifecycle Events for Enterprise plugin [com.sn_hr_lifecycle_ent]
- Employee Service Center [com.sn_hr_service_portal]

**KB: Knowledge block tests**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create knowledge block and attach block to article</td>
<td>Creates a knowledge block, publishes the block, and attaches the block to a knowledge article.</td>
<td>New York</td>
</tr>
<tr>
<td>Preview Knowledge Articles with Knowledge Blocks</td>
<td>Tests for preview of knowledge article that contains knowledge blocks.</td>
<td>New York</td>
</tr>
</tbody>
</table>

Note: Requires plugin activation of:

- Human Resources Scoped App: Core plugin (com.sn_hr_core)

To learn more about HR Service Delivery, see [HR Service Delivery](#).

**Incident Management**

Incident Management quick start tests require activating the Incident Management - ATF Tests plugin (com.snc.incident.atf).

**Incident Management test suite**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCIDENT MGMT: Incident Resolution SLA</td>
<td>Test to verify the Incident Resolution SLA baseline functionality.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>INCIDENT MGMT: Incident Response SLA</td>
<td>Test to verify the Incident Response SLA baseline functionality.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Copy Incident</td>
<td>Test to verify whether the fields from the original Incident are copied correctly to the new Incident.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Copy Incident from a Closed Incident</td>
<td>Test to verify that the Copy Incident UI action is visible for closed Incidents.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Create Standard Change from Incident</td>
<td>Test to verify the creation of a Standard Change from an Incident.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Create Emergency Change from an Incident</td>
<td>Test to verify the creation of an Emergency Change from an Incident.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Create Normal Change from an Incident</td>
<td>Test to verify the creation of a Normal Change from an Incident.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Create Problem from an Incident</td>
<td>Test to verify the creation of a Problem from an Incident.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Create Knowledge from an Incident</td>
<td>Test to verify the creation of a Knowledge from an Incident.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Incident State flow</td>
<td>Test to verify the state flow of an incident.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Reopening an Incident</td>
<td>Test to verify the reopen incident functionality.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Incident Assignment</td>
<td>Test to verify the incident assignment functionality.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>
## Incident Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCIDENT MGMT: Create child Incident using UI action and verify its fields</td>
<td>Test to verify the creation of a child Incident from an Incident through the Create Child Incident UI action. The test also verifies that the fields of the child Incident get copied correctly from the parent incident to the child incident.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Incident creation — Self Service</td>
<td>Test to verify the creation of an Incident using the Create Incident catalog item.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Parent and Child Incident state sync up</td>
<td>Test to verify that the state of a child Incident synchronizes with the parent Incident when the child Incident is created.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Parent and child Incident state sync up after reopening an Incident</td>
<td>Test to verify that the state of a child Incident synchronizes with the parent Incident when the parent Incident is reopened.</td>
<td>Madrid</td>
</tr>
<tr>
<td>INCIDENT MGMT: Verify creation of knowledge article from an Incident</td>
<td>Test to verify the creation of a knowledge article from an Incident using the Create Knowledge UI action on the Incident form. The UI action is visible when you activate the KCS Integration for Incident Management plugin (com.snc.incident.knowledge).</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
To learn more about Incident Management, see [Incident Management](#).

## Integration Commons for CMDB

**CMDB INT: CMDB Integrations Validation test suite**

*Test suite to verify the integrity of an integration using multiple tests.*

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMDB INT: Set Test Session Application</td>
<td>Modify the run server-side script to set an application name so that you can test only one integration. Otherwise, all integrations installed will be tested.</td>
<td>Paris</td>
</tr>
<tr>
<td>CMDB INT: Test Against Source Analysis</td>
<td>Test an integration against the values in the CMDB Integration Source Analysis table.</td>
<td>Paris</td>
</tr>
<tr>
<td>CMDB INT: Validate Application Feed</td>
<td>Validate all application feeds in an integration.</td>
<td>Paris</td>
</tr>
<tr>
<td>CMDB INT: Validate Discovery Source</td>
<td>Validate that the discovery source exists.</td>
<td>Paris</td>
</tr>
<tr>
<td>CMDB INT: Validate Entity Mappings</td>
<td>Validate all entity mappings of an integration.</td>
<td>Paris</td>
</tr>
<tr>
<td>CMDB INT: Validate Fields</td>
<td>Validate fields for CMDB Integrations.</td>
<td>Paris</td>
</tr>
<tr>
<td>CMDB INT: Validate Lookups</td>
<td>Validate CMDB integration lookups.</td>
<td>Paris</td>
</tr>
<tr>
<td>CMDB INT: Validate Mandatory Operations</td>
<td>Validate that all integrations for mandatory operations exist for mapped fields.</td>
<td>Paris</td>
</tr>
</tbody>
</table>
CMDB INT: CMDB Integrations Validation test suite

Test suite to verify the integrity of an integration using multiple tests.

(continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMDB INT: Validate Operations</td>
<td>Validate all operations for an integration.</td>
<td>Paris</td>
</tr>
<tr>
<td>CMDB INT: Validate References</td>
<td>Validate CMDB integration references.</td>
<td>Paris</td>
</tr>
<tr>
<td>CMDB INT: Validate Related Entries</td>
<td>Validate all related classes against the data dictionary for related entries.</td>
<td>Paris</td>
</tr>
<tr>
<td>CMDB INT: Validate Relationships</td>
<td>Validate CMDB integration relationships.</td>
<td>Paris</td>
</tr>
</tbody>
</table>

To learn more about Integration Commons for CMDB, see Integration Commons for CMDB.

**Investment Funding**

The Investment Funding quick start tests require activating the Investment Funding - ATF Tests plugin (com.snc.investment_planning.atf).

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
</table>
| Validation of top-down funding and unfunding | • Validate that the correct amount is allocated from top investment to child investments while funding.  
• Validate that the correct amount is returned back to the top investment from the child investment while unfunding. | Orlando          |
<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation of bottom-up funding</td>
<td>• Validate that the correct amount is requested by the child investment from the parent investment.</td>
<td>Paris</td>
</tr>
<tr>
<td></td>
<td>• Validate that the correct amount is allocated back to the child investment from the parent investment.</td>
<td></td>
</tr>
</tbody>
</table>

To learn more about Investment Funding, see Investment Funding.

**Knowledge Management**

Knowledge Management quick start tests require activating the Knowledge Management Core plugin (com.glideapp.knowledge), the Knowledge Management Advanced Installer plugin (com.snc.knowledge_advanced.installer), the Knowledge Blocks plugin (com.snc.knowledge_blocks), and the Customer Service Management Demo Data plugin (com.snc.customerservice.demo).

**KM: Knowledge Management test suite**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>KM : Create KCS Template Article with Approval Publish workflow</td>
<td>Verify the creation of a KCS template article with approval publish workflow.</td>
<td>Madrid</td>
</tr>
<tr>
<td><img src="https://via.placeholder.com/15" alt="Note" /> <strong>Note:</strong> Requires the Knowledge Management Advanced Installer plugin.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM: Create MultiVersioned standard Article with approval publish workflow</td>
<td>Verify the creation of a multi-versioned standard article with approval publish workflow.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>KM: Article level subscription</td>
<td>Verify that users can subscribe to a knowledge article.</td>
<td>New York</td>
</tr>
<tr>
<td>KM: User criteria covering canRead and canContribute for KnowledgeBase and canRead at Article level</td>
<td>Verify the creation of canRead and canContribute user criteria for knowledge base and canRead user criteria for article.</td>
<td>Madrid</td>
</tr>
<tr>
<td>KM: Create a KCS article from a case</td>
<td>Verify the creation of a KCS article from a case.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>KM: Knowledge Base</td>
<td>Verify that users can subscribe to a Knowledge Base.</td>
<td>New York</td>
</tr>
<tr>
<td>Level Subscriptions</td>
<td><img src="image" alt="Note: Requires the Knowledge Management Advanced Installer plugin." /></td>
<td></td>
</tr>
<tr>
<td>KM: AQI</td>
<td>Verify the creation, assignment, and review of an AQI checklist.</td>
<td>New York</td>
</tr>
<tr>
<td><img src="image" alt="Note: Requires the Knowledge Management Advanced Installer plugin." /></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM: Requestor performs search, view the article, provide feedback (Helpful No with Feedback Task)</td>
<td>Verify the search request, review, provision of feedback as not helpful, and creation of a feedback task for an article on the Knowledge Management Service Portal.</td>
<td>New York</td>
</tr>
<tr>
<td><img src="image" alt="Note: Requires the Knowledge Management Advanced Installer plugin." /></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM: Create and edit an article from Feedback Task form</td>
<td>Confirm the availability of the <strong>Create Article</strong> and <strong>Edit Article</strong> buttons on a Knowledge Feedback Task form.</td>
<td>New York</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><strong>Note:</strong> Requires the Knowledge Management Advanced Installer plugin.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KM: Search for pinned articles and checking the highest click rank</td>
<td>Verify the search request and that the click rank value of a pinned article was added to the Knowledge Searches (ts_query_kb) table. <strong>Note:</strong> Requires the Knowledge Management Advanced Installer plugin.</td>
<td>New York</td>
</tr>
<tr>
<td>KM: Create an Ownership Group and check edit permissions and subscriptions</td>
<td>Verify the assignment of an article to an ownership group, that all members of the ownership group are subscribed to the article, and have permission to edit. <strong>Note:</strong> Requires the Knowledge Management Advanced Installer plugin.</td>
<td>New York</td>
</tr>
<tr>
<td>KM: Create, search for, and view knowledge articles with knowledge blocks</td>
<td>Verify the creation of a knowledge block and its addition to a knowledge article. Also, verify that the knowledge block content is searchable.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>KM: Validate the Knowledge - Approval Retire workflow</td>
<td>Verify the success of Knowledge - Approval Retire workflow by retiring a published knowledge article.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td>Note: Requires the Knowledge Blocks plugin.</td>
<td></td>
</tr>
<tr>
<td>KM: Validate feedback task assignment to the ownership group manager</td>
<td>Verify the assignment of a feedback task to the ownership group manager of a knowledge article.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td>Note: Requires the Knowledge Management Advanced Installer plugin.</td>
<td></td>
</tr>
<tr>
<td>KM: Validate that the article template selector includes knowledge</td>
<td>Verify that the article template selector includes all knowledge bases with contribute access and article templates applicable to each knowledge base.</td>
<td>Orlando</td>
</tr>
<tr>
<td>bases and article templates</td>
<td>Note: Requires Knowledge Management Advanced Installer.</td>
<td></td>
</tr>
</tbody>
</table>
## KM: Knowledge Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>KM: Validate the Knowledge - Approval Publish workflow</td>
<td>Verify the success of a Knowledge - Approval Publish workflow by recalling, rejecting, and then approving a knowledge article through the approval process.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires the Knowledge Management Advanced Installer plugin.</td>
<td></td>
</tr>
<tr>
<td>KM: Make an outdated version of an article the latest version</td>
<td>Verify that an outdated version of a knowledge article is made the latest version.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires the Knowledge Management Advanced Installer plugin.</td>
<td></td>
</tr>
<tr>
<td>KM: Validate that the mapped related articles appear in the Related articles widget</td>
<td>Verify that the related articles mapped to a knowledge article appear in the Related articles widget on the knowledge article view page.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires the Knowledge Management Advanced Installer plugin.</td>
<td></td>
</tr>
<tr>
<td>KM: Validate that the mapped related catalog</td>
<td>Verify that the related catalog items mapped</td>
<td>Orlando</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>items appear in the Related items widget</td>
<td>to a knowledge article appear in the Related items widget on the knowledge article view page.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>👌 Note: Requires the Knowledge Management Advanced Installer plugin.</td>
<td></td>
</tr>
<tr>
<td>KM: Validate that a versioned article is created, published, checked out, and retired</td>
<td>Verify the success of UI actions on a versioned knowledge article by creating, publishing, updating, and retiring the knowledge article in Agent Workspace.</td>
<td>Quebec</td>
</tr>
<tr>
<td></td>
<td>👌 Note: Requires the Knowledge Management Advanced Installer plugin.</td>
<td></td>
</tr>
<tr>
<td>KM: Verify the scheduled publish feature for knowledge articles</td>
<td>Verify the scheduled publish feature for knowledge articles by creating an article in approval publish workflow, putting it for scheduled for publish and observing the change in workflow state from the Scheduled for publish state to the Published state.</td>
<td>Quebec</td>
</tr>
</tbody>
</table>
### KM: Knowledge Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>KM: Validate the Knowledge - Approval Publish workflow in Agent Workspace</td>
<td>Verify the success of a Knowledge - Approval Publish workflow by recalling and then approving a knowledge article through the approval process in Agent Workspace.</td>
<td>Quebec</td>
</tr>
<tr>
<td><strong>Note:</strong> Requires the Knowledge Management Advanced Installer plugin.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** Requires the Knowledge Management Advanced Installer plugin.

To learn more about Knowledge Management, see Knowledge Management.

### Major Incident Management

Major Incident Management quick start tests require activating the Incident Management - Major Incident Management plugin (com.snc.incident.mim).

### MIM: Major Incident Management test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIM: Create a Major Incident</td>
<td>Test to verify the creation of major incident from the application navigation module.</td>
<td>Orlando</td>
</tr>
<tr>
<td>MIM: Create a Major Incident Candidate</td>
<td>Test to verify the Create a Major Incident Candidate module from navigation menu.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>MIM: Create a Major Incident from a Major Incident Candidate</td>
<td>Test to verify the creation of a major incident from a major incident candidate.</td>
<td>Orlando</td>
</tr>
<tr>
<td>MIM: Major Incident Candidate creation when it matches Major Incident Management trigger rule</td>
<td>Test to verify the creation of a major incident candidate when conditions to create a major incident match the major incident management trigger rule.</td>
<td>Orlando</td>
</tr>
<tr>
<td>MIM: Propose a major Incident (Assignment group empty)</td>
<td>Test to verify auto assignment of major incident candidate when an incident is proposed as a major incident and the Assignment group and the Assigned to fields are empty.</td>
<td>Orlando</td>
</tr>
<tr>
<td>MIM: Propose a major Incident (Assignment group is not empty)</td>
<td>Test to verify the auto assignment of a major incident when an incident is proposed as major incident and the Assignment Group and the Assigned to fields are not empty.</td>
<td>Orlando</td>
</tr>
<tr>
<td>MIM: Promote a candidate to major Incident (Assignment group is empty)</td>
<td>Test to verify auto assignment of major incident when a candidate is promoted to a major incident and the Assignment Group is empty.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>---------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>MIM: Promote a candidate to major Incident (Assignment group is not empty)</td>
<td>Test to verify auto assignment of major incident when a candidate is promoted to a major incident and the <strong>Assignment group</strong> is not empty.</td>
<td>Orlando</td>
</tr>
<tr>
<td>MIM: Reject a Major Incident Candidate</td>
<td>Test to verify the rejection of a major incident candidate.</td>
<td>Orlando</td>
</tr>
<tr>
<td>MIM: Demote a Major Incident</td>
<td>Test to verify whether a major incident gets demoted to an incident.</td>
<td>Orlando</td>
</tr>
<tr>
<td>MIM: Major Incident closure</td>
<td>Test to validate the major incident closure functionality.</td>
<td>Orlando</td>
</tr>
<tr>
<td>MIM: State sync up with ICP and ICT</td>
<td>Test is to verify the state sync up with Incident Communication Plan and Incident Communication Task.</td>
<td>Orlando</td>
</tr>
<tr>
<td>MIM: ICP attached to an Incident based on conditions and its state sync up with ICT</td>
<td>Test is to verify Incident Communication Plan attached to an incident based on defined conditions.</td>
<td>Orlando</td>
</tr>
<tr>
<td>MIM: Resolving MI and PIR</td>
<td>Test is to verify resolving a Major Incident and Post Incident Report.</td>
<td>Orlando</td>
</tr>
<tr>
<td>MIM: Major Incident workbench layout verification</td>
<td>Test is to verify Major Incident workbench layout verification.</td>
<td>Orlando</td>
</tr>
<tr>
<td>MIM: Verify communication task from MI workbench</td>
<td>Test is to verify communication task from MI workbench.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
To learn more about Major Incident Management, see [Major Incident Management](#).

### On-Call Scheduling

Activate the On-Call Scheduling plugin (com.snc.on_call_rotation) to run the On-Call Scheduling quick start tests.

**On-Call: On-Call scheduling ATF Suites**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-Call: Verify request time-off when PTO approval preferences is Not allowed</td>
<td>Verify whether you can request a time-off when PTO approval preferences is set to <strong>Not allowed</strong>.</td>
<td>Orlando</td>
</tr>
<tr>
<td>On-Call: Create overlapping shifts without selecting a template</td>
<td>Verify whether you can create overlapping shifts without selecting a template.</td>
<td>Orlando</td>
</tr>
<tr>
<td>On-Call: Create overlapping shifts when allow shift overlap is set to No</td>
<td>Verify whether you can create overlapping shifts when <strong>Allow overlap</strong> is set to <strong>No</strong>.</td>
<td>Orlando</td>
</tr>
<tr>
<td>On-Call: Request time-off when PTO approval preferences is with approval</td>
<td>Verify whether you can request a time-off when PTO approval preference is <strong>With approval</strong>.</td>
<td>Orlando</td>
</tr>
<tr>
<td>On-Call: Assign shift managers for maintaining on-call schedules.</td>
<td>Verify whether shift managers can maintain on-call schedules.</td>
<td>Orlando</td>
</tr>
<tr>
<td>On-Call: Search on-call schedules by user name</td>
<td>Verify whether you can search for on-call schedules by user name.</td>
<td>Orlando</td>
</tr>
<tr>
<td>On-Call: Search on-call schedules by group name or schedule name</td>
<td>Verify whether you can search for on-call schedules by group name or schedule.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>On-Call: Roster details for overlapping shifts with the escalation rule 'Escalate to incoming shift'</td>
<td>Verify the roster details for overlapping shifts when the escalation rule is set to <strong>Escalate to incoming shifts</strong>.</td>
<td>Orlando</td>
</tr>
<tr>
<td>On-Call: Calendar preview with timezone options</td>
<td>Verify whether the calendar preview is available with the timezone options.</td>
<td>Orlando</td>
</tr>
<tr>
<td>On-Call: Create overlapping shifts by selecting a template</td>
<td>Verify whether you can create overlapping shifts by selecting a template.</td>
<td>Orlando</td>
</tr>
<tr>
<td>On-Call: Show active shifts</td>
<td></td>
<td>Orlando</td>
</tr>
<tr>
<td>On-Call: Make shift draft/publish form shift form</td>
<td>Verify whether you can publish/unpublish a shift and alternate between making the rota <strong>Draft</strong> and <strong>Publish</strong> state from the form.</td>
<td>Orlando</td>
</tr>
<tr>
<td>On-Call: Roster Details for overlapping shifts with the escalation rule 'Escalate to outgoing shift'</td>
<td>Verify the roster details for overlapping shifts when the escalation rule is set to <strong>Escalate to outgoing shifts</strong>.</td>
<td>Orlando</td>
</tr>
<tr>
<td>On-Call: Roster Details for overlapping shifts with the escalation rule 'Escalate to all shifts'</td>
<td>Verify the roster details for overlapping shifts when the escalation rule is set to <strong>Escalate to all shifts</strong>.</td>
<td>Orlando</td>
</tr>
<tr>
<td>On-call: Verify On-Call workbench</td>
<td>Test to verify the On-Call workbench layout.</td>
<td>Orlando</td>
</tr>
<tr>
<td>On-call: Hide or show shifts</td>
<td>Verify whether you can hide or show shifts from On-Call calendar</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
To learn more about On-Call Scheduling, see On-Call Scheduling.

**Metric Intelligence**

Metric Intelligence quick start tests require activating the Metric Intelligence plugin (com.snc.sa.metric).

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>OI: Health Metrics Collection</td>
<td>An individual test that verifies the following:</td>
<td>Madrid</td>
</tr>
<tr>
<td></td>
<td>• There is at least one Metric Intelligence Metrics extension that is running.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• There are entries for the ‘Health Metrics’ in the Metric to CI table.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• For each of the Metric Intelligence Metrics extensions that are currently running, that</td>
<td></td>
</tr>
<tr>
<td></td>
<td>there is data stored in MetricBase.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If the test initially fails, wait until the Metric Intelligence Metrics extension runs for at</td>
<td></td>
</tr>
<tr>
<td></td>
<td>least 10-15 minutes. Then try again.</td>
<td></td>
</tr>
</tbody>
</table>

To learn more about Metric Intelligence, see Metric Intelligence.

**Predictive Intelligence**

Predictive Intelligence quick start tests require activating the Predictive Intelligence [com.glide.platform_ml] plugin. In order to execute critical upgrade tests on existing machine learning solutions, you need to create a basic authorization profile named ml_atf in the Basic Auth Configurations table (sys_auth_profile_basic.list). To run the tests successfully, the user attached to the ml_atf authorization profile must have the ml_admin role.
### Predictive Intelligence: Classification and Similarity Solution Prediction test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>PI: Presence of ML model artifacts persisted in glide</td>
<td>Verify all the trained ML model artifacts are persisted in glide (sys_attachments table) after model training/instance cloning so that ML prediction calls are successful.</td>
<td>New York</td>
</tr>
<tr>
<td>PI: Valid setup of ML user (sharedservice.worker) in glide</td>
<td>Validate if the ML user in glide (sharedservice.worker) is active and not logged out so that model training is successful.</td>
<td>New York</td>
</tr>
<tr>
<td>PI: Glide upgrade test for Classification solution</td>
<td>Validate that the classification model prediction on existing active models is producing the same class membership and confidence value results after a glide upgrade.</td>
<td>New York</td>
</tr>
<tr>
<td>PI: Glide upgrade test for Similarity solution</td>
<td>Validate that the similarity model prediction API calls on active models are successful after a glide upgrade.</td>
<td>New York</td>
</tr>
</tbody>
</table>

### Problem Management

Problem Management quick start tests require activating the Problem Management Best Practice — Madrid plugin (com.snc.best_practice.problem.madrid) and the Problem Management — ATF Tests plugin (com.snc.problem.atf). For all state related test, the Problem Management State Model (com.snc.best_practice.problem.madrid.state_model) plugin needs to be active.
<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRB MGMT: Cancel a Problem when the state of the Problem is Assess</td>
<td>Verify that when a Problem in the <strong>Assess</strong> state is canceled, the state of the Problem changes to <strong>Closed</strong> with <strong>Resolution code</strong> as <strong>Canceled</strong>.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Cancel a Problem when the state of the Problem is Root Cause Analysis</td>
<td>Verify that when a Problem is in the <strong>Root Cause Analysis</strong> state and is canceled, the state of the Problem changes to <strong>Closed</strong> with <strong>Resolution code</strong> as <strong>Canceled</strong>.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Mark a Problem as Duplicate when the state of the Problem is Assess</td>
<td>Verify that when a Problem is in the <strong>Assess</strong> state and is marked as duplicate, the state of the Problem changes to <strong>Closed</strong> with <strong>Resolution code</strong> as <strong>Duplicate</strong>.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Mark a Problem as Duplicate when the state of the Problem is Root Cause analysis</td>
<td>Verify that when a Problem is in the <strong>Root Cause Analysis</strong> state and is marked as duplicate, the state of the Problem changes to <strong>Closed</strong> with <strong>Resolution code</strong> as <strong>Duplicate</strong>.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Accept Risk of Problem (problem.acceptrisk.move_to_closed:false,state:Fix in Progress)</td>
<td>Verify that when a Problem state is <strong>Fix in Progress</strong> and the risk is accepted, then the Problem state changes to <strong>Resolved</strong> with <strong>Resolution code</strong> as <strong>Risk Accepted</strong>.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>
### PRB MGMT: Problem Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="https://via.placeholder.com/50" alt="Note:" /></td>
<td>The test is valid when Problem property <strong>Accept Risk</strong> moves the Problem to Closed state instead of Resolved state <em>(problem.acceptrisk.move_to_closed)</em> is false.</td>
<td></td>
</tr>
<tr>
<td>PRB MGMT: Accept Risk of Problem (problem.acceptrisk.move_to_closed:true,state:Fix in Progress)</td>
<td>Verify that when a Problem state is <strong>Fix in Progress</strong> and the risk is accepted, then the Problem state changes to <strong>Closed</strong> with Resolution code as <strong>Risk Accepted</strong>.</td>
<td>Madrid</td>
</tr>
<tr>
<td><img src="https://via.placeholder.com/50" alt="Note:" /></td>
<td>The test is valid when Problem property <strong>Accept Risk</strong> moves the Problem to Closed state instead of Resolved state <em>(problem.acceptrisk.move_to_closed)</em> is true.</td>
<td></td>
</tr>
<tr>
<td>PRB MGMT: Accept Risk of Problem (problem.acceptrisk.move_to_closed:false,state:Root Cause Analysis)</td>
<td>Verify that when a Problem state is <strong>Root Cause Analysis</strong> and the risk is accepted, then the Problem state changes to <strong>Resolved</strong> with Resolution code as <strong>Risk Accepted</strong>.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>----------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><strong>PRB MGMT: Problem Management test suite (continued)</strong></td>
<td><strong>Note:</strong> The test is valid when Problem property Accept Risk moves the Problem to Closed state instead of Resolved state (problem.acceptrisk.move_to_closed) is false.</td>
<td></td>
</tr>
<tr>
<td>PRB MGMT: Accept Risk of Problem</td>
<td>Verify that when a Problem state is Root Cause Analysis and the risk is accepted, then the Problem state changes to Closed with Resolution code as Risk Accepted.</td>
<td>Madrid</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>The test is valid when Problem property Accept Risk moves the Problem to Closed state instead of Resolved state (problem.acceptrisk.move_to_closed) is true.</td>
<td></td>
</tr>
<tr>
<td>PRB MGMT: Reanalyze Problem which is closed-Risk Accepted from state Root Cause Analysis</td>
<td>Verify that when a Problem is reanalyzed after it is Closed with the Resolution code as Risk Accepted, Problem state changes to Root Cause Analysis.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Create Emergency Change from Problem</td>
<td>Verify the creation of Emergency Change from a Problem.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRB MGMT: Create Normal Change from Problem</td>
<td>Verify the creation of Normal Change from a Problem.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Problem State Management</td>
<td>Verify problem state management.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Reanalyze a Problem from Complete</td>
<td>Verify that when a Problem is reanalyzed after it is Closed with the Resolution code as Fix Applied, Problem state changes to Root Cause Analysis.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Reanalyze a Problem which is canceled from state Assess</td>
<td>Verify that when a Problem is reanalyzed after it is Closed with the Resolution code as Canceled, Problem state changes to Root Cause Analysis.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Reanalyze a Problem which is canceled from state Root Cause Analysis</td>
<td>Verify that when a Problem is reanalyzed after it is Closed with the Resolution code as Canceled, Problem state changes to Root Cause Analysis.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Reanalyze Problem which is closed- Risk Accepted from state Fix in progress</td>
<td>Verify that when a Problem is reanalyzed after it is Closed with the Resolution code as Risk Accepted, Problem state changes to Root Cause Analysis.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Create a Known Error article from Problem</td>
<td>Verify creation of Known Error article from a Problem.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>
### PRB MGMT: Problem Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRB MGMT: Risk Accept reason on Incident</td>
<td>Verify the <strong>Risk Accepted reason</strong> is copied to the Incidents which are awaiting resolution of a Problem.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Communicate Fix</td>
<td>Verify the communicate fix functionality.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Communicate Workaround</td>
<td>Verify the communicate workaround functionality.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Fix information on Incident</td>
<td>Verify that when a Problem is resolved, the state of the Incidents that are awaiting resolution of the Problem changes to <strong>Resolved</strong>. The fix notes of the Problem are copied to the Incidents.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PRB MGMT: Problem task (Type:General) state management</td>
<td>Verify Problem task state management of a general type Problem.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>

To learn more about Problem Management, see [Problem Management](#).

### Project Portfolio Management

Project Portfolio Management quick start tests require activating the PPM Standard - ATF Tests plugin (com.snc.financial_planning_pmo.atf).

### PMO: Financial Tests for verifying cost rollups and demand to project conversion test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMO: Verify cost plan roll up to project/demand and program</td>
<td>Validate the total planned cost rollup from</td>
<td>Madrid</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
PMO: Financial Tests for verifying cost rollups and demand to project conversion test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMO: Verify cost plan roll up to project/demand, program and portfolio</td>
<td>Validate the total planned cost rollup from project and demand to portfolio.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PMO: Verify financials of Project created from Demand - Simple Financials</td>
<td>Validate the financial tab fields of a project created from a demand.</td>
<td>Madrid</td>
</tr>
<tr>
<td>PMO: Verify financials of Project created from Demand - With budget, cost plans, benefit plans</td>
<td>Validate the budget, cost plan, and benefit plan of a project created from a demand.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>

PMO: Project Management tests for validating basic life cycle and project rollups test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validate PPM Cycle from Ideation to demand to project closure</td>
<td>Validate the flow of creating an idea, converting the idea to a demand, and then converting the demand to a project.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Note: This test fails if the PPM Standard Multicurrency (com.snc.ppm_multicurrency) plugin is active.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Validation of State and Date Rollup for Automatic Project</td>
<td>Validate the date and state rollup from tasks for a project of type Automatic.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Validation of State and Date rollup for Manual Project</td>
<td>Validate the date and state rollup from tasks for a project of type Manual.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>
### PMO: Project Management tests for validating basic life cycle and project rollups test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation of project percent complete when all tasks are Closed Incomplete</td>
<td>Validate the project percent complete when all the tasks are closed as Closed Incomplete.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Validate Project is not 100 percent complete if it has atleast one task as closed incomplete</td>
<td>Validate the project percent complete is not 100% when at least one of the tasks is closed as Closed Incomplete or Closed Skipped with task % complete less than 100%.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
| Validate waterfall project does not show Agile Planning Board       | Validate that the waterfall projects cannot use the following:  
  • Add stories or epics  
  • Add agile phase  
  • Access Agile board from the project                                                                                                                                   | Orlando         |
<p>| Validate dates are rolled up from existing projects to program       | Validate that the start and end dates of the project are rolled up to the program to which the project belongs.                                                                                              | Paris           |
| Validate dates are rolled up from new projects to program            | Validate that the start and end dates of a new project are rolled up to the program to which the project belongs.                                                                                             | Paris           |
| Add projects and demands with risks or issues to program            | Validate that the projects and demands with risks or issue records associated with them are added to the program.                                                                                             | Paris           |
| Add projects and demands with benefit plans to program               | Validate that the projects and demands with benefit plans associated with them are added to the program.                                                                                                 | Paris           |
| Add projects and demands with cost plans to program                 | Validate that the projects and demands with cost plans associated with them are added to the program.                                                                                                    | Paris           |
| Validate program dates on addition of existing demands              | Validate that the start and end date of the program are adjusted on addition of an existing demand.                                                                                                        | Paris           |</p>
<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validate program dates on addition of new demands</td>
<td>Validate that the start and end date of the program are adjusted on addition of a new demand.</td>
<td>Paris</td>
</tr>
<tr>
<td>Validate program state rollup</td>
<td>Validate that the program state is rolled up from state of all the projects in the program.</td>
<td>Paris</td>
</tr>
<tr>
<td>Verify Demand task due date column field value empty</td>
<td>Validate that the <strong>Due date</strong> field is empty for a new demand task.</td>
<td>Rome</td>
</tr>
<tr>
<td>Verify baseline is created on Project &amp; Demand when demand is converted to project</td>
<td>Validate that a project and demand baseline is created when a demand is converted to a project.</td>
<td>Rome</td>
</tr>
<tr>
<td>Verify timecard financial appears on Demand when time card submitted against demand task</td>
<td>Validate that the actual cost of the demand is updated when a time card is processed for a demand task.</td>
<td>Rome</td>
</tr>
<tr>
<td>Child Test Suite: Validation of Move Project Action</td>
<td></td>
<td>Orlando</td>
</tr>
<tr>
<td>Validate Planned start date of a project can be shifted by using the Move project action</td>
<td>Validate that the Planned start date of a project in Planning or Open state can be updated to a later or earlier date than the current planned start date using the Move project related link.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Validate Move Project is disabled when Project is selected for execution</td>
<td>Validate that the Move Project option is not available if the Project is in Execution phase.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Validate Move Project functionality with sprint dates</td>
<td>Validate that the sprint start and end dates are cleared when the project is moved using the Move Project related link.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
PMO: Project Management tests for validating basic life cycle and project rollups test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>populated for an agile phase</td>
<td>This test is available only when Agile Development 2.0 plugin (com.snc.sdlc.agile.2.0) is activated.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Validate Move Project functionality with external dependencies and related entities</td>
<td>Validate that the external dependencies and related entities are also shifted and redrawn when the project is moved using the Move Project related link.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Validate Move Project functionality with different project states</td>
<td>Validate that the Move Project does not work when the project is in Work In Progress or Closed Complete state.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Child Test Suite: Verify RIDAC flow of a Project</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Verify RIDAC flow for Risk</td>
<td>Validate the RIDAC flow for a risk associated with a project.</td>
<td>Quebec</td>
</tr>
<tr>
<td>Verify RIDAC flow for Issue</td>
<td>Validate the RIDAC flow for an issue associated with a project.</td>
<td>Quebec</td>
</tr>
<tr>
<td>Verify RIDAC flow for Decision</td>
<td>Validate the RIDAC flow for a decision associated with a project.</td>
<td>Quebec</td>
</tr>
<tr>
<td>Verify RIDAC flow for Action</td>
<td>Validate the RIDAC flow for an action associated with a project.</td>
<td>Quebec</td>
</tr>
<tr>
<td>Verify changes in fields of risk form</td>
<td>Validate that any change is the Risk form fields such as Risk rank, Risk value, and Probability is updated successfully upon submission.</td>
<td>Quebec</td>
</tr>
<tr>
<td>Child Test Suite: Verify RIDAC flow of a Demand</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Verify RIDAC flow for Risk</td>
<td>Validate the RIDAC flow for a risk associated with a demand.</td>
<td>Quebec</td>
</tr>
<tr>
<td>Verify RIDAC flow for Issue</td>
<td>Validate the RIDAC flow for an issue associated with a demand.</td>
<td>Quebec</td>
</tr>
</tbody>
</table>
## PMO: Project Management tests for validating basic life cycle and project rollups test suite
(continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verify RIDAC flow for Decision</td>
<td>Validate the RIDAC flow for a decision associated with a demand.</td>
<td>Quebec</td>
</tr>
<tr>
<td>Verify RIDAC flow for Action</td>
<td>Validate the RIDAC flow for an action associated with a demand.</td>
<td>Quebec</td>
</tr>
<tr>
<td>Verify changes in fields of risk form</td>
<td>Validate that any change is the Risk form fields such as Risk rank, Risk value, and Probability is updated successfully upon submission.</td>
<td>Quebec</td>
</tr>
</tbody>
</table>

## PMO: Resource Management tests for verifying the resource plan flows test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verify user resource plan flow from Planned to Canceled state</td>
<td>Validate that the resource plan of a project can be moved to canceled state from planned state.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Verify group resource plan flow from Planned to Complete state</td>
<td>Validate that the resource plan of a project can be moved to complete state from planned state.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Verify role resource plan flow from Planned to Allocated state</td>
<td>Validate that the resource plan of a project can be moved to allocated state from planned state.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Verify aggregated cost of all resource plans roll up to the corresponding</td>
<td>Validate that the aggregated cost of all resource plans on a project or demand roll up to the Planned Cost and Allocated Cost fields and the Resource Cost section of respective projects and demands.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
### PMO: Resource Management tests for verifying the resource plan flows test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>project or demand fields</td>
<td><a href="#">Description</a></td>
<td><a href="#">Orlando</a></td>
</tr>
<tr>
<td>Verify Copy Resource plan option</td>
<td>Validate that the Copy Resource plan option creates an exact copy of the source resource plan in the Planning state</td>
<td>Orlando</td>
</tr>
<tr>
<td>Verify resource plan aggregate roll up from project/demand to program</td>
<td>Validate that the aggregated cost of all resource plans on a project or demand roll up to the total planned cost of the associated program.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
| Verify records on completion of a resource plan | Validate the changes in a resource plan on completion:  
- The state of the resource plan is updated to Completed.  
- If the completion date is earlier than the resource plan end date, the end date of the resource plan is updated with the completion date. If the completion date entered is later than the resource plan end date, the resource plan end date is retained.  
- All the requested and allocation records for the resource plan for the period after the completion date are deleted. If there are any actual hours logged against an allocation, that allocation is not deleted. For those allocation records, the allocated hours become zero and the actual hours are retained.  
- The available and allocated hours for resources are updated in the aggregate tables. | Orlando |
<p>| Verify records on completion of a resource plan with Planned Duration as allocation type: | Validate the following on completion of a resource plan with Planned Duration as allocation type: | Orlando |</p>
<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
</table>
| Planned Duration as allocation type       | • The state of the resource plan is updated to Completed.  
• Allocations are not deleted.  
• End date of the allocation is updated to the completion date.                                                                                           |                 |
| Verify the RP replan Capability           | Validate that when a cancelled resource plan is replanned, the state of the resource plan changes to Planning.                                                                                             | Orlando         |
| Verify whether change in resource plan is reflected in corresponding cost plan | Validate that when a resource plan is updated, the corresponding cost plan is updated accordingly. For example, if the total planned cost is 500 USD, and the planned hours is 10, and you change the planned hours to 20, the total planned cost is updated to 1000 USD. | Paris           |
| Resource-Test the default population of resource plan start & end date | Validate the following on creating a user or group resource plan from the related list of a demand:  
• If a demand is created without a start date and end date, the user or group resource plan has task as demand and no start and end date.  
• If a demand is created with a start date and end date, the user resource plan has task as demand and the start date and end date as added for the demand. | Paris           |
| Validate that actual hours in operational resource plan and time card are equal | Validate that when a time card category is mapped with an operational work type, on submitting the time card for the operational resource plan associated with that work type, the actual hours in the resource plan and the time card are equal. | Quebec          |
| Verify Resource plan auto population for Operational plans | Validate that the operational resource plans associated with a time card are automatically retrieved on the time card when time is logged.                                                                       | Rome            |
PMO: Resource Management tests for verifying the resource plan flows test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verify resource plan auto population for non-operational resource plans</td>
<td>Validate that the resource plans associated with a project, project task, or demand for a time card are automatically retrieved on the time card when time is logged.</td>
<td>Rome</td>
</tr>
</tbody>
</table>

To learn more about Project Portfolio Management, see Project Portfolio Management.

Project Currency test suite

Project currency quick start tests require activating the PPM Standard Multicurrency – ATF Tests plugin (com.snc.ppm_multicurrency.atf).

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verify cost in project currency on cost plan</td>
<td>Validate the calculation of cost line breakdown with budget reference rate and verify roll up to cost plan and also for the project in project currency.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Verify benefit in project currency on benefit plan</td>
<td>Validate the calculation of benefit line breakdown with budget reference rate and verify roll up to benefit plan and also for the project in project currency.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

To learn more about PPM Standard, see PPM Standard.

Reporting

The Reporting quick start test Automated Test Framework - Reporting plugin (com.glide.automated_testing_impl.report) is active by default or instance reboot.
Note: Reporting quick start tests do not test report access from dashboards. To test dashboards, see Quick start tests for Dashboards.

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report Visibility</td>
<td>Confirm whether reports are still visible to users whom they are shared with.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>

To learn more about Reporting, see

**Software Asset Management**
Software Asset Management quick start tests require activating the Software Asset Management Professional plugin (com.snc.samp). Some quick start tests require activating the following additional plugins.

- Software Asset Management - Spend Detection (com.sn_sam_spend)
- Software Asset Management Professional for Microsoft (com.snc.samp.microsoft)
- Software Asset Management Professional for SAP (com.sn_samp_sap)
- Software Asset Management - SaaS License Management Integrations (com.sn_sam_saas_int)

**Software Asset Management test suite**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
</table>
| SAM - Oracle PaaS BYOL| • Validates the addition of the new Serverless Hardware [cmdb_ci_serverless_hardware] table, which stores information about PaaS devices.  
                        | • Validates the license compliance of Oracle Database servers in Amazon Web Services (AWS) PaaS environments. | Rome            |
Software Asset Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note:</td>
<td>Requires the Software Asset Management Professional for Oracle (com.snc.samp.oracle) plugin and the CMDB CI Class Models store application.</td>
<td></td>
</tr>
<tr>
<td>SAM - Product Lifecycle on Software Model</td>
<td>Validates that the Software Lifecycle tab on the Software Model form is showing records.</td>
<td>Quebec</td>
</tr>
<tr>
<td>SAM - BYOL</td>
<td>• Validates the purchase date on the Software Entitlement form</td>
<td>Quebec</td>
</tr>
<tr>
<td></td>
<td>• Validates the addition of newly added column legacy_license on the License Metric Results [samp_license_metric_result] and License Position Report [samp_license_position_report] tables</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Validates the addition of newly added columns, cloud_license_type and cloud_license_type_source in the Software Installations [cmdb_sam_sw_install] table</td>
<td></td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><strong>Note:</strong> Requires the Software Asset Management Professional for Microsoft (com.snc.samp.microsoft) plugin.</td>
<td><strong>SAM - Validate CIS Suites</strong> Validates reconciliation of Core Infrastructure Server (CIS) suites along with downgrade rights.</td>
<td><strong>Paris</strong></td>
</tr>
<tr>
<td><strong>Note:</strong> Requires demo data and the Software Asset Management Professional for Microsoft (com.snc.samp.microsoft) plugin.</td>
<td><strong>SAM - Validate upgrade/downgrade during Reconciliation for Microsoft publisher</strong> Validates upgrade and downgrade rights during reconciliation for Microsoft products.</td>
<td><strong>Paris</strong></td>
</tr>
</tbody>
</table>
## Software Asset Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>† <strong>Note:</strong> Requires demo data and the Software Asset Management Professional for Microsoft (com.snc.samp.microsoft) plugin.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAM - Generate demand to consolidate SaaS applications</td>
<td>Validates generation and submission of a demand on SaaS applications.</td>
<td>Paris</td>
</tr>
<tr>
<td>† <strong>Note:</strong> Requires the Software Asset Management - Spend Detection (com.sn_sam_spend) plugin.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAM - Software Model and Software Entitlement checks for SAP Engines</td>
<td>Verifies that the Software Model and Software Entitlement forms change when the product is an SAP engine.</td>
<td>Paris</td>
</tr>
<tr>
<td>† <strong>Note:</strong> Requires the Software Asset Management Professional for SAP (com.sn_samp_sap) plugin.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAM - Downgrade Rights on Software Model</td>
<td>Validates that the downgrade rights pushed from the content service are correctly populated on the Downgrade Rights</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>related list on the software model form.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAM - Multi-core pack validation on Software Entitlement</td>
<td>Validates the functionality of new fields for a multi-core pack on software entitlements.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Note: Requires the Software Asset Management Professional for Microsoft (com.snc.samp.microsoft) plugin.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAM - Downgrade Rights on Software Entitlement</td>
<td>Validates that the downgrade rights pushed from the content service are correctly populated on the Downgrade Rights related list on the software entitlement form.</td>
<td>Orlando</td>
</tr>
<tr>
<td>SAM - Validate Reconciliation for Custom Product</td>
<td>Validates reconciliation for a custom product.</td>
<td>Orlando</td>
</tr>
<tr>
<td>SAM - Software Spend Transaction</td>
<td>Validates the creation of a Software Spend Transaction.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Note: Requires the Software Asset Management - Spend Detection (com.sn_sam_spend) plugin.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>---------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>SAM - Pin and Unpin Publishers on License Workbench</td>
<td>Validates the pin and unpin of publishers on License Workbench.</td>
<td>Orlando</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires demo data.</td>
<td></td>
</tr>
<tr>
<td>SAM - Software Model and Software Entitlement</td>
<td>Tests that a user can create a software model and software entitlement and validates those records.</td>
<td>New York</td>
</tr>
<tr>
<td>SAM - Software Installation and Discovery Model</td>
<td>Tests that a user can create a software installation and discovery model and validates those records.</td>
<td>New York</td>
</tr>
<tr>
<td>SAM - Software Entitlement Creation Using Custom PPN</td>
<td>Creates a custom software product, a custom DMAP for the custom product, a custom Part Number for the custom DMAP, a software entitlement using the custom Part Number, and verifies that a software model is automatically created.</td>
<td>New York</td>
</tr>
<tr>
<td>SAM - Software Model Checks for SAP Named Users</td>
<td>Tests that the software model form changes when the publisher is SAP and the product is Named Users.</td>
<td>New York</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><strong>SAM - Software Model Checks for SaaS</strong></td>
<td>Tests that the Software Model form changes when a SaaS product is selected.</td>
<td>New York</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires the Software Asset Management Professional for SAP (com.sn_samp_sap) plugin.</td>
<td></td>
</tr>
<tr>
<td><strong>SAM - Validate Fields on SaaS Software Products</strong></td>
<td>Tests that the Subscription software and Ignore installs fields are present on the Software Product form.</td>
<td>New York</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Requires the Software Asset Management - SaaS License Management Integrations (com.sn_sam_saas_int) plugin.</td>
<td></td>
</tr>
</tbody>
</table>

To learn more about Software Asset Management, see Software Asset Management.
## Security Incident Response

Security Incident Response quick start tests require activating Security Incident Response plugin (com.snc.security_incident) and loading the demo data.

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIR: Create Security Incident</td>
<td>Determine whether a user can successfully create a security incident from the security incident form.</td>
<td>Madrid</td>
</tr>
<tr>
<td>SIR: Create Security Incident via Security Incident Catalog</td>
<td>Determine whether a user can successfully create a security incident from the catalog.</td>
<td>Madrid</td>
</tr>
<tr>
<td>SIR: Security Incident life cycle</td>
<td>Validate the response tasks of the Policy Violation workflow.</td>
<td>Madrid</td>
</tr>
<tr>
<td>SIR: Threat Lookup</td>
<td>Validates the Threat Lookup capability.</td>
<td>Orlando</td>
</tr>
<tr>
<td>SIR: PIR Assessments OOTB configuration</td>
<td>Use this test to validate PIR assessments and base system configurations.</td>
<td>Rome</td>
</tr>
<tr>
<td>SIR: PIR Assessments conditional configuration</td>
<td>Verify that security incidents matching the mandatory conditional rule are not closed without completing the post incident assessment. Verify that the security incidents matching the optional conditional rule can be closed without completing the post incident assessment. Verify that assessments are not generated for</td>
<td>Rome</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td>the security incidents that do not match any rule.</td>
<td></td>
</tr>
<tr>
<td>SIR: PIR run time verification</td>
<td>Verify that PIR reports are configured and attached to the security incidents as per the new design.</td>
<td>Rome</td>
</tr>
<tr>
<td>SIR: PIR design time setup verification</td>
<td>Verify that the security incident is mapped with the report template based on the administrator configuration.</td>
<td>Rome</td>
</tr>
</tbody>
</table>

To learn more about Security Incident Response, see Security Incident Response.

**Service Level Management**

Service Level Management quick start tests require activating the Service Level Management - ATF Tests plugin (com.snc.service_level_management.atf).

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLM: Service Level Management Task SLA completed workflow</td>
<td>Tests that a user can create a task SLA and complete the workflow.</td>
<td>Orlando</td>
</tr>
<tr>
<td>SLM: Service Level Management Task SLA cancelled workflow</td>
<td>Tests that a user can create a task SLA and cancel the workflow.</td>
<td>Orlando</td>
</tr>
<tr>
<td>SLM: Service Level Management Task SLA timer REST API</td>
<td>Tests that the SLA Timer API response matches data expected by SLA Timer Seismic component.</td>
<td>Paris</td>
</tr>
</tbody>
</table>

To learn more about Service Level Management, see Service Level Management.

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Service Mapping

Service Mapping quick start tests require activating the Service Mapping (com.snc.service-mapping) plugin and loading the demo data.

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>SM OOTB: Tests SM application visibility by roles</td>
<td>Validate the visibility of Service Mapping applications and modules for different roles. For example, the test verifies that a user logged in with the sm_user role, cannot access the Administration module under Service Mapping.</td>
<td>Madrid</td>
</tr>
<tr>
<td>SM OOTB: Service map verification</td>
<td>Use this test template to create custom tests for verifying that the topology of the most significant services is unchanged. Configure values under Test Run Data Sets, to identify the name of the service to test and the nodes that you expect to find in this service. For the node name values, enter the node attribute exactly as it appears on the map, for example, &quot;Apache server.&quot; If the attribute name for a node is truncated or shows that a node is a CI cluster, configure this node name value to reflect the way it appears on the map, for example,</td>
<td>Orlando, updated Paris</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>---------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>“11x ExchangeF” for a group of 11 Exchange FrontEnd servers.</td>
<td>Note: You cannot use this test to verify the service content beyond the top, unexpanded level.</td>
<td></td>
</tr>
<tr>
<td>SM OOTB: Check UI accessibility after an upgrade</td>
<td>Run this test to check that the Service Mapping UI is fully functional after an upgrade.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

To learn more about Service Mapping, see Service Mapping.

**Service Portfolio Management Premium**

Service Portfolio Management Premium quick start tests require activating the Service Portfolio Management Premium plugin (com.snc.spm).

**Service Portfolio Management Premium - ATF Tests test suite**

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Portfolio Editor: Not Portfolio Owner, Read Only Taxonomy Access</td>
<td>Ensure a portfolio editor can only read associated non-owned portfolio taxonomies.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Portfolio Editor: Cannot create new Portfolios</td>
<td>Ensure a portfolio editor cannot create new portfolios.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Portfolio Editor: Valid Portfolio Owner Taxonomy Access</td>
<td>Ensure a portfolio owner has access to taxonomies within owned portfolios.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Service Editor: Access</td>
<td>Ensure a service editor can only edit owned services and offerings.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Portfolio Editor: Can Modify Owned Portfolios</td>
<td>Ensure a portfolio editor can modify and update owned portfolios.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Create a Portfolio</td>
<td>Ensure a portfolio admin can create a new portfolio.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Portfolio Editor: Cannot Modify Non-Owned Portfolios</td>
<td>Ensure a portfolio editor cannot modify non-owned portfolios.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Service Workflow</td>
<td>Verify that a service cannot move forward to Catalog phase without a service portfolio, taxonomy node, and service offering attached to it. Verify a service cannot move backward from the Catalog phase to the Pipeline phase.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Service Portfolio (Normal)</td>
<td>Create a service portfolio, taxonomy layer, and taxonomy nodes.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Create a Portfolio + 3 Taxonomy Layers</td>
<td>Create a service portfolio with three taxonomy layers.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Service Portfolio (Exception) - Taxonomy Layer Definition Set 01</td>
<td>Create a service portfolio, taxonomy layer, and taxonomy nodes.</td>
<td>Orlando</td>
</tr>
<tr>
<td>SPM: Create a Portfolio</td>
<td>Create a service portfolio.</td>
<td>Paris</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>SPM: Create a Portfolio + 3 Taxonomy Layers</td>
<td>Create a service portfolio, with three taxonomy layers.</td>
<td>Paris</td>
</tr>
<tr>
<td>SPM: Service Portfolio (Normal)</td>
<td>Create a service portfolio, taxonomy layer, and taxonomy nodes.</td>
<td>Paris</td>
</tr>
<tr>
<td>SPM: Service Portfolio (Exception) - Taxonomy Layer Definition Set 01</td>
<td>Create a service portfolio, taxonomy layer, and taxonomy nodes.</td>
<td>Paris</td>
</tr>
<tr>
<td>SPM: Portfolio Editor: Can Modify Owned Portfolios</td>
<td>Ensure a portfolio editor can modify portfolios that they own.</td>
<td>Paris</td>
</tr>
<tr>
<td>SPM: Portfolio Editor: Cannot Modify Non-Owned Portfolios</td>
<td>Ensure a portfolio editor cannot modify portfolios that they do not own.</td>
<td>Paris</td>
</tr>
<tr>
<td>SPM: Portfolio Editor: Cannot create new Portfolios</td>
<td>Ensure a portfolio editor cannot create new Portfolios.</td>
<td>Paris</td>
</tr>
<tr>
<td>SPM: Portfolio Editor: Valid Portfolio Owner Taxonomy Access</td>
<td>Ensure portfolio owners have access to taxonomies that they own.</td>
<td>Paris</td>
</tr>
<tr>
<td>SPM: Portfolio Editor: Not Portfolio Owner, Read Only Taxonomy Access</td>
<td>Ensure a portfolio editor can only read the taxonomies of portfolios they do not own.</td>
<td>Paris</td>
</tr>
<tr>
<td>SPM: Service Editor: Access</td>
<td>Ensure a service editor can only edit services and offerings that they own or are a delegate of.</td>
<td>Paris</td>
</tr>
</tbody>
</table>
Service Portfolio Management Premium - ATF Tests test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPM: Service Viewer: Access</td>
<td>Ensure a service viewer has access to view services.</td>
<td>Paris</td>
</tr>
<tr>
<td>SPM: Service Workflow</td>
<td>Verify a service cannot move forward to Catalog phase without a service portfolio, taxonomy node, and service offering attached to it. Verify a service cannot move backward from Catalog to Pipeline phase.</td>
<td>Paris</td>
</tr>
</tbody>
</table>

To learn more about Service Portfolio Management Premium, see Service Portfolio Management Premium.

Skills Management

Skills Management quick start tests require activating the Skills Management plugin (com.snc.skills_management).

Skills Management test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Skills MGMT: User skill level inheritance when user is part of multiple groups</td>
<td>Verify that the user is assigned the highest skill level when the user belongs to multiple groups that have been assigned the same skills with different skill levels.</td>
<td>New York</td>
</tr>
<tr>
<td>Skills MGMT: Add skills to lowest level category</td>
<td>Verify that skills can be added to the lowest level category.</td>
<td>New York</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Skills MGMT: Create a child category under a parent category</td>
<td>Verify that a lower-level category can be created when the flag for Add skills is unchecked in the parent category.</td>
<td>New York</td>
</tr>
<tr>
<td>Skills MGMT: Create a skill category</td>
<td>Verify that a skill category can be created on the skill category form.</td>
<td>New York</td>
</tr>
<tr>
<td>Skills MGMT: Create skill level type and skill levels</td>
<td>Define the skill level type and different skill levels for the type.</td>
<td>New York</td>
</tr>
<tr>
<td>Skill MGMT: Skill level inheritance from group to user</td>
<td>Verify that skill levels can be inherited from a group to the users of the group and that the Inherited and Skill level inherited fields are set to true.</td>
<td>New York</td>
</tr>
<tr>
<td>Skills MGMT: Create a skill from Manage IT Skills user interface.</td>
<td>Verify that you can create a skill from the Manage IT Skills user interface.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Skills MGMT: Add skills and skill levels to users in Manage IT Skills user interface.</td>
<td>Verify that you can add skills and associated skill levels to users in the Manage IT Skills user interface.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Skills MGMT: Select a skill and add users to the skill in Manage IT Skills user interface.</td>
<td>Verify that you can select a skill and add the skill and associated skill levels to one or more users in the Manage IT Skills user interface.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Skills MGMT: Select a skill that does not have a skill level and add users to that skill in the</td>
<td>Verify that you can select a skill that does not have a skill level and add the skill to one or more users</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
Skills Management test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manage IT Skills user</td>
<td>in the Manage IT skills user interface.</td>
<td></td>
</tr>
<tr>
<td>interface.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

To learn more about Skills Management, see Skills Management.

Test Management 2.0

Test Management 2.0 quick start tests require activating the Test Management 2.0 plugin (com.snc.test_management.2.0), and the Test Management 2.0 - ATF Tests plugin (com.snc.test_management.2.0.atf).

Test Management 2.0: Test version test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create test version should create test</td>
<td>Validate test creation and version.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Should be able to mark test version as ready</td>
<td>Validate test state when test has verification steps.</td>
<td>Madrid</td>
</tr>
<tr>
<td>when it contains verification steps</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Should not able to mark test version as ready</td>
<td>Validate test state when test does not have verification steps.</td>
<td>Madrid</td>
</tr>
<tr>
<td>when it does not contain verification step</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Marking a test version as ready should retire</td>
<td>Validate test state when marking test ready.</td>
<td>Madrid</td>
</tr>
<tr>
<td>other test version in ready state</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Test Management 2.0: Test results rollup test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>When test run closed, should update</td>
<td>Validate execution state progress.</td>
<td>Madrid</td>
</tr>
<tr>
<td>execution suite progress</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Should not be able to assign a test not in</td>
<td>Validate test assignment.</td>
<td>Madrid</td>
</tr>
<tr>
<td>ready state</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Test Management 2.0: Test results rollup test suite (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test progress should roll up for test plan and test cycle</td>
<td>Validate test progress for test plan and test cycle.</td>
<td>Madrid</td>
</tr>
</tbody>
</table>

To learn more about Test Management 2.0, see Test Management 2.0.

Vendor Manager Workspace

Vendor Manager Workspace quick start tests require activating the Vendor Manager Workspace plugin (com.snc.vlm.vmw) and loading demo data.

<table>
<thead>
<tr>
<th>Vendor Manager Workspace - Tests test suite</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>VLM: Create a Vendor</td>
</tr>
</tbody>
</table>

To learn more about Vendor Manager Workspace, see Vendor Manager Workspace.

Vendor Risk Management

GRC: Vendor Risk Management quick start tests require activating the Vendor Risk Management plugin (com.sn_vdr_risk_asmt) and loading demo data.

<table>
<thead>
<tr>
<th>GRC: Vendor Risk Management Quick Start Tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>GRC: Create Engagement Assessment</td>
</tr>
<tr>
<td>GRC: Create Vendor Assessment</td>
</tr>
</tbody>
</table>
### GRC: Vendor Risk Management Quick Start Tests (continued)

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRC: Vendor Portal - Answer and Return Assessment</td>
<td>Vendor contact answers and submits assessment in the Service Vendor Portal.</td>
<td>Quebec (compatible with Paris and Orlando)</td>
</tr>
<tr>
<td>GRC: Vendor Tiering Assessment</td>
<td>Selects and submits an assessment to respective assessors after changing the duration.</td>
<td>Quebec (compatible with Paris and Orlando)</td>
</tr>
</tbody>
</table>

### Vulnerability Response

Vulnerability Response quick start tests require activating the Vulnerability Response application (sn_vul) and loading the demo data.

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>VR: Create Remediation Target Rule</td>
<td>Create a Remediation Target Rule.</td>
<td>Madrid</td>
</tr>
<tr>
<td>VR: Create Vulnerability Group Rule</td>
<td>Create a Vulnerability Group Rule.</td>
<td>Madrid</td>
</tr>
<tr>
<td>VR: Create Vulnerable Item via Form</td>
<td>Determine whether a user can successfully create a vulnerable item from the Vulnerable Item form.</td>
<td>Madrid</td>
</tr>
<tr>
<td>VR: Vulnerability Group Life Cycle</td>
<td>Determine whether a user can successfully resolve a vulnerability group.</td>
<td>Madrid</td>
</tr>
<tr>
<td>VR: Vulnerable Item life cycle</td>
<td>Determine whether a user can successfully move a vulnerable item through its life cycle, and also determine whether a closed vulnerable item can be reopened.</td>
<td>Madrid</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>VR: Rollup Calculator</td>
<td>Determine whether the rollup risk calculator can provide an overall risk score for an entire group of vulnerable items using the scores for all the vulnerable items in a vulnerability group.</td>
<td>New York</td>
</tr>
<tr>
<td>VR: Vulnerability Response Assignment Rules</td>
<td>Determine whether a sample set of assignment rules can successfully auto-assign vulnerable items to an assignment group for remediation.</td>
<td>New York</td>
</tr>
<tr>
<td>VR: Vulnerability Calculators</td>
<td>Test the vulnerability calculators.</td>
<td>New York</td>
</tr>
<tr>
<td>VR: CI Lookup - Qualys</td>
<td>Create a new lookup rule with method &quot;field_matching&quot; called &quot;Lookup By Network Adapter&quot; for Qualys. Determine whether a configuration item is successfully matched in the Discovered Item table by network adapter and IP address with the new lookup rule.</td>
<td>Orlando</td>
</tr>
<tr>
<td>VR: Create Normal and Emergency Change Request</td>
<td>Determine whether the user can successfully create normal and emergency change requests from a vulnerability group.</td>
<td>Orlando</td>
</tr>
<tr>
<td>VR: Split Vulnerability Group</td>
<td>Determine whether the user can successfully split a vulnerability group.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Test</td>
<td>Description</td>
<td>Release version</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>VR: Update VG state when a CHG is cancelled.</td>
<td>Determine whether the State field on a vulnerability group successfully transitions when a change request that is associated with the vulnerability group is cancelled.</td>
<td>Orlando</td>
</tr>
<tr>
<td>VR: Update VG state when a CHG transitions to Review.</td>
<td>Determine whether the State field on a vulnerability group successfully transitions when a change request that is associated with it moves to the Review state.</td>
<td>Orlando</td>
</tr>
<tr>
<td>VR: CI Lookup - Rapid7</td>
<td>Test CI lookup using the existing Rapid7 Vulnerability Integration lookup rule, <strong>IP Address</strong>.</td>
<td>Orlando</td>
</tr>
<tr>
<td>VR: CI lookup - Qualys</td>
<td>Test CI lookup by creating a new lookup rule for the Qualys Vulnerability Integration</td>
<td>Orlando</td>
</tr>
<tr>
<td>VR: Exception Approval Workflow for VI</td>
<td>Create an exception request and verify that the approval process is working.</td>
<td>Orlando</td>
</tr>
<tr>
<td>VR: False Positive Approval Workflow for VI</td>
<td>Create a false positive exception request and verify that the approval process is working.</td>
<td>Orlando</td>
</tr>
<tr>
<td>VR: Application Vulnerability Response (AVR)</td>
<td>Determine whether your rules and calculators are working correctly. Verify that updates are working.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>
## Test Description | Release version
--- | ---
Remediation target rules: VI import test | Tests VR remediation target rules during import | Paris

To learn more about Vulnerability Response, see [Vulnerability Response](#).

### Walk-up Experience

Walk-up Experience quick start tests require activating the Walk-up Experience plugin (com.snc.walkup) and loading demo data.

#### Walk-up Experience - Tests test suite

<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
<th>Release version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a location</td>
<td>Ensure users with the Walk-up administrator [sn_walkup.walkup_admin] role can access all existing walk-up queues, create new queue locations, and configure queues appropriately.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Onsite checkin (ITIL User)</td>
<td>Verify that users with the ITIL role can check into an onsite queue location.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Onsite checkin (ESS User)</td>
<td>Verify that users with the ESS role can check into an onsite queue location.</td>
<td>Orlando</td>
</tr>
<tr>
<td>Onsite checkin (Guest)</td>
<td>Verify that guest users can check into an onsite queue location.</td>
<td>Orlando</td>
</tr>
</tbody>
</table>

To learn more about Walk-up Experience, see [Walk-up Experience](#).

### Test step categories

Find test steps for a particular user interface or Now Platform feature.

The Add Test Step panel lists test categories in the left pane. You can directly search for a specific test step by its name in the search field at the top of the
You can also select **All Steps** in the left pane to list all available test steps in the middle pane. A description of the selected step appears in the right pane.

<table>
<thead>
<tr>
<th>Panel</th>
<th>Form</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Steps</td>
<td>Open a New Form</td>
<td>Service Catalog</td>
</tr>
<tr>
<td>Form</td>
<td>Open an Existing Record</td>
<td><strong>Open a Catalog Item</strong></td>
</tr>
<tr>
<td>Service Catalog in Service Portal</td>
<td>Set Field Values</td>
<td>Open a Catalog Item</td>
</tr>
<tr>
<td>Application Navigator</td>
<td>Field Values Validation</td>
<td>Opens a catalog item.</td>
</tr>
<tr>
<td>Custom UI</td>
<td>Field State Validation</td>
<td><strong>Additional Considerations</strong></td>
</tr>
<tr>
<td>Service Catalog</td>
<td>UI Action Visibility</td>
<td>Optionally, you can specify the form's view name. Keep in mind that this can only be done for users that have access to that view.</td>
</tr>
<tr>
<td>Forms in Service Portal</td>
<td>Add Attachments to Form</td>
<td></td>
</tr>
<tr>
<td>REST</td>
<td>Click Modal Button</td>
<td></td>
</tr>
<tr>
<td>Responsive Dashboards</td>
<td>Click a UI Action</td>
<td></td>
</tr>
<tr>
<td>Server</td>
<td>Submit a Form</td>
<td></td>
</tr>
<tr>
<td>Reporting</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

You can select a specific test category to list only the test steps available for that test category in the middle panel. A description of the selected step appears in the right panel.
Service Catalog in Service Portal category
Validate catalog item transactions and requester flows from Service Portal.

Activation of the Automated Test Framework for Service Catalog in Service Portal
These ATF test steps require activation of the Automated Test Framework Service Catalog Service Portal (com.glide.automated_testing_impl.service_catalog_portal) plugin, which is active by default on new instances. Administrators may need to activate the plugin on instances upgraded from earlier versions.

Support for parametrized tests
Service Catalog in Service Portal step configurations support parametrized tests. For more information on parametrized tests, refer to Parameterized tests.

Open a Record Producer (SP)
Open a record producer in the Service Portal.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Portal</td>
<td>Portal for which you want to test this step.</td>
</tr>
<tr>
<td>Page</td>
<td>Service Catalog page associated with the test step.</td>
</tr>
</tbody>
</table>
### Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Record Producer</td>
<td>Record producer that you want to open.</td>
</tr>
<tr>
<td></td>
<td>⚠️ <strong>Note:</strong> You should have access to the record producer.</td>
</tr>
<tr>
<td>Query Parameters</td>
<td>URL query parameters for the page, such as <code>sys_id</code>.</td>
</tr>
</tbody>
</table>

### Open a Catalog Item (SP)

Open a catalog item in the Service Portal.

### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Portal</td>
<td>Portal for which you want to test this step.</td>
</tr>
<tr>
<td>Page</td>
<td>Service Catalog page associated with the test step.</td>
</tr>
<tr>
<td>Catalog Item</td>
<td>Catalog item that you want to open.</td>
</tr>
<tr>
<td></td>
<td>⚠️ <strong>Note:</strong> You should have access to this catalog item.</td>
</tr>
<tr>
<td>Query Parameters</td>
<td>URL query parameters for the page, such as <code>sys_id</code>.</td>
</tr>
</tbody>
</table>
Open an Order Guide (SP)

Open an order guide in the Service Portal.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Portal</td>
<td>Portal for which you want to test this step.</td>
</tr>
<tr>
<td>Page</td>
<td>Service Catalog page associated with the test step.</td>
</tr>
<tr>
<td>Order Guide</td>
<td>Order guide that you want to open.</td>
</tr>
</tbody>
</table>

**Note:** You should have access to this order guide.
Add row to multi-row variable set (SP)

Add a row to a multi-row variable set included in the current catalog item in Service Portal. You can use this step configuration only when the current catalog item contains a multi-row variable set. Use this step after the Open a Catalog Item(SP) step, Open a Record Producer(SP) step, or Open an Order Guide (SP) step. If a contextual value has been used for the Open a Catalog Item(SP) step, Open a Record Producer(SP) step, or Open an Order Guide (SP) step, set the catalog item in this step.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Catalog Item</td>
<td>Catalog item whose multi-row variable set requires an addition of a row. If an item is not already selected, you can either search for the item, or insert a reference to the contextual value of the item from a list of available parameters.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-Row Variable Set</td>
<td>Multi-row variable set for which a row should be added.</td>
</tr>
</tbody>
</table>

Save current row of multi-row variable set (SP)

Save the current row of a multi-row variable set included in the current catalog item in Service Portal. You can use this step configuration only when the current catalog item contains a multi-row variable set. Use this step after the Add row to multi-row variable set (SP) step.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Assert Type</td>
<td>Criteria for the test to pass.</td>
</tr>
</tbody>
</table>

Successfully saved row to a multi-row variable set

Test passes only if the current row of the multi-row variable is saved.

Cannot save a row to multi-row variable set
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Test passes only if the current row of the multi-row variable is not saved.</td>
</tr>
</tbody>
</table>

**Set Variable Values (SP)**

Set variable values for a catalog item or record producer in the Service Portal.

For a catalog item, use this step after using the Open a Catalog Item (SP) step, and before using the Order Catalog Item (SP) step. For a record producer, use this step after using the Open a Record Producer (SP) step, and before using the Submit Record Producer (SP) step.

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Use the condition builder to set the field value. The condition builder displays an appropriate control for the field data type. For example, a reference field displays a <strong>Lookup record</strong> control.</td>
</tr>
<tr>
<td>Item</td>
<td>Catalog item or record producer for which you want to set variable values.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-Row Variable Set</td>
<td>Multi-row variable set for which variable values should be set. <strong>Note:</strong> Use the Add row to multi-row variable set (SP) step configuration prior to the current step configuration.</td>
</tr>
<tr>
<td>Variable Values</td>
<td>List of variables and the values that you want to set for them. <strong>Note:</strong> You can set the value for multiple variables.</td>
</tr>
</tbody>
</table>

**Note:** Custom variables and custom variable with labels are not supported for Set Variable Values step configuration.

**Validate Variable Values (SP)**

Validates variable values of a catalog item or record producer in Service Portal. For a catalog item, use this step after using the Open a Catalog Item (SP) step, and before using the Order Catalog Item (SP) step. For a record producer, use this step after using the Open a Record Producer (SP) step, and before using the Submit Record Producer (SP) step.

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Note:</td>
<td>Use the condition builder to set the field value. The condition builder displays an appropriate control for the field data type. For example, a reference field displays a <strong>Lookup record</strong> control.</td>
</tr>
<tr>
<td>Item</td>
<td>Catalog item or record producer whose variables should be validated.</td>
</tr>
<tr>
<td>Multi-Row Variable Set</td>
<td>Multi-row variable set for which variable values should be validated.</td>
</tr>
<tr>
<td>Note:</td>
<td>Use the Add row to multi-row variable set (SP) step configuration prior to the current step configuration.</td>
</tr>
<tr>
<td>Catalog Conditions</td>
<td>Conditions for variable validation. If the conditions are met, the test passes.</td>
</tr>
<tr>
<td>Note:</td>
<td>The label of a variable associated with a variable set reflects the variable set name. The format is <code>variable_set_name » variable_name</code>.</td>
</tr>
</tbody>
</table>

**Note:** Custom variables and custom variable with labels are not supported for Validate Variable Values step configuration.

**Variable State Validation (SP)**

Validates the state of variables in Service Portal. Possible variable states are mandatory, not mandatory, read only, not read only, visible, and not visible.

### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute</td>
</tr>
</tbody>
</table>
### Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Item</td>
<td>Catalog item or record producer whose variables should be validated.</td>
</tr>
<tr>
<td>Multi-Row Variable Set</td>
<td>Multi-row variable set for which variable states should be validated.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Use the Add row to multi-row variable set (SP) step configuration prior to the current step configuration.</td>
</tr>
<tr>
<td>Visible</td>
<td>List of the catalog item variables that must be visible for the step to pass.</td>
</tr>
<tr>
<td>Not visible</td>
<td>List of the catalog item variables that must be hidden for the step to pass.</td>
</tr>
<tr>
<td>Read only</td>
<td>List of the catalog item variables that must be read-only for the step to pass.</td>
</tr>
<tr>
<td>Not read only</td>
<td>List of the catalog item variables that must not be read-only for the step to pass.</td>
</tr>
<tr>
<td>Mandatory</td>
<td>List of the catalog item variables that must be mandatory for the step to pass.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Not mandatory</td>
<td>List of the catalog item variables that must not be mandatory for the step to pass.</td>
</tr>
</tbody>
</table>

**Note:** Custom variables and custom variable with labels are not supported for Variable State Validation step configuration.

**Validate Price and Recurring Price (SP)**

Validate the price and recurring price of a catalog item in Service Portal. Use this step after using the Open a Catalog Item (SP) step, and before using the Order Catalog Item (SP) step. This step is not applicable for a record producer.

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Price</td>
<td>Price of the catalog item for the step to pass.</td>
</tr>
<tr>
<td>Recurring price</td>
<td>Recurring price of the catalog item for the step to pass.</td>
</tr>
<tr>
<td>Recurring price frequency</td>
<td>Recurring price frequency of the catalog item for the step to pass.</td>
</tr>
</tbody>
</table>
Navigate within Order Guide (SP)

Navigate within an order guide.

## Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
</tbody>
</table>
| Guide Step           | Order guide step to which you want to navigate:  
  • Describe Needs
  • Choose Options
  • Summary                                                                                                                                       |
| Assert Type          | Criteria for the test to pass.  
  **Navigation Successful**                                                                                                                      |
### Set Catalog Item Quantity (SP)

Set the quantity for a catalog item in Service Portal. This step is not applicable for a record producer. Use this step after using the Open a Catalog Item (SP) step, and before using the Order Catalog Item (SP) step.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Execution order</strong></td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Quantity</td>
<td>Quantity of the catalog item that you want to order.</td>
</tr>
</tbody>
</table>

### Validate Order Guide Items (SP)

Validate items included in the order guide.
**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Catalog Items</td>
<td>Catalog items that you want to validate.</td>
</tr>
</tbody>
</table>

**Note:** You should have access to these catalog items.

**Review Order Guide Summary (SP)**

Review the order guide summary in the Service Portal.
### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Catalog Items</td>
<td>Catalog items that you want to review.</td>
</tr>
<tr>
<td>Price</td>
<td>Price of the catalog item for the step to pass.</td>
</tr>
</tbody>
</table>

**Note:** You should have access to these catalog items.

### Review Item in Order Guide (SP)

Review individual items in the order guide and choose whether or not to include the item.
### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Catalog Item</td>
<td>Catalog item that you want to review. <strong>Note:</strong> You should have access to this catalog item.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included</td>
<td>Selected if the catalog item should be included in the order guide, otherwise unselected.</td>
</tr>
</tbody>
</table>

Add Item to Shopping Cart (SP)

Add the current catalog item to the shopping cart in Service Portal. Use this step after using the Open a Catalog Item (SP) step.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Assert Type</td>
<td>Criteria for the test to pass.</td>
</tr>
</tbody>
</table>

**Successfully added item to Shopping Cart**

Test passes only if the catalog item is successfully added to the shopping cart.
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cannot add item to Shopping Cart</td>
<td>Test passes only if the catalog item cannot be added to the shopping cart.</td>
</tr>
</tbody>
</table>

Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cart_item_id</td>
<td>The sys_id of the added catalog item.</td>
</tr>
</tbody>
</table>

Add Order Guide to Shopping Cart (SP)

Add an order guide to the shopping cart.

Input

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Assert Type</td>
<td>Criteria for the test to pass.</td>
</tr>
<tr>
<td></td>
<td><strong>Cannot add order guide to shopping cart</strong></td>
</tr>
<tr>
<td></td>
<td>Test passes only if the order guide cannot be added to the shopping cart.</td>
</tr>
<tr>
<td></td>
<td><strong>Successfully added order guide to shopping cart</strong></td>
</tr>
<tr>
<td></td>
<td>Test passes only if the order guide is successfully added to the shopping cart.</td>
</tr>
</tbody>
</table>

**Order a Catalog Item (SP)**

Click **Order Now** for the current catalog item in the Service Portal. Use this step after using the Open a Catalog Item (SP) step. After this step, you cannot use any other steps on the catalog item. If the two-step checkout is false, a request is generated for the catalog item. If the two-step checkout is true, you are redirected to the cart preview page.
### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Assert type</td>
<td>Criteria for the test to pass.</td>
</tr>
<tr>
<td></td>
<td>Successfully ordered Catalog Item</td>
</tr>
<tr>
<td></td>
<td>Test passes only if the catalog item is successfully ordered.</td>
</tr>
<tr>
<td></td>
<td>Cannot order Catalog Item</td>
</tr>
<tr>
<td></td>
<td>Test passes only if the catalog item cannot be ordered.</td>
</tr>
</tbody>
</table>

### Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>The table to which the submitted request belongs.</td>
</tr>
<tr>
<td>record_id</td>
<td>The sys_id of the submitted request.</td>
</tr>
</tbody>
</table>
Submit an Order Guide (SP)

Click **Order Now** to order an order guide. Do not add more than one record producer to the order guide.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Assert Type</td>
<td>Criteria for the test to pass.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Test passes only if the order guide cannot be ordered.</td>
</tr>
<tr>
<td></td>
<td><strong>Successfully ordered the Order Guide</strong></td>
</tr>
<tr>
<td></td>
<td>Test passes only if the order guide is successfully ordered.</td>
</tr>
</tbody>
</table>

Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>The table containing the submitted order guide.</td>
</tr>
<tr>
<td>record_id</td>
<td>The sys_id of the submitted order guide.</td>
</tr>
</tbody>
</table>

Submit Record Producer (SP)

Submit the current record producer in the Service Portal. Use this step after using the Open a Record producer (SP) step. After this step, you cannot use any other steps on the record producer.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <code>Execution order</code> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
</tbody>
</table>
## Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Assert Type</td>
<td>Criteria for the test to pass.</td>
</tr>
</tbody>
</table>

**Successfully submitted Record Producer**

Test passes only if the record producer is submitted successfully.

**Cannot submit Record Producer**

Test passes only if the record producer cannot be submitted.

## Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>The table containing target record of the record producer.</td>
</tr>
</tbody>
</table>

## Application Navigator category

Verify the functionality of menus and modules in the application navigator.

## Application Menu Visibility

Verifies the visibility, or lack thereof, of selected application menus in the application navigator (left navigation bar). For example, you create a test that first impersonates a user, then verifies that specified application menus (such as Self-Service and Reports) are visible, or are not visible, to that user.

## Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
</tbody>
</table>
### Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Active</strong></td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td><strong>Application</strong></td>
<td>The application scope in which the system runs this test or test suite.</td>
</tr>
<tr>
<td><strong>Test</strong></td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td><strong>Step config</strong></td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td><strong>Notes</strong></td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td><strong>Navigator</strong></td>
<td>Navigator version to assert against</td>
</tr>
<tr>
<td></td>
<td>• <strong>UI15</strong>: If you have existing steps with UI15, you can change it to UI16</td>
</tr>
<tr>
<td></td>
<td>• <strong>UI16</strong>: If you are creating new steps, you will have UI16 by default.</td>
</tr>
<tr>
<td><strong>Visible assert type</strong></td>
<td>Specifies how application menus selected in the <strong>Visible application menus</strong> field should be tested for visibility in the application navigator.</td>
</tr>
<tr>
<td></td>
<td>• <strong>At least these application menus are visible</strong>: At minimum, all the selected application menus are visible in the application navigator.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Only these application menus are visible</strong>: Only the selected application menus are visible in the application navigator.</td>
</tr>
<tr>
<td><strong>Visible application menus</strong></td>
<td>Application menus whose visibility in the application navigator is being verified.</td>
</tr>
<tr>
<td><strong>Not visible assert type</strong></td>
<td>Specifies how application menus selected in the <strong>Not visible application menus</strong> field should be tested for lack of visibility in the application navigator.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• At least these application menus are not visible:</td>
<td>At minimum, all the selected application menus are not visible in the application navigator.</td>
</tr>
<tr>
<td>• Only these application menus are not visible:</td>
<td>Only the selected application menus are not visible in the application navigator.</td>
</tr>
<tr>
<td>Not visible application menus</td>
<td>Application menus whose lack of visibility in the application navigator is being verified.</td>
</tr>
</tbody>
</table>

Related information

Create an application menu

Module Visibility

Verify the visibility, or lack thereof, of selected modules in the application navigator (left navigation bar). For example, create a test that first impersonates a user, then verifies that specified modules (such as Homepage and My Requests) are visible, or are not visible, to that user.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>The application scope in which the system runs this test or test suite.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
</tbody>
</table>
## Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Navigator</td>
<td>Navigator version to assert against</td>
</tr>
<tr>
<td></td>
<td>• UI15: If you have existing steps with UI15, you can change it to UI16</td>
</tr>
<tr>
<td></td>
<td>• UI16: If you are creating new steps, you will have UI16 by default.</td>
</tr>
<tr>
<td>Visible assert type</td>
<td>Specifies how modules selected in the <strong>Visible modules</strong> field should be</td>
</tr>
<tr>
<td></td>
<td>tested for visibility in the application navigator.</td>
</tr>
<tr>
<td></td>
<td>• <strong>At least these modules are visible:</strong> At minimum, the modules selected</td>
</tr>
<tr>
<td></td>
<td>in the <strong>Visible modules</strong> field are visible in the application navigator.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Only these modules are visible:</strong> Only the specific modules selected in</td>
</tr>
<tr>
<td></td>
<td>the <strong>Visible modules</strong> field are visible in the application navigator.</td>
</tr>
<tr>
<td>Visible modules</td>
<td>Modules whose visibility in the application navigator is being verified.</td>
</tr>
<tr>
<td>Not visible assert type</td>
<td>Specifies how modules selected in the <strong>Not visible modules</strong> field should</td>
</tr>
<tr>
<td></td>
<td>be tested for lack of visibility in the application navigator.</td>
</tr>
<tr>
<td></td>
<td>• <strong>At least these modules are not visible:</strong> At minimum, the modules</td>
</tr>
<tr>
<td></td>
<td>selected in the <strong>Not visible modules</strong> field are not visible in the</td>
</tr>
<tr>
<td></td>
<td>application navigator.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Only these modules are not visible:</strong> Only the specific modules selected</td>
</tr>
<tr>
<td></td>
<td>in the <strong>Not visible modules</strong> field are not visible in the application</td>
</tr>
<tr>
<td></td>
<td>navigator.</td>
</tr>
<tr>
<td>Not visible modules</td>
<td>Modules whose lack of visibility in the application navigator is being</td>
</tr>
<tr>
<td></td>
<td>verified.</td>
</tr>
</tbody>
</table>

### Related information

**Create a module**

**Navigate to Module**

Open a module from the application navigator, as if a user had clicked it. The module must be visible to the currently executing user to navigate to it.
Note: Not all pages are currently testable. Wherever the module takes you is your responsibility.

### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Module</td>
<td>Module that should be opened. To navigate to the selected module, the module must be visible to the currently executing user in the application navigator. The following modules are not supported and cannot be tested:</td>
</tr>
<tr>
<td></td>
<td>• Modules that are separators</td>
</tr>
<tr>
<td></td>
<td>• Modules that do not link to a specific page, but instead execute client-side JavaScript (such as Studio and the Script Debugger)</td>
</tr>
<tr>
<td></td>
<td>• Modules that link to external websites, such as the ServiceNow documentation site (docs.servicenow.com)</td>
</tr>
<tr>
<td></td>
<td>• Modules that reload or redirect the entire page</td>
</tr>
</tbody>
</table>
Related information

Create a module

Custom UI category

Validate the behavior of page components on custom user interfaces.

Set Component Values (Custom UI)

Set component values on a custom UI page.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Component values</td>
<td>Page components to be tested and the value each page component should have for the step to succeed.</td>
</tr>
</tbody>
</table>
### Assert Text on Page (Custom UI)

Assert that the specified text is or is not on a custom UI page.

#### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This</td>
</tr>
</tbody>
</table>

**Note:** The Value field defaults to the last retrieved value of the selected component.
**Component Value Validation (Custom UI)**

Validate a component value on a custom UI page.

<table>
<thead>
<tr>
<th><strong>Inputs</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Field</strong></td>
</tr>
<tr>
<td>Execution order</td>
</tr>
<tr>
<td>Active</td>
</tr>
<tr>
<td>Application</td>
</tr>
<tr>
<td>Test</td>
</tr>
<tr>
<td>Step config</td>
</tr>
</tbody>
</table>

**Note:** The text to be searched is case-sensitive.

**Note:** Starting with the Rome release, this test step also works for Workspace for any new tests.

**Note:** The Assert Text on Page test step also accepts text containing white spaces.
### Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Component values</td>
<td>Page components to be tested and the value each page component should have for the step to succeed.</td>
</tr>
</tbody>
</table>

**Note:** The **Value** field defaults to the last retrieved field value of the selected component.

### Click Component (Custom UI)

Click a component on a custom UI page.

### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
</tbody>
</table>
Component State Validation (Custom UI)

Validate the state of a specified component on a custom UI page.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This</td>
</tr>
</tbody>
</table>
Open Service Portal Page
Open a portal page. Test designers must first open a page in a portal before testing UI components on the page.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Execution order | Integer specifying the order in which the test executes this step.  
As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values. |
| Active        | Option to activate this test step for use.                                                                                                   |
| Application   | Application scope in which the system runs this step.                                                                                       |
| Test          | Read-only name of the test that you're adding the step to.                                                                                   |
| Step config   | Read-only name of the step.                                                                                                                  |
| Description   | Description of the test step. This field value is automatically set based on...                                                             |
Form category

Validate the functionality of fields and UI actions on a form.

Note: Test steps that include the Form UI field give you the option to select an available UI. For any available workspace, navigation between tabs is not supported. Use the Open a New Form or Open an Existing Record step to reopen a form.

Open a New Form

Open a form to a new record in the specified table and Form UI.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
</tbody>
</table>
### Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Form UI</td>
<td>Option to select an available UI.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Workspaces are not supported in Internet Explorer (IE). See <a href="#">KB0683275</a> for more details.</td>
</tr>
<tr>
<td>Table</td>
<td>Name of the table for the new form.</td>
</tr>
<tr>
<td>View</td>
<td>Name of the view in which you want this form to open. The testing user must have access to that view. If the name is not a valid form view, the form opens in its default view.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This field appears when <strong>Standard UI</strong> is chosen from <strong>Form UI</strong>.</td>
</tr>
</tbody>
</table>

### Open an Existing Record

Open a form to an existing record in the specified table and **Form UI**.

**Note:** Using an existing record may cause unexpected behavior for this test. See [Automated Test Framework design considerations](#) for more information.

### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
</tbody>
</table>
### Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Form UI</td>
<td>Option to select an available UI.</td>
</tr>
<tr>
<td>![Note:](Note: Workspaces are not supported in Internet Explorer (IE). See KB0683275 for more details.)</td>
<td></td>
</tr>
<tr>
<td>Table</td>
<td>Name of the table for the record you want to open.</td>
</tr>
<tr>
<td>Record</td>
<td>Record ID of the record that you want to open and name of the document that you want to open.</td>
</tr>
<tr>
<td>View</td>
<td>Name of the view in which you want this form to open. The testing user must have access to that view. If the name is not a valid form view, the form opens in its default view.</td>
</tr>
<tr>
<td>![Note:](Note: This field appears when Standard UI is chosen from Form UI.)</td>
<td></td>
</tr>
</tbody>
</table>

### Set Field Values

Set the fields on the current form to the specified values.

To run this step, your test must have already opened a form using either the **Open a New Form** or **Open an Existing Record** step. It is recommended to not run this step directly after a **Submit a Form** or **Click a UI Action** step. This is because they can redirect your test to a different page based on the navigation stack configuration on your instance or the script defined in the clicked UI action. Unless you are certain that the UI action will take you to a specific page, you should explicitly use an **Open a New Form** step after **Submit a Form** or **Click a UI Action** to ensure that the test is on the form as expected. Ensure that the test keeps passing consistently when added to a suite.
The **Field Values Validation**, **Set Field Values**, **Field State Validation**, and **UI Action Visibility** steps can appear in any order.

ℹ️ **Note:** This step waits for the form to load before setting field values.

ℹ️ **Note:** This step doesn't support reference qualifiers, neither at test design time nor at test runtime.

ℹ️ **Note:** A modal form appears either on top of another form or a list. To submit a modal form after setting the field values, your test must have already opened it on top of a form or a list.

---

![Image of test step configuration](image)

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>The application scope in which the system runs this test or test suite.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Note:</td>
<td>Use the condition builder to set the field value. The condition builder displays an appropriate control for the field data type. For example, a reference field displays a <strong>Lookup record</strong> control.</td>
</tr>
<tr>
<td>Form UI</td>
<td>Option to select an available UI.</td>
</tr>
<tr>
<td>Note:</td>
<td>Workspaces are not supported in Internet Explorer (IE). See <a href="#">KB0683275</a> for more details.</td>
</tr>
<tr>
<td>Table</td>
<td>Table that contains the field whose value you want to set.</td>
</tr>
<tr>
<td>Set field value</td>
<td>Assigns a value to a field on an open form.</td>
</tr>
</tbody>
</table>

**Field Values Validation**

Validate field values on the current form.

To run this step, your test must have already opened a form using either the **Open a New Form** or **Open an Existing Record** step. It is recommended to not run this step directly after a **Submit a Form** or **Click a UI Action** step. This is because they can redirect your test to a different page based on the navigation stack configuration on your instance or the script defined in the clicked UI action. Unless you are certain that the UI action will take you to a specific page, you should explicitly use an **Open a New Form** step after **Submit a Form** or **Click a UI Action** to ensure that the test is on the form as expected. Ensure that the test keeps passing consistently when added to a suite.

The **Field Values Validation**, **Set Field Values**, **Field State Validation**, and **UI Action Visibility** steps can appear in any order.

For the **Field Values Validation** step, specify the values that you want to test using the standard conditions builder. You can test several conditions against the same field. This step passes if the overall condition is satisfied and fails if the condition is not satisfied. The **Conditions** field is case-sensitive and requires
to have the exact value as on the form. To test the values of individual fields independently of each other, include a separate Field Values Validation step for each value that you test.

**Note:** The Field Values Validation step works only with fields that belong to the record for the open form. For example, with the incident table, this step is not able to validate the Additional comments, Approval history, Comments, or Work notes fields because these UI controls are not actual fields on the incident record. These UI controls make it convenient to work with related tables. To validate these cases, use the Server test step, Record Validation, instead.

### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
</tbody>
</table>

**Note:** Use the condition builder to set the field value. The condition builder displays an appropriate control for the field data type. For example, a reference field displays a Lookup record control.

| Form UI | Option to select an available UI. |
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note</td>
<td>Workspaces are not supported in Internet Explorer (IE). See KB0683275 for more details.</td>
</tr>
</tbody>
</table>

Table

Table that contains the fields whose values you want to validate.

Conditions

Fields and values to be validated. Includes only fields that are visible on the open form. This field is case-sensitive.

**Field State Validation**

Validate the state of specified fields. States validated can include mandatory, non-mandatory, read-only, non-read-only, visible, and non-visible.

You can specify a maximum time to wait for the states of the fields to match the conditions in this step.

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form UI</td>
<td>Option to select an available UI.</td>
</tr>
<tr>
<td></td>
<td>❚ Note: Workspaces are not supported in Internet Explorer (IE). See KB0683275 for more details.</td>
</tr>
<tr>
<td>Table</td>
<td>Table that contains the fields whose states you want to validate.</td>
</tr>
<tr>
<td>Visible</td>
<td>Validates whether the fields on this form are visible. The test fails if the fields are not visible.</td>
</tr>
<tr>
<td>Not visible</td>
<td>Validates whether the fields on this form are visible. The test fails if the fields are visible.</td>
</tr>
<tr>
<td>Read only</td>
<td>Validates whether the fields on this form are read only. The test fails if the fields are not read only.</td>
</tr>
<tr>
<td>Not read only</td>
<td>Validates whether the fields on this form are read only. The test fails if the fields are read only.</td>
</tr>
<tr>
<td>Mandatory</td>
<td>Validates whether the fields on this form are mandatory. The test fails if the fields are not mandatory.</td>
</tr>
<tr>
<td>Not Mandatory</td>
<td>Validates whether the fields on this form are mandatory. The test fails if the fields are mandatory.</td>
</tr>
</tbody>
</table>

**UI Action Visibility**

Verify if a UI action is visible on the current form. To run this step, your test must have already opened a form using either the **Open a New Form** or **Open an Existing Record** step.

The default visible UI actions vary depending on the user that you’re currently impersonating.

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
</tbody>
</table>
## Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails,</td>
</tr>
<tr>
<td></td>
<td>the system repeats the step until it reaches the duration of the timeout.</td>
</tr>
<tr>
<td></td>
<td>If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based</td>
</tr>
<tr>
<td></td>
<td>on the field values of the test step. This field appears after the test step</td>
</tr>
<tr>
<td></td>
<td>is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Form UI</td>
<td>Option to select an available UI.</td>
</tr>
<tr>
<td></td>
<td><img src="https://example.com" alt="Note:" /> Workspaces are not supported in Internet</td>
</tr>
<tr>
<td></td>
<td>Explorer (IE). See <a href="https://example.com">KB0683275</a> for more details.</td>
</tr>
<tr>
<td>Table</td>
<td>The table with the UI actions to be tested.</td>
</tr>
<tr>
<td>Visible</td>
<td>Validates whether UI actions on this form are visible. The test fails if</td>
</tr>
<tr>
<td></td>
<td>the UI actions are not visible.</td>
</tr>
<tr>
<td>Not Visible</td>
<td>Validates whether UI actions on this form are visible. The test fails if</td>
</tr>
<tr>
<td></td>
<td>the UI actions are visible.</td>
</tr>
</tbody>
</table>

### Add Attachments to Form

Add one or more mandatory attachments to the current form. Select the attachments that the test step adds to the form from the Upload Attachments list.

## Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an</td>
</tr>
<tr>
<td></td>
<td>incremental value. This value causes the test to execute</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Test</td>
<td>Name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Upload Attachments</td>
<td>Button to add one or more mandatory attachments to the form.</td>
</tr>
</tbody>
</table>

**Click Modal Button**

Click a button within a modal in the specified **Form UI**.

Specify your testing by selecting either the standard platform UI or workspace UI from the **Form UI** field. If you select the standard platform UI, this test step selects the button by ID on the specified modal and validates the following conditions:

- UI page was opened in a modal.
- Button is visible and enabled.

If you select an available workspace UI, this test step selects either the Confirm or Cancel action and optionally sets the field values within the modal. This step succeeds only if a modal dialog is open on the form, and if the specified button exists on that modal dialog.

Only modals opened with the following g_modal functions are supported:

- alert
- confirm
- confirmDestroy
- showFields

⚠️ Note: Click Modal Button now supports global and scoped application modals.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds for the modal and the button to appear.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Form UI</td>
<td>Option to select an available UI.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Workspaces are not supported in Internet Explorer (IE). See <a href="#">KB0683275</a> for more details.</td>
</tr>
<tr>
<td>UI page</td>
<td>The UI modal dialog to be tested.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This field appears only if <strong>Standard UI</strong> is selected from <strong>Form UI</strong>.</td>
</tr>
<tr>
<td>Button</td>
<td>The ID attribute of the button element. For example, the <strong>OK</strong> button has an ID of <strong>OK_button</strong> in the UI modal dialog.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This field appears only if <strong>Standard UI</strong> is selected from <strong>Form UI</strong>.</td>
</tr>
<tr>
<td>Modal values</td>
<td>The field values to be set in the modal.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This field appears only if a workspace is chosen as <strong>Form UI</strong>.</td>
</tr>
<tr>
<td>Modal action</td>
<td>The action to be selected in the modal.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note</td>
<td>This field appears only if a workspace is chosen as Form UI.</td>
</tr>
</tbody>
</table>

**Assert type**
The effect on the tests on clicking the modal button.

- **Modal closed and page is reloaded or redirected**: The test passes only if the modal was closed and the page was reloaded or redirected to another page.

- **Modal closed and page is not reloaded or redirected**: The test passes only if the modal was closed and the page was not reloaded or redirected to another page.

- **Modal not closed**: The test passes only if the modal still exists and was not closed.

**Assertion timeout**
Number of seconds allowed before the assert fails. If the assert fails, the system repeats the assert until the duration of the Timeout is reached. If the validation fails after the Timeout duration has passed, the step fails.

**Click a UI Action**
Click a UI action on the current form.

When this step runs, the system performs the action normally activated by that control. The test step also validates that the current form contains the control and that the control is visible and enabled. To run this step, your test must have already opened a form using either the **Open a New Form** or **Open an Existing Record** step. It is recommended to not run this step directly after a **Submit a Form** or **Click a UI Action** step. This is because they can redirect your test to a different page based on the navigation stack configuration on your instance or the script defined in the clicked UI action. Unless you are certain that the UI action will take you to a specific page, you should explicitly use an **Open a New Form** step after **Submit a Form** or **Click a UI Action** to ensure that the test is on the form as expected. Ensure that the test keeps passing consistently when added to a suite.

In the Rome release, this step supports UI actions of type **Form context menu**.
Note: Don’t write tests that depend on the system displaying a specific page after executing a Submit a Form or Click a UI Action step. After these test steps, the system returns to the page that was open before the form was opened. The test cannot determine what that page was, so writing a test that expects a particular page can lead to unpredictable results.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Form UI</td>
<td>Option to select an available UI.</td>
</tr>
</tbody>
</table>

Note: Workspaces are not supported in Internet Explorer (IE). See KB0683275 for more details.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table</td>
<td>The table containing the UI action to click.</td>
</tr>
<tr>
<td>UI action</td>
<td>The UI action to click.</td>
</tr>
<tr>
<td>Assert type</td>
<td>The effect on the tests with clicking of a UI action.</td>
</tr>
<tr>
<td>• Form submission canceled in browser: The step passes only if form submission is canceled.</td>
<td></td>
</tr>
<tr>
<td>• Form submitted to server: The step passes only if the form is submitted.</td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>The table for the form that contains this UI action.</td>
</tr>
<tr>
<td>record</td>
<td>The sys_id of the record on which the action was clicked.</td>
</tr>
</tbody>
</table>

Submit a Form

Submit the current form.

To run this step, your test must have already opened a form using either the Open a New Form or Open an Existing Record step. It is recommended to not run this step directly after a Submit a Form or Click a UI Action step. This is because they can redirect your test to a different page based on the navigation stack configuration on your instance or the script defined in the clicked UI action. Unless you are certain that the UI action will take you to a specific page, you should explicitly use an Open a New Form step after Submit a Form or Click a UI Action to ensure that the test is on the form as expected. Ensure that the test keeps passing consistently when added to a suite.

ℹ️ Note: Don’t write tests that depend on the system displaying a specific page after executing a Submit a Form or Click a UI Action step. After these test steps, the system returns to the page that was open before the form was opened. The test cannot determine what that page was, so writing a test that expects a particular page can lead to unpredictable results.
Note: A modal form appears either on top of another form or a list. To submit a modal form, your test must have already opened it on top of a form or a list.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Form UI</td>
<td>Option to select an available UI.</td>
</tr>
</tbody>
</table>

**Note:** Workspaces are not supported in Internet Explorer (IE). See [KB0683275](#) for more details.

<table>
<thead>
<tr>
<th>Assert type</th>
<th>The effect on the tests on submitting a form.</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Form submitted to server</td>
<td>The step passes only if form is submitted.</td>
</tr>
<tr>
<td>• Form submission canceled in browser</td>
<td>The step passes only if form submission is canceled.</td>
</tr>
<tr>
<td>• None</td>
<td>Make no assertion on submitting the form. The step always passes when using this assert type regardless of the result of submitting the form.</td>
</tr>
</tbody>
</table>

Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>The table for the submitted record.</td>
</tr>
<tr>
<td>record</td>
<td>The sys_id of the submitted record.</td>
</tr>
</tbody>
</table>

Service Catalog category

Validate single catalog item transactions as well as requester and fulfiller flows in Service Catalog.

Activation of the Automated Test Framework for Service Catalog

These test steps require activation of the The Automated Test Framework Service Catalog (com.glide.automated_testing_impl.service_catalog) plugin, which is active by default on new instances. Administrators may need to activate the plugin on instances upgraded from earlier versions.
Support for parametrized tests

Service Catalog step configurations support parametrized tests. For more information on parametrized tests, refer to Parameterized tests.

Variable editor support

After opening a record that supports variable editor (requested item, catalog task, or incident), you can add step configurations to set variable values, or validate variable states or values. Use the step configurations in the following order to support variable editor.

1. Step configurations to order a catalog item or record producer in the Service Catalog category.
2. Open an Existing Record step configuration in the Form category.
3. Set Variable Values, Validate Variable Values, or Variable State Validation step configuration in the Service Catalog category.

Note: Custom variables and custom variable with labels are not supported for Set Variable Values, Validate Variable Values, and Variable State Validation step configurations.

Open a Catalog Item

Open a catalog item.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Catalog Item</td>
<td>Catalog item that you want to open.</td>
</tr>
<tr>
<td></td>
<td>✋ Note: You should have access to this catalog item.</td>
</tr>
</tbody>
</table>

Open a Record Producer

Open a record producer.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Record Producer</td>
<td>Record producer that you want to open.</td>
</tr>
<tr>
<td></td>
<td>✋ Note: You should have access to the record producer.</td>
</tr>
</tbody>
</table>

Set Variable Values

Set variable values for the current catalog item or the record producer. For a catalog item, use this step after opening a catalog item page using the Open a Catalog Item step, and before using the Order Catalog Item step. For a record
producer, use this step after opening a record producer page using the Open a Record Producer step, and before using the Submit Record Producer step.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td></td>
<td>Note: Use the condition builder to set the field value. The condition builder displays an appropriate control for the field data type. For example, a reference field displays a Lookup record control.</td>
</tr>
<tr>
<td>Item</td>
<td>Catalog item or record producer for which you want to set variable values.</td>
</tr>
<tr>
<td>Variable Values</td>
<td>List of variables and the values that you want to set for them.</td>
</tr>
<tr>
<td></td>
<td>Note: You can set the value for multiple variables.</td>
</tr>
</tbody>
</table>

**Set Catalog Item Quantity**

Set the quantity for the current catalog item.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Item</td>
<td>Catalog item whose quantity you want to set.</td>
</tr>
<tr>
<td>Quantity</td>
<td>Quantity you want to set for the catalog item.</td>
</tr>
</tbody>
</table>

**Validate Variable Values**

Validate variable values on the current catalog item or record producer. For a catalog item, use this step after opening a catalog item page using the **Open a Catalog Item** step, and before using the **Order Catalog Item** step. For a record producer, use this step after opening a record producer page using the **Open a Record Producer** step, and before using the **Submit Record Producer** step.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
</tbody>
</table>
**Inputs (continued)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Use the condition builder to set the field value. The condition builder displays an appropriate control for the field data type. For example, a reference field displays a <strong>Lookup record</strong> control.</td>
</tr>
<tr>
<td>Item</td>
<td>Catalog item or record producer whose variables should be validated.</td>
</tr>
<tr>
<td>Catalog Conditions</td>
<td>Conditions for variable validation. If the conditions are met, the test passes.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The label of a variable associated with a variable set reflects the variable set name. The format is <strong>variable_set_name » variable_name</strong>.</td>
</tr>
</tbody>
</table>

**Variable State Validation**

Validate the state of variables. Possible variable states are mandatory, not mandatory, read only, not read only, visible, and not visible.

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Catalog item</td>
<td>Catalog item for which you want to validate the state.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> You should have access to this catalog item.</td>
</tr>
<tr>
<td>Visible</td>
<td>List of the catalog item variables that must be visible for the step to pass.</td>
</tr>
<tr>
<td>Not visible</td>
<td>List of the catalog item variables that must be hidden for the step to pass.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read only</td>
<td>List of the catalog item variables that must be read-only for the step to pass.</td>
</tr>
<tr>
<td>Not read only</td>
<td>List of the catalog item variables that must not be read-only for the step to pass.</td>
</tr>
<tr>
<td>Mandatory</td>
<td>List of the catalog item variables that must be mandatory for the step to pass.</td>
</tr>
<tr>
<td>Not mandatory</td>
<td>List of the catalog item variables that must not be mandatory for the step to pass.</td>
</tr>
</tbody>
</table>

**Validate Price and Recurring Price**

Validate price and recurring price of a catalog item. Use this step after opening a catalog item page using the **Open a Catalog Item** step, and before using the **Order Catalog Item** step.

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Price</td>
<td>Price of the catalog item for the step to pass.</td>
</tr>
<tr>
<td>Recurring price</td>
<td>Recurring price of the catalog item for the step to pass.</td>
</tr>
<tr>
<td>Recurring price frequency</td>
<td>Recurring price frequency of the catalog item for the step to pass.</td>
</tr>
</tbody>
</table>

Add Item to Shopping Cart

Add a catalog item to the shopping cart. Use this step after opening a catalog item page using the Open a Catalog Item step. After this step, you cannot use any other steps on the catalog item.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Assert Type</td>
<td>Criteria for the test to pass.</td>
</tr>
</tbody>
</table>

**Successfully added item to Shopping Cart**

Test passes only if the catalog item is successfully added to the shopping cart.

**Cannot add item to Shopping Cart**

Test passes only if the catalog item cannot be added to the shopping cart.

Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cart_item_id</td>
<td>The sys_id of the added catalog item.</td>
</tr>
</tbody>
</table>

**Order Catalog Item**

Clicks **Order Now** for a catalog item. Use this step after opening a catalog item page using the Open a Catalog Item step. After this step, you cannot use any other steps on the catalog item. If the two-step checkout is false, a request is generated for the catalog item. If the two-step checkout is true, you are redirected to the cart preview page.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of</td>
</tr>
</tbody>
</table>
## Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>the timeout.</td>
<td>If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Assert type</td>
<td>Criteria for the test to pass.</td>
</tr>
<tr>
<td><strong>Successfully ordered Catalog Item</strong></td>
<td>Test passes only if the catalog item is successfully ordered.</td>
</tr>
<tr>
<td><strong>Cannot order Catalog Item</strong></td>
<td>Test passes only if the catalog item cannot be ordered.</td>
</tr>
</tbody>
</table>

## Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_id</td>
<td>The sys_id of the created catalog request.</td>
</tr>
<tr>
<td>cart</td>
<td>The sys_id of the cart holding the catalog item.</td>
</tr>
</tbody>
</table>

## Submit Record Producer

Submits the current record producer. Use this step after opening the record producer page using the Open a Record producer step. After this step, you cannot use any other steps on the catalog item.
### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Assert Type</td>
<td>Criteria for the test to pass.</td>
</tr>
<tr>
<td><strong>Successfully submitted Record Producer</strong></td>
<td>Test passes only if the record producer is submitted successfully.</td>
</tr>
<tr>
<td><strong>Cannot submit Record Producer</strong></td>
<td>Test passes only if the record producer cannot be submitted.</td>
</tr>
</tbody>
</table>

### Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record_id</td>
<td>The sys_id of target record of the record producer.</td>
</tr>
</tbody>
</table>

### Forms in Service Portal category

Validate the functionality of fields and UI actions in Service Portal form widgets.
# Service Portal dependency

Creating automated Service Portal steps requires knowledge of the ServiceNow data model and Service Portal form and data structures.

## Open a Form (SP)

Opens a form in a portal.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Portal</td>
<td>Portal in which the defined form opens. Service Portal is the default.</td>
</tr>
<tr>
<td>Page</td>
<td>Page to open in the defined portal. The form page is the default.</td>
</tr>
<tr>
<td>Table</td>
<td>Table containing the form to open.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the record to open. Default is -1, which opens a new record.</td>
</tr>
</tbody>
</table>
### Field Values (SP)

Sets the values of fields in a form. To use this step, you must have already opened a form using the **Open a Form (SP)** test step.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
</tbody>
</table>

**Note:** Use the condition builder to set the field value. The condition builder displays an appropriate control for the field data type. For example, a reference field displays a **Lookup record** control.

<table>
<thead>
<tr>
<th>Table</th>
<th>The table for the form on which to set field values. The value should be the table in the <strong>Open a Form (SP)</strong> step.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field values</td>
<td>Fields for which to set values and the values to be set for those fields.</td>
</tr>
</tbody>
</table>
Field Values Validation (SP)

Validates field values on the current form based on defined conditions. To use this step, you must have already opened a form using the **Open a Form (SP)** test step.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Use the condition builder to set the field value. The condition builder displays an appropriate control for the field data type. For example, a reference field displays a <strong>Lookup record</strong> control.</td>
</tr>
<tr>
<td>Table</td>
<td>The table that contains the form on which to validate fields. The value should be the table in the <strong>Open a Form (SP)</strong> step.</td>
</tr>
<tr>
<td>Conditions</td>
<td>Conditions used to validate one or more fields on the form. If the condition evaluates to true, the step passes.</td>
</tr>
</tbody>
</table>

Field State Validation (SP)

Validates field states on a form in Service Portal.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Table</td>
<td>The table for the form on which to validate field states. The value should be the table in the <strong>Open a Form (SP)</strong> step.</td>
</tr>
<tr>
<td>Visible</td>
<td>Validates whether the fields on this form are visible. The test fails if the fields are not visible.</td>
</tr>
<tr>
<td>Not visible</td>
<td>Validates whether the fields on this form are visible. The test fails if the fields are visible.</td>
</tr>
<tr>
<td>Read only</td>
<td>Validates whether the fields on this form are read only. The test fails if the fields are not read only.</td>
</tr>
<tr>
<td>Not read only</td>
<td>Validates whether the fields on this form are read only. The test fails if the fields are read only.</td>
</tr>
<tr>
<td>Mandatory</td>
<td>Validates whether the fields on this form are mandatory. The test fails if the fields are not mandatory.</td>
</tr>
<tr>
<td>Not Mandatory</td>
<td>Validates whether the fields on this form are mandatory. The test fails if the fields are mandatory.</td>
</tr>
</tbody>
</table>

**Add Attachments to Form (SP)**

Test the functionality of attaching a file to a Service Portal form widget.
To use this step, you must have already opened a form using the **Open a Form (SP)** test step or **Open Service Portal Page steps**. This step can’t be used after a **Submit Form** step or **Click a UI Action** step has been used.

### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Upload Attachments</td>
<td>Button to attach one or more files to the form.</td>
</tr>
</tbody>
</table>

### UI Action Visibility Validation (SP)

Determines whether a UI action on the current Service Portal form is visible. To use this step, you must have already opened a form using the **Open a Form (SP)** test step.

Service Portal only supports server UI actions. The setRedirectURL() method and client UI actions are not supported. UI action visibility can vary depending on the currently logged in or impersonated user.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>in. You can change this default order by editing the Execution order values.</td>
<td></td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Table</td>
<td>The table for the form on which to check UI action visibility. The value should be the table in the <strong>Open a Form (SP)</strong> step.</td>
</tr>
<tr>
<td>Visible</td>
<td>Fields from the UI Actions table to be checked for visibility. Includes only form-based UI actions. The test fails if the UI action is not visible on the form for the currently logged in user.</td>
</tr>
<tr>
<td><em>Note:</em> If the list contains UI actions with the same name, check the form to determine the sys_id of the element. Then filter by sys_id in the UI Action table to select the correct element in the step.</td>
<td></td>
</tr>
<tr>
<td>Not visible</td>
<td>Fields from the UI Actions table to be checked for visibility. Includes only form-based UI actions. The test fails if the UI action is visible on the form for the currently logged in user.</td>
</tr>
<tr>
<td><em>Note:</em> If the list contains UI actions with the same name, check the form to determine the sys_id of the element. Then filter by sys_id in the UI Action table to select the correct element in the step.</td>
<td></td>
</tr>
</tbody>
</table>

**Click UI Action (SP)**

Selects a UI action on the current Service Portal form and outputs the table and sys_id of the record on which the action was selected.
To use this step, you must have already opened a form using the **Open a Form (SP)** test step. After using this step, you cannot use any other form steps.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Table</td>
<td>The table for the form on which to click a UI action. The value should be the table in the <strong>Open a Form (SP)</strong> step.</td>
</tr>
<tr>
<td>UI action</td>
<td>The UI action to click, selected from the UI Actions table. Includes only form-based UI actions.</td>
</tr>
<tr>
<td>Assert type</td>
<td>Specifies where to check for form submission after clicking the UI action:</td>
</tr>
</tbody>
</table>
### Field Description

- **--None--**: Selects the UI action without validating mandatory or other fields.

- **Form submission canceled in browser**: Checks whether the form was canceled in the browser and did not reach the server due to validation or other issues.

- **Form submitted to server**: Checks whether the form was submitted to the server.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record_id</td>
<td>Sys_id of the record on which the action was clicked.</td>
</tr>
<tr>
<td>table</td>
<td>Table with the clicked UI action.</td>
</tr>
</tbody>
</table>

### Submit a Form (SP)

Submits the current form in a Service Portal page and outputs the table and sys_id of the submitted record.

To use this step, you must have already opened a form using the **Open a Form (SP)** test step. After using this step, the page closes. You can't use any other steps on the current page.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Assert type</td>
<td>Specifies where to check for form submission:</td>
</tr>
<tr>
<td></td>
<td>- <strong>--None--</strong>: Submits the form without validating mandatory or other fields.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Form submitted to server</strong>: Checks if the form was submitted to the server.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Form submission canceled in browser</strong>: Checks whether the form was canceled in the browser and did not reach the server due to validation or other issues.</td>
</tr>
</tbody>
</table>

**Outputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record_id</td>
<td>Sys_id of the submitted record.</td>
</tr>
<tr>
<td>table</td>
<td>Table for the submitted record.</td>
</tr>
</tbody>
</table>

**List and Related List**

Validate the functionality and visibility of records and UI actions in lists and related lists.

**Validate Related List Visibility**

Validate the visibility of the selected related lists on a form.

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps</td>
</tr>
</tbody>
</table>
### Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Table</td>
<td>Name of the table of the parent form where related list visibility is validated.</td>
</tr>
<tr>
<td>Visible</td>
<td>List of related lists to assert as visible.</td>
</tr>
<tr>
<td>Not visible</td>
<td>List of related lists to assert as not visible.</td>
</tr>
</tbody>
</table>

### Apply Filter to List

Apply a filter to a list to find the required record.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>List type</td>
<td>Option to select the type of list.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>List</td>
<td></td>
</tr>
<tr>
<td>Related list</td>
<td></td>
</tr>
</tbody>
</table>

Assert Type
Specifies the assertion on executing the test step:
- **At least one matching record found**: The step succeeds if there is at least one matching record.
- **No matching record found**: The step fails if there is any matching record.
- **One matching record found**: The step succeeds if there is exactly one matching record.

Table
If the **List type** is **List**, this field is the table of the opened list.
If the **List type** is **Related list**, this field states the name of the table of the parent form where the list belongs.

Related List
The related list to apply a filter to.

Note: This field is available when **Related list** is selected from **List type**.

List filter
Filter conditions to apply to the list.

Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>first_record</td>
<td>The first record found in the list after applying the indicated filter.</td>
</tr>
</tbody>
</table>

Validate Record Present in List

Validate the presence of a record in a list. A valid form must be open and the list containing the record must be visible to proceed.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Inputs</td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>Execution</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
</tbody>
</table>

### Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Test</td>
<td>Name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>List type</td>
<td>Option to select the type of list.</td>
</tr>
<tr>
<td></td>
<td>- List</td>
</tr>
<tr>
<td></td>
<td>- Related list</td>
</tr>
<tr>
<td>Assert type</td>
<td>Specifies the assertion on executing the test step:</td>
</tr>
<tr>
<td></td>
<td>- Record is present in the list</td>
</tr>
<tr>
<td></td>
<td>- Record is not present in the list</td>
</tr>
<tr>
<td>Table</td>
<td>Name of the parent table where the list belongs.</td>
</tr>
<tr>
<td>Related list</td>
<td>The related list which has the record to validate.</td>
</tr>
<tr>
<td>Record</td>
<td>The record whose presence is validated.</td>
</tr>
</tbody>
</table>

**Note:** This field is available when **Related list** is selected from **List type**.

### Open a Record in List

Open a specific record in a list.
**Inputs (continued)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Test</td>
<td>Name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>List type</td>
<td>Option to select the type of list.</td>
</tr>
<tr>
<td></td>
<td>• List</td>
</tr>
<tr>
<td></td>
<td>• <strong>Related list</strong></td>
</tr>
<tr>
<td>Table</td>
<td>Name of the parent table where the list belongs.</td>
</tr>
<tr>
<td>Related List</td>
<td>The related list which has the record to open.</td>
</tr>
<tr>
<td>Record</td>
<td>The record to be opened.</td>
</tr>
</tbody>
</table>

**Note:** This field is available when **Related list** is selected from **List type**.

**Validate List UI Action Visibility**

Validate that a UI action is visible in a list. If you’re impersonating a user, the visibility of a UI action can change depending on the user being impersonated.

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Execution order | Integer specifying the order in which the test executes this step.  
As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them. You can change this default order by editing the **Execution order** values. |
| Application   | Application scope in which the system runs this step.                                                                                       |
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Test</td>
<td>Name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>List type</td>
<td>Option to select the type of list.</td>
</tr>
<tr>
<td></td>
<td>• List</td>
</tr>
<tr>
<td></td>
<td>• Related list</td>
</tr>
<tr>
<td>Table</td>
<td>Name of the table of the parent form.</td>
</tr>
<tr>
<td>Related List</td>
<td>The related list which has the UI actions.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This field is available when Related list is selected from List type.</td>
</tr>
<tr>
<td>Visible</td>
<td>List of UI actions to assert as visible.</td>
</tr>
<tr>
<td>Not visible</td>
<td>List of UI actions to assert as not visible.</td>
</tr>
</tbody>
</table>

Click a List UI Action

Select a list UI action in a list on a form.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Test</td>
<td>Name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>List type</td>
<td>Option to select the type of list.</td>
</tr>
<tr>
<td></td>
<td>• List</td>
</tr>
<tr>
<td></td>
<td>• Related list</td>
</tr>
<tr>
<td>Table</td>
<td>Name of the table where the UI action belongs.</td>
</tr>
<tr>
<td>Related list</td>
<td>The related list which has the UI actions.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This field is available when Related list is selected from List type.</td>
</tr>
<tr>
<td>List action</td>
<td>The list UI action to be clicked.</td>
</tr>
<tr>
<td>Action type</td>
<td>The type of UI action to be clicked.</td>
</tr>
<tr>
<td>Apply to</td>
<td>The record on which the UI action is applied.</td>
</tr>
<tr>
<td>Assert type</td>
<td>The assertion made by the step when executed.</td>
</tr>
<tr>
<td></td>
<td>• None: Click a UI action and continue without making an assertion.</td>
</tr>
<tr>
<td></td>
<td>• Page reloaded or redirected: Click a UI action, and assert that the page was reloaded or redirected. This assert type doesn't assert form submission.</td>
</tr>
<tr>
<td>Record</td>
<td>The record to which the UI action is applied.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This field appears when you choose Single record in the Apply to field.</td>
</tr>
</tbody>
</table>

**REST category**
Verify the functionality of REST calls.

**Send REST Request - Inbound - REST API Explorer**
This test step begins with the REST API Explorer. Use the REST API Explorer to create and specify the HTTP method, path, query parameters, request headers, and body of a REST request, and then send the REST request to the current instance.
When you have tested the request, the **Create Automated Test Step** button appears. Click **Create Automated Test Step** to create the test step. This button does not appear until after the request has been sent. You cannot create a test step when the request payload is larger than the maximum request payload size property.

This test step creates the same test record as the **Send REST Request - Inbound** test step. After the test step is created, you cannot go back and use the REST API Explorer to update the test. All changes must be made on the **Send REST Request - Inbound** test step form.

No HTTP response validation is performed as part of this step. The step fails if the response payload size is too big, the request parameters are invalid, or the request could not be sent. Use the assert steps to validate the response.

You cannot use this step to send a request to another instance or third party/remote address.

These inputs are for the REST API Explorer. The fields you see depend upon the API selected. For more information on using REST APIs on your instance, see **REST API**.

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace</td>
<td>Namespace for the request. Select from the list.</td>
</tr>
<tr>
<td>API Name</td>
<td>API to be used. Select from the list.</td>
</tr>
<tr>
<td>API Version</td>
<td>API version to be used. Select the version available on your instance from the list.</td>
</tr>
<tr>
<td>Path parameters</td>
<td>The part of the path after the API name. Path parameters are generally name-value pairs where the allowable values are in a list.</td>
</tr>
<tr>
<td>Query parameters</td>
<td>Name-value pairs of query parameters added to the URI after the path.</td>
</tr>
<tr>
<td>Request headers</td>
<td>Name-value pairs contained in the request header. The authentication</td>
</tr>
</tbody>
</table>
**Send REST Request - Inbound**

Create a test step to send a REST request to the current instance. Specify the HTTP method, path, query parameters, request headers, and body of a REST request.

No HTTP response validation is performed as part of this step. The step fails if the response payload size is too big, the request parameters are invalid, or the request could not be sent. Use the assert steps to validate the response.

You cannot use this step to send a request to another instance or third party/remote address.

For more information on using REST APIs, see the [REST API](#).

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
</tbody>
</table>
## Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
</tbody>
</table>
| Basic authentication| If you are using a public API, you do not need to provide authentication credentials.  
If you are using a non-public API, you must provide the basic authentication information when doing the test. If you leave this field blank, you may receive unexpected results. For example, the response-status code might be 401 Unauthorized instead of 200 OK.  
You must create or select a basic authentication profile to assign to test steps to avoid authentication issues when running the test.  
Users must have the web_service_admin role to fully leverage the REST test steps. Without this role, users cannot view or set basic authentication profiles needed for endpoints that require authentication. Without this role, users can still leverage the REST test steps for public |
## Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>APIs and unauthenticated access to non-public APIs. For full REST test step access, the atf_ws_designer role is available and contains the web_service_admin role along with the atf_test_designer role.</td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>The HTTP method to be used:</td>
</tr>
<tr>
<td></td>
<td>• GET</td>
</tr>
<tr>
<td></td>
<td>• POST</td>
</tr>
<tr>
<td></td>
<td>• PUT</td>
</tr>
<tr>
<td></td>
<td>• DELETE</td>
</tr>
<tr>
<td></td>
<td>• PATCH</td>
</tr>
<tr>
<td>Path</td>
<td>The path to be used. This field accepts only the portion of the URI after the instance name. If you use https:// &lt;instance name&gt;, you get an error.</td>
</tr>
<tr>
<td>Query Parameters</td>
<td>Query parameter names and values. Do not encode the parameter names or values.</td>
</tr>
<tr>
<td>Headers</td>
<td>Header names and values. Do not encode the header names or values.</td>
</tr>
<tr>
<td>Body</td>
<td>The body of the request.</td>
</tr>
</tbody>
</table>

### Assert Status Code

Assert that the HTTP response status code has the specified relationship to the specified value. You specify a numeric value of the status code and the relationship.

Assert steps must immediately follow a **Send REST Request - Inbound** step. You can have multiple REST assert steps following a **Send REST Request - Inbound** step, but the assert steps cannot be separated from the **Send REST Request - Inbound** step by steps from other test categories.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Operation</td>
<td>Comparison to be performed between values:</td>
</tr>
<tr>
<td></td>
<td>• is</td>
</tr>
<tr>
<td></td>
<td>• is not</td>
</tr>
<tr>
<td></td>
<td>• less than</td>
</tr>
<tr>
<td></td>
<td>• greater than</td>
</tr>
<tr>
<td></td>
<td>• less than or is</td>
</tr>
<tr>
<td></td>
<td>• greater than or is</td>
</tr>
<tr>
<td>Status code</td>
<td>Status code to be tested against the response code.</td>
</tr>
</tbody>
</table>
Assert Status Code Name

Assert that the HTTP response status code name has the specified relationship to the specified value. You specify a value of the status code name, and the relationship.

Assert steps must immediately follow a **Send REST Request - Inbound** step. You can have multiple REST assert steps following a **Send REST Request - Inbound** step, but the assert steps cannot be separated from the **Send REST Request - Inbound** step by steps from other test categories.

## Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Operation</td>
<td>Comparison to be performed between values:</td>
</tr>
</tbody>
</table>
### Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• contains</td>
<td></td>
</tr>
<tr>
<td>• does not contain</td>
<td></td>
</tr>
<tr>
<td>• is</td>
<td></td>
</tr>
<tr>
<td>• is not</td>
<td></td>
</tr>
<tr>
<td>Status code name</td>
<td>Status code name to be tested.</td>
</tr>
</tbody>
</table>

### Assert Response Time

Assert that the HTTP response time has the specified relationship to the specified value. You specify a value of the response time and the relationship.

Assert steps must immediately follow a **Send REST Request - Inbound** step. You can have multiple REST assert steps following a **Send REST Request - Inbound** step, but the assert steps cannot be separated from the **Send REST Request - Inbound** step by steps from other test categories.

### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
</tbody>
</table>
### Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Operation</td>
<td>Comparison to be performed between Response time entered and actual response time:</td>
</tr>
<tr>
<td></td>
<td>• less than</td>
</tr>
<tr>
<td></td>
<td>• greater than</td>
</tr>
<tr>
<td>Response time (ms)</td>
<td>Time in milliseconds to be compared to the actual response time.</td>
</tr>
</tbody>
</table>

### Assert Response Header

Assert the HTTP response header exists, or the header has the specified relationship to the specified value.

Assert steps must immediately follow a **Send REST Request - Inbound** step. You can have multiple REST assert steps following a **Send REST Request - Inbound** step, but the assert steps cannot be separated from the **Send REST Request - Inbound** step by steps from other test categories.
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>change this default order by editing the <strong>Execution order</strong> values.</td>
<td></td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Header</td>
<td>Header name.</td>
</tr>
<tr>
<td>Operation</td>
<td>Comparison to be performed between values:</td>
</tr>
<tr>
<td></td>
<td>• contains</td>
</tr>
<tr>
<td></td>
<td>• does not contain</td>
</tr>
<tr>
<td></td>
<td>• is</td>
</tr>
<tr>
<td></td>
<td>• is not</td>
</tr>
<tr>
<td></td>
<td>• is not empty</td>
</tr>
<tr>
<td>Value</td>
<td>Element value to be used in the test. Not shown if the <strong>Operation</strong> is <strong>is not empty</strong>.</td>
</tr>
</tbody>
</table>

**Assert Response JSON Payload is Valid**

Assert that the response payload is in valid JSON format.

Assert steps must immediately follow a **Send REST Request - Inbound** step. You can have multiple REST assert steps following a **Send REST Request - Inbound**
step, but the assert steps cannot be separated from the **Send REST Request - Inbound** step by steps from other test categories.

### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
</tbody>
</table>

**Assert Response XML Payload is Well-Formed**

Assert that the response payload is well-formed XML.

Assert steps must immediately follow a **Send REST Request - Inbound** step. You can have multiple REST assert steps following a **Send REST Request - Inbound** step, but the assert steps cannot be separated from the **Send REST Request - Inbound** step by steps from other test categories.
## Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
</tbody>
</table>

### Assert XML Payload Element

Assert the XML response payload element exists, or has the specified relationship to the specified value.

Assert steps must immediately follow a **Send REST Request - Inbound** step. You can have multiple REST assert steps following a **Send REST Request - Inbound** step, but the assert steps cannot be separated from the **Send REST Request - Inbound** step by steps from other test categories.
## Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Element path</td>
<td>XML path to the element to be evaluated. For example, /result/short_description for</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;number&quot;: &quot;INC0020001&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;short_description&quot;: &quot;test&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>Operation</td>
<td>Comparison to be performed between values:</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• contains</td>
</tr>
<tr>
<td></td>
<td>• does not contain</td>
</tr>
<tr>
<td></td>
<td>• is</td>
</tr>
<tr>
<td></td>
<td>• is not</td>
</tr>
<tr>
<td></td>
<td>• is not empty</td>
</tr>
</tbody>
</table>

Value

Element value to be used in the test. Not shown if the Operation is is not empty.

Assert JSON Response Payload Element

Assert the JSON response payload element exists, or has the specified relationship to the specified value.

Assert steps must immediately follow a Send REST Request - Inbound step. You can have multiple REST assert steps following a Send REST Request - Inbound step, but the assert steps cannot be separated from the Send REST Request - Inbound step by steps from other test categories.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
</tbody>
</table>
| Element path| SNC path to the element to be evaluated. For example, /result/short_description for 

```json
{
  "result": {
    "number": "INC0020001",
    "short_description": "test"
  }
}
```

See Importing JSON files for more information. |
| Operation   | Comparison to be performed between values: |
|             | • contains |
|             | • does not contain |
|             | • is |
|             | • is not |
|             | • is not empty |
| Value       | Element value to be used in the test. Not shown if the Operation is is not empty. |
Assert Response Payload

Assert the HTTP response payload has the specified relationship to the specified value. You specify the value and the relationship.

Assert steps must immediately follow a Send REST Request - Inbound step. You can have multiple REST assert steps following a Send REST Request - Inbound step, but the assert steps cannot be separated from the Send REST Request - Inbound step by steps from other test categories.

ℹ️  Note: The entire payload is used to look for a match. A large payload can affect performance.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Operation</td>
<td>Comparison to be performed between values:</td>
</tr>
</tbody>
</table>
Inputs (continued)
Field

Description

• contains
• does not contain
• is
• is not
• is not empty
Response body

The value of the response body to be
used in the test.
Must contain the name and value
to be compared as it appears in the
response payload. Must not contain
any curly braces.
Not shown if the Operation is is not
empty.

To check the short description in the response payload
{"result":{"number":"INC0010040","short_description":"Test ATF Incident"}}

the Response body should contain
"short_description":"Test ATF Incident"

These formats are incorrect and the step fails.
• {"short_description":"Test ATF Incident"}
• "{"short_description":"Test ATF Incident"}"
• short_description: Test ATF Incident
• short_description:"Test ATF Incident"
Email category
Use Automated Test Framework (ATF) to test email notifications, outbound email
flows, and inbound email responses.
Validate Outbound Email
Verify that a certain outbound email exists by searching for it in the Email
[sys_email] table.

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.

872


Use this step to test any email-generating script that isn’t captured in the Validate Outbound Email Generated by Flow or Validate Outbound Email Generated by Notification test steps.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Test</td>
<td>Name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Conditions</td>
<td>Conditions used to filter outbound emails.</td>
</tr>
<tr>
<td>Table</td>
<td>Table to be queried.</td>
</tr>
</tbody>
</table>

Note: During testing, this step may take longer than expected to execute. The step times out after two minutes by default.

Validate Outbound Email Generated by Flow

Verify that a certain outbound email exists by searching for it in the Email table. Use this step to test that a flow is triggered as expected.

Note: This step searches only for emails that are created by a flow Send Email action. To find an email that was sent after a notification was triggered in a flow, use the Validate Outbound Email Generated by Notification step instead. To find an email that was created in any other flow action, such as a custom scripted action, use the Validate Outbound Email step instead.
Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Test</td>
<td>Name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Source Flow</td>
<td>Flow that the outbound email was created from.</td>
</tr>
<tr>
<td>Conditions</td>
<td>Conditions used to filter outbound emails. For example, to find an email that was sent to <a href="mailto:user@example.com">user@example.com</a>, set the condition to [Recipients] [is] [<a href="mailto:user@example.com">user@example.com</a>].</td>
</tr>
<tr>
<td>Table</td>
<td>Table to be queried. By default, this step queries only the Email [sys_email] table.</td>
</tr>
</tbody>
</table>

⚠ **Note:** During testing, this step may take longer than expected to execute. The step times out after two minutes by default.

**Validate Outbound Email Generated by Notification**

Verify that a certain outbound email exists by searching for it in the Email [sys_email] table. Use this step to test that an email notification is triggered as expected.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps</td>
</tr>
</tbody>
</table>
### Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Test</td>
<td>Name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Source Notification</td>
<td>Notification that the outbound email was created from. This field references the Notification [sysevent_email_action] table.</td>
</tr>
<tr>
<td>Conditions</td>
<td>Conditions used to filter outbound emails. For example, to find an email that was sent to <a href="mailto:user@example.com">user@example.com</a>, set the condition to [Recipients] [is] [<a href="mailto:user@example.com">user@example.com</a>].</td>
</tr>
<tr>
<td>Table</td>
<td>Table to be queried. By default, this step queries only the Email [sys_email] table.</td>
</tr>
</tbody>
</table>

**Note:** During testing, this step may take longer than expected to execute. The step times out after two minutes by default.

### Generate Inbound Email

Create an incoming email record to test an inbound email flow or an inbound email action.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
<td>Name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>From</td>
<td>Address of the email sender. You can manually enter an email address or reference an email address from a user record. If you manually enter an email address, the system matches the address to an existing user record during testing.</td>
</tr>
<tr>
<td>To</td>
<td>Address of the email recipient. You can manually enter an email address or reference an email address from a user record. If you manually enter an email address, the system matches the address to an existing user record during testing.</td>
</tr>
<tr>
<td>Subject</td>
<td>Subject of the email. You can enter text or reference a string output from a previous test step.</td>
</tr>
<tr>
<td>Body</td>
<td>Content of the message body. You can enter text or reference a string output from a previous test step.</td>
</tr>
</tbody>
</table>

Output

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>output_email_record</td>
<td>Record in the Email [sys_email] table.</td>
</tr>
<tr>
<td></td>
<td>The <strong>Receive type</strong> is <strong>New</strong> and the <strong>Type</strong> is <strong>send-ready</strong>. For more information on what these fields mean, see <em>System email log and mailboxes</em>.</td>
</tr>
</tbody>
</table>

Generate Inbound Reply email

Create a reply email record to test how the system handles a user response to an email notification.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Test</td>
<td>Name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Target Table</td>
<td>Table of the target record.</td>
</tr>
<tr>
<td>Target Record</td>
<td>Record that the reply email updates. Selecting a target record also applies a watermark to the reply email.</td>
</tr>
<tr>
<td>From</td>
<td>Address of the email sender. You can manually enter an email address or reference an email address from a user record. If you manually enter an email address, the system matches the address to an existing user record during testing.</td>
</tr>
<tr>
<td>To</td>
<td>Address of the email recipient. You can manually enter an email address or reference an email address from a user record. If you manually enter an email address, the system matches the address to an existing user record during testing.</td>
</tr>
<tr>
<td>Subject</td>
<td>Subject of the email. You can enter text or reference a string output from a previous test step.</td>
</tr>
<tr>
<td>Body</td>
<td>Content of the message body. You can enter text or reference a string output from a previous test step.</td>
</tr>
</tbody>
</table>

Output

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| output_reply_email_record | Record in the Email [sys_email] table.  
The **Receive type** is **Reply** and the **Type** is **send-ready**. For more information on what these fields mean, see **System email log and mailboxes**. |
Generate Random String

Generate a string of random alphanumeric characters that you can use as test data for another step.

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Test</td>
<td>Name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Length</td>
<td>Number of characters in the generated string. The maximum length is 10,000 characters. If you leave the field blank, the string length is 10 characters by default.</td>
</tr>
</tbody>
</table>

**Output**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>random_string</td>
<td>String of random alphanumeric characters.</td>
</tr>
</tbody>
</table>

**Server category**

Perform server-side operations. For example, query and update a record, impersonate a user, or run a server-side script.

**Create a User**

Create a user with specified roles and groups for the test. The user record gets rolled back after the test completes.
**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Test</td>
<td>Name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>First name</td>
<td>First name of the user.</td>
</tr>
<tr>
<td>Last name</td>
<td>Last name of the user.</td>
</tr>
<tr>
<td>Roles</td>
<td>Assigned roles of the user.</td>
</tr>
<tr>
<td>Groups</td>
<td>Assigned groups of the user.</td>
</tr>
<tr>
<td>Impersonate this user</td>
<td>Option to impersonate the new user.</td>
</tr>
</tbody>
</table>

**Outputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>The user ID of the user impersonated.</td>
</tr>
</tbody>
</table>

**Impersonate**

Impersonate the specified user for the test.

**Impersonate** specifies a user for executing subsequent steps in this test. It works for both server-side and browser-side steps and stays in effect until changed with another **Impersonate** step or until the test ends. The impersonation automatically ends when the test is over.
Note:

- Do not impersonate a user with the test author role. Doing so can lead to conflicts that interfere with executing the test.
- Tests which involve impersonated users which no longer exist fail.

**Tip:** It is recommended to create a new user to avoid data dependencies. See Create a User, for more information.

- Do not rely on user IDs being consistent across different instances. The system dynamically assigns users IDs, so the ID for a particular user often differs from one instance to the next.
- When exporting and importing automated tests, keep in mind that update sets do not update the user field.
- Tests can impersonate users with the snc_external role, which allows testing users who do not have login access. See Explicit Roles for requirements of the snc_external role.

### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Execution order | Integer specifying the order in which the test executes this step.  
As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values. |
| Active      | Option to activate this test step for use.                                 |
| Application | Application scope in which the system runs this step.                      |
| Test        | Read-only name of the test that you're adding the step to.                  |
| User        | The user ID for the test to impersonate.                                   |

### Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>The user ID of the user impersonated.</td>
</tr>
</tbody>
</table>
Search for a Catalog Item

Searches for a catalog item or record producer in the specified catalog and category. You can perform this step both in Platform and Service Portal.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Search in Portal only</td>
<td>Selected if the search is restricted to the Service Portal only. Otherwise, not selected.</td>
</tr>
<tr>
<td>Search term</td>
<td>Term used to search for a catalog item.</td>
</tr>
<tr>
<td>Catalog</td>
<td>Catalog in which to search for the catalog item.</td>
</tr>
<tr>
<td>Category</td>
<td>Category in which to search for the catalog item.</td>
</tr>
<tr>
<td>Assert item</td>
<td>Catalog items that should be available in the search results.</td>
</tr>
<tr>
<td>Assert Type</td>
<td>Specifies how searching the catalog item affects the test:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Assert Item present in search result:</strong> Test passes only if the assert item is present in the search result.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Assert Item not present in search result:</strong> Test passes only if the assert item is not present in the search result.</td>
</tr>
</tbody>
</table>
Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalog_item_id</td>
<td>The sys_id of the first catalog item found that the user can view.</td>
</tr>
</tbody>
</table>

Record Query

Query the database to verify that a record exists matching the conditions set in this step.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td></td>
<td>As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Enforce security</td>
<td>Selected to enforce ACLs. Otherwise, not selected.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
</tbody>
</table>
### Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong></td>
<td>Use the condition builder to set the field value. The condition builder</td>
</tr>
<tr>
<td></td>
<td>displays an appropriate control for the field data type. For example, a</td>
</tr>
<tr>
<td></td>
<td>reference field displays a <strong>Lookup record</strong> control.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Assert type</th>
<th>Specifies how querying the record affects the test:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• <strong>There is at least one record matching the query</strong>: The test fails if</td>
</tr>
<tr>
<td></td>
<td>there are no records matching the query.</td>
</tr>
<tr>
<td></td>
<td>• <strong>No records match the query</strong>: The test fails if any records match the</td>
</tr>
<tr>
<td></td>
<td>query.</td>
</tr>
</tbody>
</table>

| Table       | The table to be queried.                                                    |
| Conditions  | Conditions used to run the query.                                           |

### Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>The table queried.</td>
</tr>
<tr>
<td>first_record</td>
<td>The first record resulting</td>
</tr>
<tr>
<td></td>
<td>from the query.</td>
</tr>
</tbody>
</table>

*Note:* If you don’t update your record query test step, the original record query test step still functions the same way as before, irrespective of it being termed as *(Deprecated)*.

### Record Insert

Inserts a record into a table with the field values you specify.

### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution</td>
<td>Integer specifying the order in which the test executes this step.</td>
</tr>
<tr>
<td>order</td>
<td>As you create steps, the system automatically assigns each step an</td>
</tr>
<tr>
<td></td>
<td>incremental value. This value causes the test to execute steps in the</td>
</tr>
<tr>
<td></td>
<td>order that you created them in. You can change this default order by editing</td>
</tr>
<tr>
<td></td>
<td>the <strong>Execution order</strong> values.</td>
</tr>
</tbody>
</table>
### Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Use the condition builder to set the field value. The condition builder displays an appropriate control for the field data type. For example, a reference field displays a <em>Lookup record</em> control.</td>
</tr>
<tr>
<td>Assert type</td>
<td>Specifies how inserting the record affects the test:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Record successfully inserted:</strong> Test fails if the record was not successfully inserted.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Record was not inserted:</strong> Test fails if the record was successfully inserted.</td>
</tr>
<tr>
<td>Enforce security</td>
<td>Selected to enforce ACLs and the read-only role. Otherwise, not selected.</td>
</tr>
<tr>
<td>Table</td>
<td>The table into which the record should be inserted.</td>
</tr>
<tr>
<td>Conditions</td>
<td>Specific field values to be set when the test runs this step.</td>
</tr>
</tbody>
</table>

### Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>The table to which the new record belongs.</td>
</tr>
<tr>
<td>record_id</td>
<td>The sys_id of the new record.</td>
</tr>
</tbody>
</table>

### Record Update

Changes field values on a record on the server.
Note: To ensure that the changes were applied, follow this step with a **Record Validation** step.

### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Assert type</td>
<td>Specifies how updating the record affects the test:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Record successfully updated</strong>: Test fails if the record was not successfully updated.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Record was not updated</strong>: Test fails if the record was successfully updated.</td>
</tr>
<tr>
<td>Enforce security</td>
<td>Selected to enforce ACLs and the read-only role. Otherwise, not selected.</td>
</tr>
<tr>
<td>Table</td>
<td>The table containing the record to be updated.</td>
</tr>
<tr>
<td>Record</td>
<td>ID of the record to be updated.</td>
</tr>
<tr>
<td>Field values</td>
<td>Fields for which you want to set values and the values you want to set for those fields.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong></td>
<td>Use the condition builder to set the field value. The condition builder displays an appropriate control for the field data type. For example, a reference field displays a <strong>Lookup record</strong> control.</td>
</tr>
</tbody>
</table>

| **Note:** | Record Update step succeeds even if a field on the record is blocked by ACL. Use the **Record Validation** step after Record Update to check whether a particular field was modified, or use the **Form steps** to evaluate ACL conditions for individual fields on a record. |

**Record Delete**

Deletes a specified record in a table.

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Assert type</td>
<td>Specifies how updating the record affects the test:</td>
</tr>
</tbody>
</table>
### Record Validation

 Validates that a record meets the specified conditions on the server side.

 For the **Record Validation** step, specify the values you want to test using the standard conditions builder. You can apply several conditions to the same field. This step passes if the overall condition is satisfied and fails if it is not. If you need to test the values of individual fields independently of each other, include a separate **Record Validation** step for each value to be tested.

### Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
</tbody>
</table>
**Inputs (continued)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Use the condition builder to set the field value. The condition builder displays an appropriate control for the field data type. For example, a reference field displays a <code>Lookup record</code> control.</td>
</tr>
<tr>
<td>Assert type</td>
<td>Specifies how validating the record affects the test:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Record successfully validated:</strong> The test fails if the record does not match the conditions.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Record not found:</strong> Test fails if the record is found.</td>
</tr>
<tr>
<td>Table</td>
<td>The table that contains the field to be validated.</td>
</tr>
<tr>
<td>Record</td>
<td>The record that contains the field to be validated.</td>
</tr>
<tr>
<td>Field values</td>
<td>Specific fields to be validated when the test runs this step.</td>
</tr>
</tbody>
</table>

**Run Server Side Script**

Executes a script on the server.

**Inputs**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <code>Execution order</code> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Jasmine version</td>
<td>The version of the Jasmine testing framework to use for running the server-side script. Any new scripts you create use Jasmine version 3.1. Your existing scripts can continue to use Jasmine version 1.3, or you can upgrade them to Jasmine version 3.1.</td>
</tr>
<tr>
<td>Test script</td>
<td>The javascript for the server to execute. Supports the Jasmine testing framework.</td>
</tr>
<tr>
<td></td>
<td>Note: steps(SYS_ID) can be defined as a function to retrieve Output variable data from a step that executed earlier in the test.</td>
</tr>
</tbody>
</table>

Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record_id</td>
<td>The sys_id output by the server-side script.</td>
</tr>
<tr>
<td>table</td>
<td>The table output by the server-side script.</td>
</tr>
</tbody>
</table>

Note: If the script creates data, the system rolls back that data after all steps in the test finish.

• Test script

```javascript
// Test step 1 - add data
var now_GR = new GlideRecord('sc_task');
// this sample step's Step config has Output variables named table and record_id
outputs.table = 'sc_task';
outputs.record_id = gr.insert();
// Test step 2 - access added data and validate
// check that the record exists (or that business logic changed it)
var now_GR = new GlideRecord("sc_task");
gr.get(steps(PREVIOUS_STEP_SYS_ID).record_id);
assertEqual({name: "task gr exists", shouldbe: true, value: gr.isValidRecord()});
```
stepResult.setOutputMessage: Log a message to step results after step executes.  
Can only be called once or will overwrite previous message

• Create a suite of test scripts

```javascript
var now_GR = new GlideRecord('sc_task');
gr.setValue('short_description', 'verify task can be inserted');
var grSysId = gr.insert();
var justCreatedGR = new GlideRecord('sc_task');
if (justCreatedGR.get(grSysId)) {
  stepResult.setOutputMessage("Successfully inserted task record");
  return true; // pass the step
} else {
  stepResult.setOutputMessage("Failed to insert task record");
  return false; // fail the step
}
```

• Jasmine test

```javascript
describe('my suite of script tests', function() {
  it('should meet expectations', function() {
    expect(true).not.toBe(false);
  });
});
// make sure to uncomment jasmine.getEnv().execute(); outside the function body
```

assertEqual: A function used to compare that assertion.shouldbe == assertion.value;  
in case of failure it throws an Error and logs that the assertion by name has failed

**Note:** describe is only supported in Global scope. Use describe to create a suite of test scripts and it to define test expectations

• Logs message to test step output

```javascript
var testAssertion = {
  name: "my test assertion",
  shouldbe: "expected value"
  value: "actual value",
};
assertEqual(testAssertion); // throws Error, logs message to test step output
```

• See Step Execution Scripts for Run Server Side script example.

**Replay Request Item**

Get the item and requester from an existing request item, add the item to a new cart for that user, and place an order.
## Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Original Request Item</td>
<td>The request item to be replayed.</td>
</tr>
</tbody>
</table>

## Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>The table to which the replayed request item belongs.</td>
</tr>
<tr>
<td>request</td>
<td>The replayed request item.</td>
</tr>
</tbody>
</table>

## Related information

**Automated Test Framework use case: test a Service Catalog request**

## Log

Logs a message and stores it as a step result.

The log message can contain variables and other information pertaining to the test. The message is stored as a step result.
## Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you're adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Log</td>
<td>The message to be logged. To include the value of an output variable from a previous step, click the input value icon ( <img src="input_value_icon.png" alt="input value icon" />) and follow the procedure to Pass values from one automated test step to another.</td>
</tr>
</tbody>
</table>

### Add Attachments to Existing Record

Add one or more mandatory attachments to the specified record. Use **Upload Attachments** to select from the attachments the test step adds to the record.

## Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
</tbody>
</table>
Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
<td>Name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Name of the step.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Table</td>
<td>Valid table selection from the list.</td>
</tr>
<tr>
<td>Record</td>
<td>Pre-existing record either from before the test or inserted as a part of the test.</td>
</tr>
<tr>
<td>Upload Attachments</td>
<td>Button to add one or more mandatory attachments to the record.</td>
</tr>
</tbody>
</table>

Checkout Shopping Cart

Submits the cart and generates a request. You can perform this step both in the Now Platform and Service Portal.

Inputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the Execution order values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Number of seconds allowed before the step fails. If the validation fails, the system repeats the step until it reaches the duration of the timeout. If the validation fails after the timeout duration has passed, the step fails.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
</tbody>
</table>
## Inputs (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>Assert Type</td>
<td>Criteria for the test to pass.</td>
</tr>
<tr>
<td></td>
<td><strong>Empty cart</strong></td>
</tr>
<tr>
<td></td>
<td>Test fails if the cart is not empty.</td>
</tr>
<tr>
<td></td>
<td><strong>Successfully Checkout cart</strong></td>
</tr>
<tr>
<td></td>
<td>Test fails if the cart is not successfully checked out.</td>
</tr>
<tr>
<td>Requested For</td>
<td>User for whom the request is generated.</td>
</tr>
<tr>
<td>Delivery Address</td>
<td>Delivery address for the request.</td>
</tr>
<tr>
<td>Special Instructions</td>
<td>Special instructions for the request.</td>
</tr>
</tbody>
</table>

## Outputs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_id</td>
<td>The sys_id of the submitted request.</td>
</tr>
</tbody>
</table>

## Custom Scripted StepConfig

Provides an example of scripts for a custom step configuration.

This example checks if the user name provided starts with the letter A. This step is useful primarily to users with the [atf_test_admin] role. Users with the [atf_test_admin] role can view the example scripts by opening the record for this step in Step configurations.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution order</td>
<td>Integer specifying the order in which the test executes this step. As you create steps, the system automatically assigns each step an incremental value. This value causes the test to execute steps in the order that you created them in. You can change this default order by editing the <strong>Execution order</strong> values.</td>
</tr>
<tr>
<td>Active</td>
<td>Option to activate this test step for use.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this step.</td>
</tr>
<tr>
<td>Test</td>
<td>Read-only name of the test that you’re adding the step to.</td>
</tr>
<tr>
<td>Step config</td>
<td>Read-only name of the step.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the test step. This field value is automatically set based on the field values of the test step. This field appears after the test step is submitted.</td>
</tr>
<tr>
<td>Notes</td>
<td>Notes about the test step.</td>
</tr>
<tr>
<td>User</td>
<td>The user whose name the system checks to see if it starts with the letter <strong>A</strong>. To include the value of an output variable from a previous step, click the input value icon ( ) and follow the procedure to <strong>Pass values from one automated test step to another</strong>.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>The name of the user.</td>
</tr>
</tbody>
</table>

**Related information**

- Create custom step configuration
- Step execution scripts
- Step description generation script

**Automated Test Framework reference**

Reference information for the Automated Test Framework.
Tables excluded from rollback after running an automated test

The Automated Test Framework tracks data created by running tests and rolls back changes after testing. The system excludes certain tables from being tracked during testing.

The system excludes certain tables from being tracked or rolled back:

- The History [sys_history_line] table
- The ECC Queue table [ecc_queue].
- The Email [sys_email] Email Log [sys_email_log] tables
- The Execution Tracker [sys_execution_tracker] tables
- The Progress Worker [sys_progress_worker] table
- The Schema Change [sys_schema_change]
- The Upgrade History [sys_upgrade_history] and Upgrade History Log [sys_upgrade_history_log] tables
- The Mutex [sys_mutex] table
- The Plugin Log [sys_plugin_log] table
- The Status [sys_status] table
- The Number Counter [sys_number_counter] table
- The AMB Channel Presence [sys_amb_channel_presence] table
- Any table that extends an excluded table.
- Any table starting with the following prefixes are also excluded:
  - syslog
  - sys_amb_message
  - sys_cluster
  - cmdb_metric
  - ts_
  - v_
  - sys_delete_recovery

If your test run changes (inserts/updates/deletes) any record on these excluded tables, the system does not roll back the change after testing.
**Tests**

The Tests module opens the Test table. From here, you can add, edit, and run tests. By opening an individual test record, you can view and edit the steps comprising that test.

**Related information**

Building and running automated test suites

**Test record form**

In the Test record form, you view and edit values of fields for the test record.

**Fields**

<table>
<thead>
<tr>
<th>Field / UI Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Test name.</td>
</tr>
<tr>
<td>Description</td>
<td>Test description.</td>
</tr>
<tr>
<td>Active</td>
<td>If this test is active, true. Otherwise, false.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this test or test suite.</td>
</tr>
<tr>
<td>Test steps related list</td>
<td>The steps that this test executes.</td>
</tr>
<tr>
<td>Test results related list</td>
<td>The results from individual executions of this test.</td>
</tr>
</tbody>
</table>

**UI Actions**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update</td>
<td>Click to update the test record.</td>
</tr>
<tr>
<td>Run test</td>
<td>Click to run test.</td>
</tr>
<tr>
<td>Copy test</td>
<td>Click to copy test.</td>
</tr>
<tr>
<td>Delete</td>
<td>Click to delete this test.</td>
</tr>
</tbody>
</table>

**Suites**

The Suites module opens the Test Suites table. You can create, edit, and run test suites from this table.
Related information

Building and running automated test suites

Test suite form

The Test Suite form contains information about one test suite.

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the test suite.</td>
</tr>
<tr>
<td>Active</td>
<td>To make this test suite active, check this field.</td>
</tr>
<tr>
<td>Description</td>
<td>(Optional) Enter a description to identify the purpose of this test suite.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this test or test suite.</td>
</tr>
<tr>
<td>Parent Suite</td>
<td>To make this test suite a child of another test suite, enter the name of the parent test suite here.</td>
</tr>
<tr>
<td>Test Suite Tests</td>
<td>The tests included in this test suite.</td>
</tr>
<tr>
<td>(related list)</td>
<td></td>
</tr>
<tr>
<td>Child Test Suites</td>
<td>Any test suites that are children of this test suite.</td>
</tr>
<tr>
<td>(related list)</td>
<td></td>
</tr>
<tr>
<td>Test Suite Results</td>
<td>Results from executing this test suite.</td>
</tr>
<tr>
<td>(related list)</td>
<td></td>
</tr>
<tr>
<td>Test Suite Schedules</td>
<td>Any test suite schedules that include this test suite.</td>
</tr>
<tr>
<td>Schedules (related list)</td>
<td></td>
</tr>
</tbody>
</table>

Test results

Each time you run a test, the automated test framework creates a record of the test results. Use the Test Results module to view details about the results of individual tests and individual steps within a test. If the test generated screenshots, they appear as attachments on the test results record.

Each Test Results record displays detailed results information about one test execution, and contains links to detailed step results and test logs. Step results records display information about one step in a test result, while test log (test
results item) records contain detailed console logging and test execution information.

By default, the system deletes test and test suite results data 30 days after creation. You can modify this default retention policy as needed in the Table Cleanup module.

**Related information**

- Modify data retention policy for ATF test results
- Table Cleanup module

**Test results record**

A Test Results record contains detailed results information about one test execution. Client Error Details and Failure Details sections appear when a test fails due to error conditions, and include detailed error information you can use for troubleshooting purposes.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
<td>Name of the test.</td>
</tr>
<tr>
<td>Status</td>
<td>Result of the Test execution:</td>
</tr>
<tr>
<td></td>
<td><strong>Success</strong></td>
</tr>
<tr>
<td></td>
<td>Test passed.</td>
</tr>
<tr>
<td></td>
<td><strong>Failure</strong></td>
</tr>
<tr>
<td></td>
<td>Test failed.</td>
</tr>
<tr>
<td></td>
<td><strong>Success with warning(s)</strong></td>
</tr>
<tr>
<td></td>
<td>Test run passed but encountered client error(s) that were allowed as warnings in the Allowed Client Error table.</td>
</tr>
<tr>
<td></td>
<td><strong>Waiting</strong></td>
</tr>
<tr>
<td></td>
<td>Test waiting to start.</td>
</tr>
<tr>
<td></td>
<td><strong>Running</strong></td>
</tr>
<tr>
<td></td>
<td>Test is running.</td>
</tr>
<tr>
<td></td>
<td><strong>Skipped</strong></td>
</tr>
<tr>
<td></td>
<td>Occurs if an earlier test in the suite failed and <em>Abort on failure</em> is set to true.</td>
</tr>
<tr>
<td></td>
<td><strong>Error</strong></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>An error occurred in the test framework. For example, the test runner halted, or the server encountered an unintended scenario. The error message appears in the Summary field of the steps results record for the step that threw the error.</td>
<td></td>
</tr>
<tr>
<td>Cancelled</td>
<td>User canceled the Test.</td>
</tr>
<tr>
<td>Retain indefinitely</td>
<td>Check box to keep this record even after the specified data retention period has passed. For more information, see Table Cleanup.</td>
</tr>
<tr>
<td>Start time</td>
<td>Date and time of day at which this test started executing.</td>
</tr>
<tr>
<td>End time</td>
<td>Date and time of day at which this test finished executing.</td>
</tr>
<tr>
<td>Duration</td>
<td>Elapsed time it took to execute this test.</td>
</tr>
<tr>
<td>Previous test result</td>
<td>Results of the previous test execution, indicating if the test passed or failed. Only appears if this test result is for a re-run of a failed test and you are logged on with the admin, atf_test_admin, or atf_test_designer role. Click the information icon (i) to navigate to the original test result.</td>
</tr>
</tbody>
</table>

Note: If this test created screenshots, they appear as attachments in Manage attachments.

Client Error Details
This section only displays when a test fails due to a client error.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failing step</td>
<td>Reference to the first test step result that failed during this test. Click the information icon ( ) to view detailed step results and error information in the Step Results form.</td>
</tr>
<tr>
<td>Summary</td>
<td>The following text indicates that a client error occurred at this step, and includes the detailed error message:</td>
</tr>
<tr>
<td></td>
<td>This step failed because the client error 'DETAILED ERROR MESSAGE' was detected on the page being tested. See failing Test Logs. To ignore these errors in the next test run, use 'Add all client errors to warning/ignored list' links.</td>
</tr>
<tr>
<td></td>
<td>Test designers and developers should always investigate client errors to determine if there are issues with your business process. For more details, see Identify and resolve client errors.</td>
</tr>
<tr>
<td>First failing client error</td>
<td>Reference to the first client error that failed during the test. Click the information icon ( ) to view detailed test results and error information in the Test Logs form.</td>
</tr>
<tr>
<td></td>
<td>Note: For details on how to allow client errors as ignored or warning entries, refer to Allowed client errors</td>
</tr>
<tr>
<td>Failing step screenshot</td>
<td>Screenshot of the step that failed. Click to download the screenshot.</td>
</tr>
</tbody>
</table>

**Failure Details**

This section only displays when a test fails due to a cause other than a client error.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failing step</td>
<td>Reference to the first step result that failed during this test. Click to download the screenshot.</td>
</tr>
<tr>
<td>Summary</td>
<td>Output of the step that failed.</td>
</tr>
<tr>
<td>Failing screenshot</td>
<td>Screenshot of the step that failed.</td>
</tr>
</tbody>
</table>
**Additional information**

This section is visible for all test results.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Output generated for the test step.</td>
</tr>
<tr>
<td></td>
<td>• Test result (often Passed or Failed).</td>
</tr>
<tr>
<td></td>
<td>• For failed tests, includes the same error message detail displayed in the Summary field for the first test step that failed. See Client Error Details above.</td>
</tr>
<tr>
<td>Browsers involved</td>
<td>User agent strings returned by browsers that ran the test.</td>
</tr>
</tbody>
</table>

**Related lists**

<table>
<thead>
<tr>
<th>Related list</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step Results</td>
<td>Record for each step result and log entry for this test.</td>
</tr>
<tr>
<td>Test Log</td>
<td>Record for each test log related to this test results record.</td>
</tr>
<tr>
<td>Test Transactions</td>
<td>Record for each transaction (from the system transaction log [syslog_transaction] related to this record.</td>
</tr>
<tr>
<td></td>
<td>To view the step results associated with a transaction, click the appropriate link in the <strong>Step Results</strong> column.</td>
</tr>
<tr>
<td></td>
<td>To view the transaction logs associated with a transaction, click the appropriate link in the <strong>Transaction</strong> column.</td>
</tr>
<tr>
<td>Warnings</td>
<td>List of test logs containing client errors with a warning status. The Warnings related list only appears on test results with warnings.</td>
</tr>
</tbody>
</table>

**Related reference**

Test logs record
Step results record

**Related information**

View test results
Allowed client errors
Allow client errors from step results
Allow client errors from the test logs

Step results record

The Step Results record contains information about one step in a test result. You access specific step results from the Step Results related list in the Test Results record.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Result of the step:</td>
</tr>
<tr>
<td></td>
<td><strong>Success</strong></td>
</tr>
<tr>
<td></td>
<td>Test step passed.</td>
</tr>
<tr>
<td></td>
<td><strong>Failure</strong></td>
</tr>
<tr>
<td></td>
<td>Test step failed.</td>
</tr>
<tr>
<td></td>
<td><strong>Success with warning(s)</strong></td>
</tr>
<tr>
<td></td>
<td>Test step passed but encountered client error(s) that were allowed as warnings in the Allowed Client Error table.</td>
</tr>
<tr>
<td></td>
<td><strong>Skipped</strong></td>
</tr>
<tr>
<td></td>
<td>Occurs if an earlier test in the suite failed and Abort on failure is set to true.</td>
</tr>
<tr>
<td></td>
<td><strong>Error</strong></td>
</tr>
<tr>
<td></td>
<td>An error occurred in the test framework. For example, the test runner halted or the server encountered an unintended scenario.</td>
</tr>
<tr>
<td></td>
<td><strong>Cancelled</strong></td>
</tr>
<tr>
<td></td>
<td>User canceled the test or suite.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of test result item:</td>
</tr>
<tr>
<td></td>
<td>• Step Result</td>
</tr>
<tr>
<td>Summary</td>
<td>Summarized version of the Output field.</td>
</tr>
<tr>
<td></td>
<td>If an error occurred in the test step, the following text indicates that a client error occurred at this step, and includes a detailed exception/error message.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Description</td>
<td>This step failed because the client error 'DETAILED ERROR MESSAGE' was detected on the page being tested. See failing Test Logs. To ignore these errors in the next test run, use 'Add all client errors to warning/ignored list' links. Test designers and developers should always investigate client errors to determine if there are issues with your business process. For more details, see Identify and resolve client errors. For a Send REST Request - Inbound step, the URL for the request endpoint, and a response code. For example: The HTTP request has been sent to the endpoint '<a href="https://demonightlyus.service-now.com/api/now/table/kb_knowledge">https://demonightlyus.service-now.com/api/now/table/kb_knowledge</a>'</td>
</tr>
<tr>
<td>Output</td>
<td>Generated output for the test step.                                                                                           • For a step result, the outputs from the step, including any explanation why a step failed or was skipped.                                                                 • For a Send REST Request - Inbound step, the REST request and response including the response body. The output field is truncated at 4096 characters. ◦ The REST request and response headers are filtered to prevent sensitive information from being added to the log. A filtered header text is replaced with the text &quot;Header redacted for security.&quot; ◦ See Filter REST request and response headers for information on how to add a list of REST request and response headers to be filtered. For additional console logging and test execution information for a test step, see the Test Logs record.</td>
</tr>
<tr>
<td>Step</td>
<td>Name of the step executed.</td>
</tr>
<tr>
<td>Test Result</td>
<td>Reference to the test result associated with this step result. Click the information icon (i) to view detailed test results information.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the specific actions performed (for example, Open Create Incident Report Producer, Validate that the form matches the following condition) for this test step. For a client log, blank.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Time</td>
<td>Date and time of day at which this step or log entry started.</td>
</tr>
<tr>
<td>End Time</td>
<td>Date and time of day at which this step or log entry ended.</td>
</tr>
<tr>
<td>Duration</td>
<td>Elapsed time it took to execute this test.</td>
</tr>
<tr>
<td>Step Transactions (related list)</td>
<td>Record for each transaction (from the system transaction log [syslog_transaction] related to this step result.</td>
</tr>
</tbody>
</table>

**Related reference**
- Test results record
- Test logs record

**Related information**
- Allowed client errors
- Allow client errors from step results

**Test logs record**
The Test Results Item (test log) record contains console logging and test execution information.

During test execution, any information reported to the environment is recorded in the Test Log (sys_atf_test_result_item) table. This information can include browser console logging, results, and error messages recorded by step environments. You access specific test logs from the Test Log related list in the Test Results form.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>The result of the test log:</td>
</tr>
<tr>
<td></td>
<td>Success</td>
</tr>
<tr>
<td></td>
<td>Test step run passed.</td>
</tr>
<tr>
<td></td>
<td>Failure</td>
</tr>
<tr>
<td></td>
<td>Test step run failed.</td>
</tr>
<tr>
<td></td>
<td>Success with warning(s)</td>
</tr>
<tr>
<td></td>
<td>Test step passed but encountered client error(s) that were allowed as warnings in the Allowed Client Error table.</td>
</tr>
<tr>
<td></td>
<td>Ignored</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Client error that has been allowed with a report level of Ignored.</td>
<td></td>
</tr>
<tr>
<td>Warning</td>
<td>Client error that has been allowed with a report level of Warning.</td>
</tr>
<tr>
<td>Waiting</td>
<td>Test or suite waiting to start.</td>
</tr>
<tr>
<td>Running</td>
<td>Test or suite is running.</td>
</tr>
<tr>
<td>Skipped</td>
<td>Occurs if an earlier test in the suite failed and Abort on failure is set to true.</td>
</tr>
<tr>
<td>Error</td>
<td>An error occurred in the test framework. For example, the test runner halted or the server encountered an unintended scenario. The error message appears in the Summary field of the Steps Results record for the step that threw the error.</td>
</tr>
<tr>
<td>Cancelled</td>
<td>User canceled the test or suite.</td>
</tr>
</tbody>
</table>

**Type**

Type of test log:
- Step Result
- Client Log
- Client Error

**Note:** If a Client Error, you can optionally add it as an ignored or warning entry in the Allowed Client Errors. Doing so prevents the allowed client errors from affecting ATF test executions when they recur in future test runs. For more details, refer to Allowed client errors.

**Output**

Output generated for the test log.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step</td>
<td>Name of the step executed. This field may be blank for a client log.</td>
</tr>
<tr>
<td>Test Result</td>
<td>Reference to the test result associated with this step result. Click <img src="907" alt="link" /> to view detailed test result information.</td>
</tr>
<tr>
<td>Description</td>
<td>For a step result, the actions performed in this test step.</td>
</tr>
<tr>
<td></td>
<td>For a client log, blank.</td>
</tr>
<tr>
<td></td>
<td>For a client, this message displays: This client error occurred on the page in Browser Type Browser Version (for example, Chrome 62.0.3202.62).</td>
</tr>
<tr>
<td>Start time</td>
<td>Time at which this test step started executing.</td>
</tr>
<tr>
<td>End time</td>
<td>Time at which this test step finished executing.</td>
</tr>
<tr>
<td>Duration</td>
<td>Time duration it took to execute this test step.</td>
</tr>
<tr>
<td>Recorded at</td>
<td>Time at which this step or log entry was recorded.</td>
</tr>
<tr>
<td>Allow listed client error</td>
<td>Reference to the Allowed Client Error record (if any). Click <img src="907" alt="link" /> to view the Allowed Client Error record for this client error.</td>
</tr>
</tbody>
</table>

**Related reference**
- Test results record
- Step results record
- Allowed client error records

**Related information**
- Allow client errors from the test logs
Suite results

The Suites Results module opens the Suites Results table. You can drill down to the Test results record to view details about the results of individual test suites, the individual tests within those test suites, any child test suites, and so on.

Related information

Building and running automated test suites

Test suite results record

The Test Suite Results record displays information about the results of one execution of one test suite.

Fields

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Time</td>
<td>(Appears in list of records, but not on default record form). The time this test suite started.</td>
</tr>
<tr>
<td>Test Suite</td>
<td>The test suite that was run.</td>
</tr>
<tr>
<td>Number</td>
<td>Unique ID for this test suite results record.</td>
</tr>
<tr>
<td>Base Test Suite Result</td>
<td>If this test suite is a child in a hierarchy of test suites, the base test suite result is the unique ID of the result record for the suite at the top-most level of the hierarchy. For more information, see Example: Base test suite result.</td>
</tr>
<tr>
<td>Parent</td>
<td>If this test suite has a parent, this is the test result record for the parent suite. For more information, see Example: Parent test suite results.</td>
</tr>
</tbody>
</table>
| Status          | Result of the test or test suite execution:  
  • Canceled – user canceled the test or test suite run.  
  • Error – an error occurred in the test framework. For example, the test runner halted or the server encountered an unintended scenario. The error message appears in the summary field of the Step Results form for the test step that contains the error.  
  • Failure – test, test suite run, or test step failed due to an error.  
  • Ignored – test step passed but contains client-side JavaScript errors entered into the Allowed Browser Error table with report level of Ignored.  
  • Running – test or test suite is running. |
### Fields (continued)

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Skipped – occurs if an earlier test in the suite failed and <strong>Aborted on failure</strong> is set to <strong>true</strong>.</td>
<td></td>
</tr>
<tr>
<td>Success – test, test suite, or test step run passed.</td>
<td></td>
</tr>
<tr>
<td>Success with warning(s) – test or test suite run passed but contains a test step with client-side JavaScript errors entered into the Allowed Browser Error table with report level of Warning.</td>
<td></td>
</tr>
<tr>
<td>Waiting – test or suite run is waited to start.</td>
<td></td>
</tr>
<tr>
<td>Warning – test or suite contains client-side JavaScript errors entered into the Allowed Browser Error table with report level of Warning.</td>
<td></td>
</tr>
<tr>
<td>Run time</td>
<td>The duration it took to execute this test suite.</td>
</tr>
<tr>
<td>Retain indefinitely</td>
<td>Check box to keep this record even after the specified data retention period has passed. For more information, see Table Cleanup.</td>
</tr>
<tr>
<td>Rolled up test success count</td>
<td>How many tests were successful. The tests counted as part of the roll up are all tests included in this suite, plus all others included in suites that are descendents of this one. For more information, see Rolled up counts for test suites results.</td>
</tr>
<tr>
<td>Rolled up test failure count</td>
<td>How many tests failed. The tests counted as part of the roll up are all tests included in this suite, plus all others included in suites that are descendents of this one. For more information, see Rolled up counts for test suites results.</td>
</tr>
<tr>
<td>Rolled up test error count</td>
<td>How many tests resulted in an error. The tests counted as part of the roll up are all tests included in this suite, plus all others included in suites that are descendents of this one. For more information, see Rolled up counts for test suites results.</td>
</tr>
<tr>
<td>Rolled up test skip account</td>
<td>How many tests were skipped. The tests counted as part of the roll up are all tests included in this suite, plus all others included in suites that are descendents of this one. For more information, see Rolled up counts for test suites results.</td>
</tr>
<tr>
<td>Test Results (related list)</td>
<td>Results of the individual tests included in this test suite.</td>
</tr>
</tbody>
</table>
### Fields (continued)

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Child Test Suites Results (related list)</td>
<td>The results of any test suites that are children of this test suite.</td>
</tr>
<tr>
<td>All Test Suite Results (related list)</td>
<td>Results from this test suite and all descendent test suites.</td>
</tr>
<tr>
<td>Failed Tests in Suite (related list)</td>
<td>Results from any failed tests included in this test suite.</td>
</tr>
<tr>
<td>Previous suite result</td>
<td>Only appears if this suite result is for a re-run of failed tests and you are logged on with the atf_test_admin, atf_test_designer, or admin role. Click the information icon to navigate to the &quot;original&quot; suite result record.</td>
</tr>
</tbody>
</table>

**Related information**

**Test results**

**Test suite results examples**

Examples of relationship terms and how aggregated results roll up for test suites.
Example: Base test suite result

The base test suite is the top-level test suite in the hierarchy. For example, if you are viewing the test suite results record for (A) Test Suite 1.3.1, the base test suite result field links to the test suite results record for (B) Test Suite 1.

Example: Parent test suite results

The parent test suite is the test suite immediately above the one you are currently viewing. For example, if you are viewing the test suite results record for (A) Test Suite 1.3.1, the parent field links to the test suite results record for (B) Test Suite 1.3.
Rolled up counts for test suites results

You can view rolled up counts for a test suite for four metrics: test successes, failures, errors, and skips. Each of these sum data from all tests in the test suite plus all tests in the test suite’s descendents.

For example, if you are viewing the test suite results record for (A) Test Suite 1.3.1, the Rolled up test success count field shows the total number of successes counting results from all the tests represented by boxes labeled A.

If you are viewing the test suite results record for (B) Test Suite 1.3.1, the Rolled up test success count field shows the total number of successes counting results from all the tests represented by boxes labeled B.
Child Test Suites Results

If you are viewing the test suite results record for (A) Test Suite 1, the Child Test Suites Results related list shows result records for Test Suite 1.3 (B).
All Test Suites Results

If you are viewing the test suite results record for (A) Test Suite 1, the All Suites Results related list shows result records for all Test Suites represented by boxes labeled A.

If you are viewing the test suite results record for (B) Test Suite 1.3, the All Suites Results related list shows result records for all Test Suite 1.3.1 only (represented by the box labeled B)
Suite schedules

Open the Suites Schedules table. You can drill down to see details about the results of individual schedules or create a new schedule.

Related information

Building and running automated test suites

Suite schedule record

The Suite Results record displays information about one test suite schedule.

Fields

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A name for this test suite schedule.</td>
</tr>
<tr>
<td>Description</td>
<td>A description for this test suite schedule.</td>
</tr>
<tr>
<td>Active</td>
<td>If this schedule is active <strong>true</strong>. Otherwise, <strong>false</strong>.</td>
</tr>
<tr>
<td>Application</td>
<td>The application scope for this test suite schedule.</td>
</tr>
<tr>
<td>Run</td>
<td>The frequency with which the system runs test suites belonging to this schedule.</td>
</tr>
</tbody>
</table>
### Fields (continued)

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run as tz</td>
<td>The timezone used for the Time field.</td>
</tr>
<tr>
<td>Time</td>
<td>The time of day at which the system runs test suites belonging to this schedule.</td>
</tr>
<tr>
<td>Day</td>
<td>If Run is set to Weekly, the day of the week on which the system runs test suites belonging to this schedule. If Run is set to monthly, the day of the month.</td>
</tr>
<tr>
<td>Repeat interval</td>
<td>If Run is set to Periodically, the number of days and hours that constitute the repeat interval for running suites in this schedule.</td>
</tr>
<tr>
<td>Conditional</td>
<td>Check to enable a script to define conditions under which the system should run the test suites in this schedule.</td>
</tr>
<tr>
<td>Condition</td>
<td>If Conditional is checked, the script to execute for determining the conditions under which the system should run the test suites in this schedule.</td>
</tr>
<tr>
<td>Scheduled Suite Runs</td>
<td>The test suites the system should run on this schedule.</td>
</tr>
<tr>
<td>(related list)</td>
<td></td>
</tr>
</tbody>
</table>

### Related information

**Schedule an automated test suite**

**Scheduled suite run record**

A Scheduled Suite Run record associates a Suite Schedule record with a Test Suite.

### Fields

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schedule</td>
<td>The schedule to use.</td>
</tr>
<tr>
<td>Test Suite</td>
<td>The test suite to run.</td>
</tr>
<tr>
<td>Application</td>
<td>The application scope for this scheduled suite run.</td>
</tr>
<tr>
<td>Watch list</td>
<td>Users the system notifies when this scheduled suite run completes.</td>
</tr>
</tbody>
</table>
### Fields (continued)

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Browser name</td>
<td>If the test suite has UI components, the browser that must be used. If no client test runner is available with this browser, the system does not run the suite.</td>
</tr>
<tr>
<td>Browser version starts with</td>
<td>If the test suite has UI components, the browser version for the client test runner must start with this string. If no client test runner is available with this browser version, the system does not run the suite.</td>
</tr>
<tr>
<td>OS name</td>
<td>If the test suite has UI components, the client test runner must run under this OS. If no client test runner is available with this OS, the system does not run the suite.</td>
</tr>
<tr>
<td>OS version starts with</td>
<td>If the test suite has UI components, the OS version for the client test runner must start with this string. If no client test runner is available with this OS version, the system does not run the suite.</td>
</tr>
<tr>
<td>Test Suite Results (related list)</td>
<td>All test suite results from this scheduled run.</td>
</tr>
</tbody>
</table>

To determine the browser name and version of a browser you want to use, start a scheduled test runner with that browser, then inspect that runner's record in the **Active Scheduled Test Runners Module**.

### Related information

- **Schedule an automated test suite**

### Run

Start a client test runner and view information about test runners and test runs.

For information on how to work with test runners, see **Working with client test runners**.

### Client test runner

The Client Test Runner opens a browser window or tab for running manually-started client automated tests.
You can toggle a client test runner to act as either a manual or scheduled client test runner. For more information about scheduling test suites, see Schedule an automated test suite.

### Fields on the Client Runner Test window

<table>
<thead>
<tr>
<th>Field / UI Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form preferences icon ( getIntent() )</td>
<td>Click to display the form preferences panel.</td>
</tr>
<tr>
<td>Form preferences panel: Screenshots mode</td>
<td>Choose among:</td>
</tr>
<tr>
<td></td>
<td>• Enable for all steps</td>
</tr>
<tr>
<td></td>
<td>• Enable for failed steps</td>
</tr>
<tr>
<td></td>
<td>• Disable for all steps</td>
</tr>
<tr>
<td></td>
<td>For additional information, see Set the system property to control when the Automated Test Framework captures screenshots.</td>
</tr>
<tr>
<td>Form preferences panel: Run scheduled tests only</td>
<td>Click to toggle between:</td>
</tr>
<tr>
<td></td>
<td>• On (green): Use this client test runner to run only scheduled tests and suites.</td>
</tr>
<tr>
<td></td>
<td>• Off (gray): Use this client test runner to run only manually-started tests and suites.</td>
</tr>
</tbody>
</table>

**Scheduled client test runner**

The Scheduled Client Test Run opens a browser window for running scheduled client automated tests.

For information about scheduling automated tests, see Schedule an automated test suite and Working with scheduled test suites. You can toggle a client test runner to act as either a manual or scheduled client test runner.

### Fields on the Client Runner Test window

<table>
<thead>
<tr>
<th>Field / UI Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form preferences icon ( getIntent() )</td>
<td>Click to display the form preferences panel.</td>
</tr>
</tbody>
</table>
### Fields on the Client Runner Test window (continued)

<table>
<thead>
<tr>
<th>Field / UI Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form preferences panel: Screenshots mode</td>
<td>Choose among:</td>
</tr>
<tr>
<td></td>
<td>• Enable for all steps</td>
</tr>
<tr>
<td></td>
<td>• Enable for failed steps</td>
</tr>
<tr>
<td></td>
<td>• Disable for all steps</td>
</tr>
<tr>
<td></td>
<td>For additional information, see Set the system property to control when the Automated Test Framework captures screenshots.</td>
</tr>
<tr>
<td>Form preferences panel: Run scheduled tests only</td>
<td>Click to toggle between:</td>
</tr>
<tr>
<td></td>
<td>• On (green): Use this client test runner to run only scheduled tests and suites.</td>
</tr>
<tr>
<td></td>
<td>• Off (gray): Use this client test runner to run only manually-started tests and suites.</td>
</tr>
</tbody>
</table>

### Active manual test runners

View the client test runners table filtered to show only those runners available to run manually-started tests.

When you start a manual client test runner, the system registers that runner in the Test Runners table as active.

The data in this table is transient. While the runner is active, it reports in to the system at a specified interval. If the runner does not report in at the expected time, the system marks the runner as inactive. After a period of time the system deletes the runner. You can modify these intervals on the Automated Test Framework properties page.
### Example Active Manual Client Runner Test table

![Example Active Manual Client Runner Test table](image)

### Fields on the Client Runner Test table

<table>
<thead>
<tr>
<th>Field / UI Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td>Whether this test runner is for manual or scheduled tests. In the Active Manual Test Runners module, Type is always <strong>Manual</strong>.</td>
</tr>
<tr>
<td><strong>User</strong></td>
<td>The user logged into the browser session.</td>
</tr>
<tr>
<td><strong>Browser name</strong></td>
<td>The browser name.</td>
</tr>
<tr>
<td><strong>Browser version</strong></td>
<td>The browser version.</td>
</tr>
<tr>
<td><strong>OS name</strong></td>
<td>The name of the operating system running the browser.</td>
</tr>
<tr>
<td><strong>OS version</strong></td>
<td>The version of the operating system running the browser.</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>Whether this runner is currently online or offline. In the Active Manual Test Runners module, Status is always <strong>Online</strong>.</td>
</tr>
<tr>
<td><strong>Last checkin</strong></td>
<td>The time/date this runner most recently reported in to the system.</td>
</tr>
<tr>
<td><strong>Status reason</strong></td>
<td>If the Status is Offline, the reason why. In the Active Manual Test Runners module, Status reason is always empty.</td>
</tr>
</tbody>
</table>

### Active scheduled test runners

View the client test runners table filtered to show only those runners available to run tests to be started by a schedule.
When you start a **scheduled client test runner**, the system registers that runner in the Active Scheduled Test Runners table.

The Active Scheduled Test Runner module is useful when you create a scheduled suite run. For scheduled suite runs, you can specify the browser to use. To determine the name and version of a browser you want to use, start a scheduled test runner with that browser, then inspect that runner’s record in the Active Scheduled Test Runners module.

The data in this table is transient. While the runner is active, it reports in to the system at a specified interval. If the runner does not report in at the expected time, the system marks the runner as inactive. After a period of time the system deletes the runner.

You can modify these intervals on the Automated Test Framework properties page.

### Example Scheduled Client Runner Test table

<table>
<thead>
<tr>
<th>Field / UI Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td>Whether this test runner is for manual or scheduled tests. In the Active Scheduled Test Runners module, Type is always <strong>Scheduled</strong>.</td>
</tr>
<tr>
<td><strong>User</strong></td>
<td>The user logged into the browser session.</td>
</tr>
<tr>
<td><strong>Browser name</strong></td>
<td>The browser name.</td>
</tr>
<tr>
<td><strong>Browser version</strong></td>
<td>The browser version.</td>
</tr>
<tr>
<td><strong>OS name</strong></td>
<td>The name of the operating system running the browser.</td>
</tr>
<tr>
<td><strong>OS version</strong></td>
<td>The version of the operating system running the browser.</td>
</tr>
</tbody>
</table>
Fields on the Client Runner Test table (continued)

<table>
<thead>
<tr>
<th>Field / UI Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Whether this runner is currently online or offline. In the Active Manual Test Runners module, Status is always Online.</td>
</tr>
<tr>
<td>Last checkin</td>
<td>The time/date this runner most recently reported in to the system.</td>
</tr>
<tr>
<td>Status reason</td>
<td>If the Status is Offline, the reason why. In the Active Scheduled Test Runners module, Status reason is always empty.</td>
</tr>
</tbody>
</table>

**Waiting/running test runs**

The Waiting/Running Test Run module opens a list of records showing the tests waiting to be run.

⚠️ **Note:** To prevent conflicts, the system allows only one test to run at a given time. This is true even if you have multiple client test runner windows open. If you submit tests to run when another test is already running, the system holds the new tests in a queue to run later. If a test remains in the queue for more than ten minutes, the system cancels the test.

You can cancel execution of a waiting automated test.

**Waiting/running suite runs**

The Waiting/Running Suite Runs module opens a list of records showing the test suites waiting to be run.

**Allowed client error records**

Review the list of existing Allowed Client Error [sys_atf_whitelist] records to see which client errors produce warnings and which are ignored. Modify existing Allowed Client Error records as needed or create new ones.

**Allowed Client Error form**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report level</td>
<td>Report action to take when the client error is encountered. Options include:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Warning – Step &amp; Test will report Success with Warning(s):</strong> Test steps containing the allowed client error report a status of <strong>Success with warning(s).</strong> The error message appears in the test result.</td>
</tr>
</tbody>
</table>
Allowed Client Error form (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>result output, and is recorded in the test logs with the status Warning.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Ignored – Step &amp; Test will report Success</strong>: Test steps containing the</td>
</tr>
<tr>
<td></td>
<td>allowed client error report a status of Success. The error is recorded in</td>
</tr>
<tr>
<td></td>
<td>the test logs with an <strong>Ignored</strong> status.</td>
</tr>
<tr>
<td>Active</td>
<td>Check box to enable or disable allowing of a client error.</td>
</tr>
<tr>
<td>Error message</td>
<td>Message of the client error you want to allow.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the error you want to allow. If this client error was</td>
</tr>
<tr>
<td></td>
<td>allowed from a test result, step result, or a test log, the test log</td>
</tr>
<tr>
<td></td>
<td>description is copied into this field.</td>
</tr>
</tbody>
</table>

Related information

Allowed client errors

Reported client errors

The Reported Client Error module lists test logs across all tests that are client errors and have failed. You can review individual test log records, and allowed client errors as needed to prevent them from adversely impacting future test executions.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start time</td>
<td>Time at which this test step started executing.</td>
</tr>
<tr>
<td>Step</td>
<td>Name of the step executed.</td>
</tr>
<tr>
<td>Status</td>
<td>The result of the test log:</td>
</tr>
<tr>
<td></td>
<td><strong>Failure</strong></td>
</tr>
<tr>
<td></td>
<td>Test step run failed.</td>
</tr>
<tr>
<td>Output</td>
<td>Output generated for the test log. For a client error, the actual client error is displayed.</td>
</tr>
<tr>
<td>Duration</td>
<td>Time duration it took to execute this test step.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of test log:</td>
</tr>
<tr>
<td></td>
<td>• Client Error</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>✉️ Note: You can optionally add client errors as an ignored or warning entry in the Allowed Client Errors. Doing so prevents the allowed client errors from affecting ATF test executions when they recur in future test runs. For more details, see Allowed client errors.</td>
</tr>
<tr>
<td>Recorded at</td>
<td>Time at which this step or log entry was recorded.</td>
</tr>
</tbody>
</table>

**Related information**

- Allowed client errors
- Manually allow client errors
- Identify and resolve client errors

**Administration**

The Administration module contains forms for configuring and managing the automated test framework.

**Properties**

On the Properties form, you can set parameters that control how the system executes automated tests and test suites.

**Test/Test Suite Properties**

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable test/test suite execution</td>
<td>If checked, enables running tests and test suites on this instance. This setting is unchecked by default to prevent users from unintentionally running tests on production instances.</td>
</tr>
<tr>
<td>sn_atf.runner.enabled</td>
<td></td>
</tr>
<tr>
<td>Enable scheduled test suite execution</td>
<td>If checked, enables scheduling test suites on this instance.</td>
</tr>
<tr>
<td>sn_atf.schedule.enabled</td>
<td></td>
</tr>
</tbody>
</table>
### Test Debugging Properties

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable additional debugging</td>
<td>Enables additional debugging functionality, including adding a debugging tab on the client Test Runner UI page and saving UI Test Result JSON to the test result record.</td>
</tr>
<tr>
<td>functionality</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_atf.debug</td>
<td></td>
</tr>
</tbody>
</table>

### Screenshot Properties

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable or disable screenshot</td>
<td>• To capture screenshots for all steps, select <strong>Enable for all steps</strong>.</td>
</tr>
<tr>
<td>capture during test execution.</td>
<td>• To capture screenshots only for failed steps, select <strong>Enable for all failed steps</strong>.</td>
</tr>
<tr>
<td>sn_atf.screenshots.mode</td>
<td>• To capture no screenshots, select <strong>Disable for all steps</strong>.</td>
</tr>
</tbody>
</table>

**Note:** For additional information, see Set the system property to control when the Automated Test Framework captures screenshots.

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_atf.screenshots.use_glide</td>
<td>Improves the fidelity of screenshots on Workspaces and other interfaces.</td>
</tr>
<tr>
<td>screenshot</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** IE and Safari don't support this property.

**Note:** If this property is disabled or your current browser doesn't support it, html2canvas is used to take screenshots.

**Note:** If you are upgrading to a new release version, this property is set to false by default.

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_atf.screenshots.use_glide</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable capturing the full</td>
<td>Enables dynamic resizing of the test frame and capturing the full page while taking a screenshot.</td>
</tr>
<tr>
<td>page when taking a screenshot</td>
<td></td>
</tr>
</tbody>
</table>
### Screenshot Properties (continued)

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sn_atf.screenshots.capture_full_page</code></td>
<td><img src="sn" alt="Note:" /> By default, the property is set to false because it can significantly slow test execution time. <img src="sn" alt="Note:" /> If the property is disabled, only the portion of the page that is visible in the test frame is captured.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Screenshot timeout</td>
<td>Skips a screenshot capture attempt in the Client Test Runner if it exceeds this value in seconds. Users should review performance settings and browser caches on affected client systems before increasing this value.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of pixels for screenshot height</td>
<td>Numeric value representing the number of pixels for the screenshot height. Default value: 600</td>
</tr>
<tr>
<td><code>sn_atf.atf_test_runner.testframe.min_height</code></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of pixels for screenshot width</td>
<td>Numeric value representing the number of pixels for the screenshot width. Default value: 800</td>
</tr>
<tr>
<td><code>sn_atf.atf_test_runner.testframe.min_width</code></td>
<td></td>
</tr>
</tbody>
</table>

### Custom UI Page Data Capture Properties

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable tests with Custom UI steps to capture page data each time they are run.</td>
<td>When true, Custom UI test steps retrieve page data every time the test runs. Set this property to true when developing custom UI pages to always run tests on the most recent page version. When false, Custom UI test steps do not retrieve page data unless the test designer manually selects <code>Retrieve Components</code></td>
</tr>
</tbody>
</table>
### Custom UI Page Data Capture Properties (continued)

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>during test design. Set this property to false when UI development is complete to enable faster test runs.</td>
</tr>
</tbody>
</table>

### Test Runner Properties

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test runner timeout</td>
<td></td>
</tr>
<tr>
<td>sn_atf.runner.heartbeat.timeout</td>
<td>If there is no heartbeat from the test runner within this period of time in seconds, the status changes from online to offline. The value of this property should be between 120 and 1800. Default value: 120</td>
</tr>
<tr>
<td>Test runner heartbeat interval</td>
<td></td>
</tr>
<tr>
<td>sn_atf.runner.heartbeat.interval</td>
<td>Time interval in seconds for sending a heartbeat from the test runner to the server.</td>
</tr>
<tr>
<td>Offline test runner retention interval</td>
<td></td>
</tr>
<tr>
<td>sn_atf.runner.offline_retention.timeout</td>
<td>If an offline test runner does not communicate with the system for this period of days, the system deletes that test runner.</td>
</tr>
</tbody>
</table>

### Test Suite Report Properties

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test suite report properties</td>
<td></td>
</tr>
<tr>
<td>sn_atf.schedule.reports.suite.aging_threshold</td>
<td>The number of test suite results to display in the test suite aging report.</td>
</tr>
</tbody>
</table>

### Email Properties

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean value for results displayed in scheduled suite result emails.</td>
<td></td>
</tr>
<tr>
<td>sn_atf.schedule.suite_result_email.only_show_failed_results</td>
<td>When true, the scheduled suite test result emails only</td>
</tr>
</tbody>
</table>
### Email Properties (continued)

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>show results that failed. When false, displays all results.</td>
<td>Maximum number of test results to be displayed in scheduled suite result emails.</td>
</tr>
<tr>
<td>Maximum number of test results to be displayed in scheduled suite result emails.</td>
<td>sn_atf.schedule.suite_result_email.max_test_history</td>
</tr>
<tr>
<td>Maximum depth when printing suite results in suite result emails.</td>
<td>sn_atf.schedule.suite_result_email.max_depth</td>
</tr>
<tr>
<td>Color to indicate an ATF test failed in scheduled suite result emails.</td>
<td>sn_atf.schedule.suite_result_email.fail_color</td>
</tr>
<tr>
<td>Color to indicate an ATF test errored in scheduled suite result emails.</td>
<td>sn_atf.schedule.suite_result_email.error_color</td>
</tr>
<tr>
<td>Color to indicate an ATF test passed in scheduled suite result emails.</td>
<td>sn_atf.schedule.suite_result_email.pass_color</td>
</tr>
<tr>
<td>Color to indicate an ATF test was skipped in scheduled suite result emails.</td>
<td>sn_atf.schedule.suite_result_email.skip_color</td>
</tr>
</tbody>
</table>
### Email Properties (continued)

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_atf.schedule.suite_result_email.skip_color</td>
<td>to indicate an ATF test was skipped in scheduled suite result emails.</td>
</tr>
<tr>
<td></td>
<td>Color to indicate an ATF test was canceled in scheduled suite result emails.</td>
</tr>
<tr>
<td>sn_atf.schedule.suite_result_email.cancel_color</td>
<td>Hexadecimal code for color to indicate an ATF test was canceled in scheduled suite result emails.</td>
</tr>
</tbody>
</table>

### Step configurations

Step configuration records define how each type of step behaves.

### Automated Test Framework Step Config record

The step config record controls how a test step of this type behaves.

#### Fields

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name for this test step configuration.</td>
</tr>
<tr>
<td>Active</td>
<td><strong>True</strong> if this step configuration is active. Otherwise, <strong>false</strong>.</td>
</tr>
<tr>
<td>Step Environment</td>
<td>The <strong>step environment</strong> in which a step with this configuration can run.</td>
</tr>
<tr>
<td>Category</td>
<td>The <strong>category</strong> assigned to a step with this configuration.</td>
</tr>
<tr>
<td>Application</td>
<td>The application scope in which the system runs steps with this configuration.</td>
</tr>
<tr>
<td>Batch order constraint</td>
<td>Constrains where a step with this configuration can appear in a test:</td>
</tr>
<tr>
<td>Field / Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>• <strong>None</strong>: A step based on this configuration can appear at any point in a test.</td>
<td></td>
</tr>
<tr>
<td>• <strong>Start Batch Execution</strong>: If a test includes this a step based on this configuration, it must be the first step in the test.</td>
<td></td>
</tr>
<tr>
<td>• <strong>Run in the middle of an execution</strong>: If a test includes a step based on this configuration, it cannot be the first or last step.</td>
<td></td>
</tr>
<tr>
<td>• <strong>Stop Execution</strong>: If a test includes a step based on this configuration, it must be the last step in the test.</td>
<td></td>
</tr>
<tr>
<td>Class type</td>
<td>For custom step configurations, this field is always <strong>Script</strong>.</td>
</tr>
<tr>
<td>Order</td>
<td>An integer specifying where steps with this configuration appear in the step list on the Add Test Step dialog. For more information, see the example using the Order field.</td>
</tr>
<tr>
<td>Template reminder</td>
<td>The instructions that appear when a step with this configuration is included in a test as part of a template. For more information, see the example of using the Template reminder field.</td>
</tr>
<tr>
<td>HTML description</td>
<td>The text that appears when the cursor highlights a step with this configuration on the Create New Step dialog. For more information, see the example using the HTML description field.</td>
</tr>
<tr>
<td>Description generation script</td>
<td>Generates the text that describes a step when a test includes it. For more information, see the example using the description generation script.</td>
</tr>
<tr>
<td>Step execution script</td>
<td>Script (javascript) that runs when a step with this config runs.</td>
</tr>
<tr>
<td>Input Variables related list</td>
<td>The variables that act as inputs for a step with this config.</td>
</tr>
<tr>
<td>Output variables related list</td>
<td>The variables that act as outputs for a step with this config.</td>
</tr>
</tbody>
</table>
Examples of step config field values

Examples of where the system displays values assigned to some of the step config fields.

Order field example

The steps in the middle column are sorted according to the values of the Order field.

Category field example

The labels in the left column show the categories for available steps. You can filter the list in the middle column, by selecting a category in the left column.
Template reminder example

Here is a portion of the new record form for a test step config for Example Custom Step.
When you add a template to a test, the system generates a set of instructions for completing the template steps and saves them in the test Description field. The text in the step config's **Template reminder field** appears as the instructions for the corresponding step.
HTML description example
Here is a portion of the new record form for a custom step named Example Custom Step.
Here is how the step appears in the Add Test Step dialog.
Description generation script example

Here is a portion of the new record form for a custom step named **Example Custom Step**.

```javascript
function generateDescription()
{
  // the global variable 'step' represents the current glide record
  var description = "This is the generated description for the example step;"
  return description;
}
```

When a step of this type is included in a test, the generated description appears in the Test Steps related list.
**Step execution scripts**

In a step configuration record, the step execution script field determines what a step with this configuration does when it runs.

**Step inputs**

The input variables to a step are determined by the inputs related list in the step configuration record. The `inputs` parameter to `executeScript()` gives the script access to these variables. For example, if the inputs related list contains two records, `var1` and `var2`, the script can reference `var1` with the expression `inputs.var1` and can reference `var2` with `inputs.var2`.

**Step outputs**

The output variables to a step are determined by the outputs related list in the step configuration record. The `outputs` parameter to `executeScript()` gives the script access to these variables. For example, if the outputs related list contains two records, `out1` and `out2`, the script can reference `out1` with the expression `outputs.out1` and can reference `out2` with `outputs.out2`.

**Step result**

The `stepResult` parameter provides access to an API that controls whether the step passes or fails. It also determines the message the step writes to the log.

The method `stepResult.setSuccess()` causes the step to succeed. The method `stepResult.setFailed()` causes the step to fail.

The method `stepResult.setOutputMessage()` sets the message to write to the log when the step succeeds or fails. It takes one parameter: the string to write to the log. If the script calls `stepResult.setOutputMessage()` more than once, the most recent value set overwrites any previous value.

**Example: Record Query step execution script**

```javascript
(function executeStep(inputs, outputs, stepResult) {
    if (gs.nil(inputs.table)) {
        stepResult.setOutputMessage(gs.getMessage("The '{0}' input variable was not specified", 'table'));
        stepResult.setFailed();
    }
    return;
})
```
```
var query = new GlideRecord(inputs.table);
query.addEncodedQuery(inputs.field_values);
query.query();
if (!query.next()) {
    stepResult.setOutputMessage(gs.getMessage("No records matching query:
    {0}",
    inputs.field_values));
    stepResult.setFailed();
} else {
    stepResult.setSuccess();
    outputs.table = inputs.table;
    outputs.first_record = query.getUniqueValue();
    stepResult.setOutputMessage(gs.getMessage("Found {0} {1} records matching
query:
    {2}",
    [query.getRowCount(),
    inputs.table,
    inputs.field_values]));
}
(inputs, outputs, stepResult));
```

Example: Custom scripted step configs

```
(function executeStep(inputs, outputs, stepResult, timeout) {
    // Waits up to the timeout for some asynchronous logic to finish
    // This script checks for completion once a second for up to 60 seconds
    var counter = 1;
    // Try for up to 60 seconds
    while (counter <= timeout) {
        // If the asynchronous logic is finished, return "true" to pass the step
        // isMyAsyncLogicFinished() can be replaced with any asynchronous event that needs to be
tested
        if (isMyAsyncLogicFinished()) {
            stepResult.setOutputMessage("Success!");
            stepResult.setSuccess();
            return;
        }
        // Wait one second, and log the total number of seconds waited
        gs.info("Waited " + counter + " seconds for asynchronous logic to finish");
        sn_atf.AutomatedTestingFramework.waitOneSecond();
        counter++;
    }

```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Note: The above example can also be used for Run Server Side script by replacing `stepResult.setSuccess()` and `stepResult.setFailed()` with `return true` and `return false`.

Step description generation script

In a step configuration record, the step description generation script field determines the step description that the system generates when a step of this type is added to a test.

For an example showing how where the description generated by this script appears, see Description generation script example.

Step

The `step` parameter to `generateDescription()` gives the script access to the step object, which in turn gives access to the input variables as defined in the step configuration record. (Input variables are defined in a related list.)

For example, if – in a step config record – the inputs related list contains two records: `var1` and `var2`, the script can reference `var1` with the expression `step.inputs.var1` and can reference `var2` with `step.inputs.var2`.

Example: Record Query description generation script

```javascript
(function generateDescription(step) {
    var td = GlideTableDescriptor.get(step.inputs.table);
    if (!td) {
        gs.log("Invalid table name in test step: " + step.inputs.table);
        return gs.getMessage("Set field values");
    }
    var descriptionGenerator = new ATFStepDescriptionGenerator();
    var description = gs.getMessage("There should be at least one record in '{0}' matching " +
        "a query of\n\n{1},
    [step.inputs.table.getDisplayValue(),
```
Add output variables to scripted steps

Execute the following steps to add additional outputs in Run Server Side Script and Custom Scripted StepConfig test steps.

Adding outputs in Run Server Side Script test step

Modify the test scripts of Run Server Side Script test step to create additional outputs of your choice.

Before you begin
Role required: admin or atf_test_admin

Procedure
1. Navigate to Automated Test Framework (ATF) > Administration > Step Configurations.
2. Search and select Run Server Side Script.
The read-only Test Step Config form shows up.

⚠ Note: Although the form is read-only, new output variables can be created.

3. Scroll down to the Output Variables related list.
The list of default output variables shows up.

4. Click New to create a new output variable.

<table>
<thead>
<tr>
<th>Field</th>
<th>Input Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Type of output variable</td>
</tr>
<tr>
<td>Application</td>
<td>Scope of the user</td>
</tr>
<tr>
<td>Label</td>
<td>User-facing name of the output variable</td>
</tr>
<tr>
<td>Column name</td>
<td>Variable name used in the script</td>
</tr>
<tr>
<td>Max length</td>
<td>Length of the data type.</td>
</tr>
</tbody>
</table>
The new output variable gets added to the Output Variable related list.


6. Select the test where you want to implement the new output variable.

7. Click Add Test Step.


9. Follow the instructions to change outputs mentioned in the Test script.

```plaintext
outputs.<column_name> = "<desired value>";
```

10. Click Submit.

   The newly created output variables are now ready to be used in any subsequent steps of the test.

Adding outputs in Custom Scripted StepConfig test step

Copy the Custom Scripted StepConfig test step and customize the copied version by adding additional outputs.

Before you begin

Role required: admin or atf_test_admin
Procedure

1. Navigate to **Automated Test Framework (ATF) > Administration > Step Configurations**.

2. Search and select Custom Scripted StepConfig. The read-only Test Step Config form shows up.

   **Note:** Since the form is read-only, you must copy the test step to customize it to add more output variables.

3. Click **Copy** to have a copied version of the test step.

   **Note:** The copied version is no longer read-only.

4. Add additional output variables in the copied version by implementing the following steps:

   a. Click **New** under Output Variables related list

   b. Modify **Step execution script** to add more output variables

   **Note:** See Adding outputs in Run Server Side Script test step to add more output variables.

   **Note:** You can use these steps to customize the test step only in the copied version.

5. Reuse the copied version of the test step in any test whenever required.
Test templates
The Test Templates module opens a list of available templates. From this module, you create, view, and edit test templates.

To view or edit a test template, click the row for the template you want. For more information, see Edit automated test steps template.

For information how to add a new template, see Create an automated test steps template. For information on how to use a template when creating a test, see Add a predefined list of steps (template) to an automated test.

Automated Test Template record
The Test Template record contains information about one test template.

Fields
<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of this test template.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope in which the system runs this test or test suite.</td>
</tr>
<tr>
<td>Test Template</td>
<td>The tests to include in this template.</td>
</tr>
<tr>
<td>Description</td>
<td>(Optional) Enter a description to identify the purpose of this test template.</td>
</tr>
</tbody>
</table>

Step configuration categories
The Step Configuration Categories module opens a list of records specifying the step categories on the Add Step dialog. From this module, you can add, delete, and edit these categories.

Categories are used for filtering the step list in the Add Step dialog. For more information, see Category field example.

Test step config category form
On the Test Step Config Category form, you specify a retention policy for a set of records on a given test results table.
Fields

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name for this step config category.</td>
</tr>
<tr>
<td>Step Environment</td>
<td>The step environment for this step config category.</td>
</tr>
<tr>
<td>Display name</td>
<td>The category name that appears in the middle column of the Add Test Step dialog when this category is selected. See an example in Step Config Category Display name.</td>
</tr>
<tr>
<td>Application scope</td>
<td>The application scope to which this policy applies.</td>
</tr>
</tbody>
</table>

Table cleanup

The Table Cleanup module opens a list of records specifying the retention policies for test result and test suite result tables and the records within them. From this module, you can view and modify these policies.

By default, the system deletes system records related to test results and test suite results 30 days after creation. To modify the retention policies for a table and its records, click the table for which you want to modify policies. For more information, see Modify data retention policy for ATF test results.

ℹ️ Note: Table cleanup policies are platform-specific policies. See Table cleaner for more information.

Autoflush form

On the Auto Flush form, you specify a retention policy for a set of records on a given test results table.

Fields

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tablename</td>
<td>The table containing the records to which the policy applies.</td>
</tr>
<tr>
<td>Matchfield</td>
<td>The field for which the system monitors duration.</td>
</tr>
<tr>
<td>Age in seconds</td>
<td>The amount of time (in seconds) the system wait before deleting the records.</td>
</tr>
<tr>
<td>Active</td>
<td>True if this policy is active. Otherwise, false.</td>
</tr>
</tbody>
</table>
### Fields (continued)

<table>
<thead>
<tr>
<th>Field / Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application scope</td>
<td>The application scope to which this policy applies.</td>
</tr>
<tr>
<td>Cascade delete</td>
<td>If checked, the system deletes all matching records plus any records referring to them. If unchecked, the system deletes matching records, but not records referring to them.</td>
</tr>
<tr>
<td>Conditions</td>
<td>The filter conditions defining the records in this table to which the policy applies.</td>
</tr>
</tbody>
</table>

**Note:** Table cleanup policies are platform-specific policies. See Table cleaner for more information.

### Step environments

A test step environment specifies where the step executes (for example, server versus browser). In this release, custom step configs can use only the Server-Independent environment.

In this release, you cannot add custom test step environments.

### ServiceNow application repository

After you develop and test a custom application, you can make the application available to company instances by publishing it to the ServiceNow application repository.

The ServiceNow application repository is a central repository for all scoped applications that are published by all ServiceNow customers. The application repository allows ServiceNow customers to upload and distribute applications between their instances. When you access the application repository, you can see and manage only the applications that are published by your own organization. You can't see or manage applications that are published by other organizations.

After you have designed, developed, and successfully tested a custom application, you can publish your application to the ServiceNow application repository to share it to other instances in your company.

### Entitlements

An entitlement refers to permission given to an instance to install a scoped application from the application repository. An instance must be entitled to an application in order for you to be able to install the application on the instance.
By default, after you publish an application to the application repository, all your company instances are entitled to the application automatically. To limit which company instances are entitled to the application, access the application repository by going to https://apprepo.service-now.com, and then change the entitlement type for the application. You can also entitle an instance again if the application entitlement has already been removed. For more information, see Manage application entitlements from the application repository.

**Using the application repository**

You can access the application repository by going to https://apprepo.service-now.com.

After you publish an application to the application repository, you can:

- Install an application from the application repository
- Manage application entitlements from the application repository
- Delete an application from the application repository
- Release a scope from the application repository
- View scopes that are available to your company
- View keys that are available to your company

**Manage applications**

Learn how to manage the applications you publish to the ServiceNow application repository.

After you have designed, developed, and successfully tested a custom application, you can publish your application to the ServiceNow application repository to share it to other instances in your company. You can see, manage and customize only the applications that are published by your own organization or scoped applications that are installed via plugins using the ServiceNow Application Repository. You can't see or manage, or customize applications that are published by other organizations.

**Manage application entitlements from the application repository**

Add or remove application entitlements to limit which instances the application can be installed on.

**Before you begin**

You can manage only the applications that you've published to the application repository. For more information, see Publish an application to the application repository.

Role required: none
About this task
By default, after you publish an application to the application repository, all your company instances are entitled to the application automatically. To limit which company instances are entitled to the application, access the application repository by going to https://apprepo.service-now.com, and then change the entitlement type for the application. You can also entitle an instance again if the application entitlement has already been removed.

Procedure
2. Log in using your HI credentials.
3. Next to the application listing, click Select Action and then click Manage Entitlements.
4. Choose an entitlement type for your application.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remove all existing entitlements</td>
<td>None of your company instances can install the application.</td>
</tr>
<tr>
<td>Entitle all instances</td>
<td>Any of your company instances can install the application. This option is selected by default when you publish an application to the application repository.</td>
</tr>
<tr>
<td>Entitle selected instances</td>
<td>Only instances that you select can install the application. Pick which instances to entitle by moving instances from the Available Instances list to the Selected Instances list.</td>
</tr>
</tbody>
</table>

5. Click OK.

Release a scope from the application repository
Release a scope from the application repository so that the scope can be used to create new scoped applications.

Before you begin
You can release an application scope only if the scope isn’t being used by any application. If your scope is being used by an application, follow the steps in Delete an application from the application repository before releasing the scope.
Role required: customer_admin on HI

**About this task**
The application repository stores the scopes of all your custom applications. You cannot create a new application using a scope that is being stored in the application repository. To *release* a scope means to remove the scope from the application repository so that you can create new applications using the scope.

**Procedure**

1. Go to [https://apprepo.service-now.com](https://apprepo.service-now.com).
2. Log in using your HI credentials.
3. Open the Scopes tab.
4. Next to the scope listing, click the trash icon (🗑).

**Publish an application to the application repository**

Publish a custom application to the application repository so that it can be installed on other instances in your organization.

**Before you begin**
To allow a developer to publish an application to the application repository, delegate the Publish to App Repo permission to the developer. For more information, see [Delegate development and deployment permissions to personnel](#).

Role required: admin, or delegated_developer with Publish To App Repo permission enabled

**Procedure**

1. Navigate to *System Applications > My Company Applications*.
2. Open the In Development tab.
3. Open the application record that you want to publish to the application repository.
4. Click the *Publish to My Application Repository* related link.
5. Click *Submit*.

**What to do next**
Install the application on company instances so that your organization can start using it. For more information, see [Install an application from the application repository](#).
By default, after you publish an application to the application repository, all your company instances are entitled to the application automatically. To limit which company instances are entitled to the application, access the application repository by going to https://apprepo.service-now.com, and then change the entitlement type for the application.

Note:
The Can Edit Application in Studio property defaults to true for new applications, but you can set it false before publishing.

In Studio, when an application customization has the Can Edit Application in Studio property set to false, the user sees this warning:

For more information, see Manage application entitlements from the application repository and Access ServiceNow Studio.

Install an application from the application repository
Install your application on an instance so that employees can start using the application that you developed.

Before you begin
• Publish an application to the application repository.

• Check the entitlement type of the application to ensure that your instance is entitled to the application. For more information, see Manage application entitlements from the application repository.

Role required: admin
Procedure

2. Find the application.
3. Next to the application listing, select a version to install.
4. Click Install.

Delete an application from the application repository

Delete an application from the application repository so that it’s no longer available to your company instances.

Before you begin

• Publish an application to the application repository
• You can delete an application only if the application is not installed on any of your company instances. Uninstall the application on all your instances before deleting it from the application repository.

Role required: customer_admin on HI

Procedure

2. Log in using your HI credentials.
3. Next to the application listing, click Select Action and then click Flag for Deletion.
4. On the confirmation window, click Yes.
   After you flag an application for deletion, the application is deleted automatically after 90 days.
5. To delete the application immediately:
   a. Open the Flagged Apps tab.
   b. Next to the application listing, click Select Action and then click Delete Immediately.
Using MID Server with source control

The ServiceNow® MID Server enables communication and the movement of data between a ServiceNow instance and external applications, data sources, and services.

How bundle files work with MID Server

The .bundle file helps source control function with a MID Server. A bundle file is the way Git packages a local repository in a single file. This makes sharing or moving the repository simpler and more streamlined. The file is then sent to the MID Server, which passes it on to the remote repository.

The outgoing.bundle (commit operations) and incoming.bundle (apply remote changes) are attached to the MID Server attachment table [ecc_agent_attachment] for any request that goes to the MID Server. The outgoing.bundle is created on the instance while the incoming.bundle is created on the MID Server.

After an operation completes successfully, the bundle file is “promoted” into a golden.bundle that is attached to the Repository configuration table [sys_repo_config]. There is one golden bundle per repository and it’s used to initialize the repository on a node that has not performed any Source Control operations yet.

The Auto Flush tool [sys_auto_flush] is a “table cleaner” that removes any ecc_agent_attachment record older than 30 days by default. This action removes the corresponding attachment as well, as they are not necessary after the cleanup operation.

⚠️ Note: If you want to customize the cleanup operation for bundle files, it is best to create and configure an additional auto-flush tool rather than changing any of the default settings.

The bundle files are kept on the MID Server and then saved to the Import directory on the MID Server.

On the MID Server, the bundle file is saved in the Import folder. This folder is under the user directory defined by the system property (user.dir), which users can configure. The bundle file is removed as part of the system flushing at the end of every operation.

Working with the MID Server

- Avoid conflicts with Discovery and create files for the system attachment [sys_attachment] table: MID Server
- Learn about system properties restrictions: Configure attachment system properties
Manage customizations to applications

You can manage your company's customizations for applications that belong to other organizations or a scoped ServiceNow plugin. You cannot change the actual applications that belong to other organizations (purchased from the Store) or the global plugins provided in the ServiceNow base system. But you can create and manage your company's customizations for store applications or scoped applications that are installed via plugins, using the ServiceNow Application Repository.

Before you begin
Role required: admin

The entitlements of your company's application-customizations are controlled solely by the entitlements of the respective store or vendor-released application. There are no separate entitlements for application-customizations, so you can't customize instances that don't have an entitlement to a base store application.

The development instance that you are using to create application-customizations should have its respective store or plugin application entitled and installed.

About this task
To customize an application

There are two ways to create application-customizations:

• In ServiceNow Studio, open the Store application or the application installed via plugin. To learn more, see Access ServiceNow Studio.

Or

1. Navigate to System Applications> All.
2. Search for the installed application you want to customize.

   Note: You can open the application in either Studio or select it as the current working application in the Store application record.

a. On the stack menu, click Edit in Studio to open this application in Studio.
Or

b. Click the name of the application to open its table record. In the Related Links section of the record, select **Switch to this Application** to make it the current working application in the Application picker.

**Related information**

*Manage application entitlements from the application repository*

**Publish customizations to an application repository**

After you have designed, developed, and successfully tested your customizations to a store application or a scoped plugin, you can publish it to the ServiceNow application repository to share it to other instances in your company.

**Before you begin**

Role required: admin, or delegated_developer with Publish To App Repo permission enabled

To enable a developer to publish an application to the application repository, delegate the Publish to App Repo permission to the developer. For more information, see *Delegate development and deployment permissions to personnel*.

**Procedure**

1. Navigate to **System Applications > All Available Applications > All**.
2. Find the application by typing its name in the Filter Navigation box.

![Filter navigator](image)

3. Open the application record for which you want to publish the customization to the application repository.

4. Click the **Publish Customizations to My App Repo** related link or in ServiceNow Studio, use the **FilePublish** menu item.

5. Click **Submit**.

**Install customizations from an application repository**

Install your customization to a store application on an instance.
Before you begin
Role required: admin

Publish a customization of a store application to the application repository. Check the entitlement type of the application to ensure that your instance is entitled to the application.

Procedure
1. Navigate to System Applications > All Available Applications > All.
2. Find the application.
3. Select a customization version to install.

Note: For scoped applications installed via plugins, the base system version is the ServiceNow release version that you are currently using (Paris, Quebec, Rome, and so on).

4. Click Install.

ServiceNow CLI
The ServiceNow CLI is a command-line interface that lets you perform instance operations from your local system. You can extend the CLI to include new commands that meet your application’s needs.

Benefits
The ServiceNow CLI lets you:
• Perform basic CRUD operations on records in your instance.
• Develop custom components and deploy them to your instance to personalize a UI.
• Create custom commands that enable you to manage custom applications from the command line.
• Use the ServiceNow CLI in scripts to simplify setup tasks and operational activities.

Architecture
Commands are stored in a table on the instance you are connected to. When the ServiceNow CLI connects to the instance, it receives all the available commands supported by that instance.

Commands map to a REST endpoint that executes asynchronously. For more information, see Create a custom command in ServiceNow CLI.

Activating ServiceNow CLI
Install ServiceNow CLI by requesting it from the ServiceNow Store. Visit the ServiceNow Store website to view all the available apps and for information about submitting requests to the store. For cumulative release notes information for all released apps, see the ServiceNow Store version history release notes.

Configuration file
The ServiceNow CLI stores profile information in a config.json file which, by default, is stored in your home directory at the following path:

• Linux and Mac: ~/.snc/config.json
• Windows: %USERPROFILE%\.snc\config.json

The CLI uses this file to determine what information to use to connect to an instance, and what settings to use to generate output. By default, the

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
ServiceNow CLI uses the settings found in the default profile to connect to an instance. To use alternate settings, you can create and reference additional named profiles. For more information, see Configure and manage your ServiceNow CLI connection profiles.

The following example shows a configuration file with a default profile and a named profile. Each profile can use different credentials and specify different hosts and output formats.

```json
{
  "profiles":{
    "default":{
      "host":"https://myinstance.service-now.com",
      "loginmethod":"basic",
      "username":"admin",
      "output":"json",
      "hostversion":"Paris",
      "appversion":"1.0"
    },
    "user1":{
      "host":"https://otherinstance.service-now.com",
      "loginmethod":"basic",
      "username":"user1",
      "output":"yaml",
      "hostversion":"Paris",
      "appversion":"1.0"
    }
  }
}
```

⚠️ Note: Sensitive credential information is only stored in the OS keychain, not in the configuration file.

**Command structure**

ServiceNow CLI commands follow this structure:

1. The base call to the `snc` program.
2. The top-level command group followed by any child command groups.
3. The command that specifies which operation to perform.
4. General CLI arguments required by the operation. You can specify arguments in any order.

```sh
$ snc <command-group> <command> [arguments]
```
Arguments can take various types of input values, such as numbers, strings, and JSON objects. The types supported depend on the command you specify.

**Argument values**
Many argument values in the ServiceNow CLI are simple string or numeric values, such as the table and table name in the following example.

```bash
$snc record create --table incident --data "{short_description: 'New Incident'}"
```

You can surround strings that do not contain any space characters with quotation marks or not. However, you must use quotation marks around strings that include one or more space characters.

**Output formats**
The ServiceNow CLI supports four output formats:

- **json**: The output is formatted as JSON. This is the default.

```
{
  "default": {
    "appversion": "1.0.8",
    "host": "https://myinstance.service-now.com",
    "hostversion": "Paris",
    "loginmethod": "basic",
    "output": "json",
    "username": "admin"
  },
  "user1": {
    "appversion": "1.0.8",
    "host": "https://otherinstance.service-now.com",
    "hostversion": "Paris",
    "loginmethod": "basic",
    "output": "yaml",
    "username": "admin"
  }
}
```

- **yaml**: The output is formatted as YAML. Use YAML to handle the output with services and tools that emit or consume YAML-formatted strings.

```
default:
  appversion: 1.0.8
  host: https://myinstance.service-now.com
  hostversion: Paris
  loginmethod: basic
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
output: json
username: admin

user1:
  appversion: 1.0.8
  host: https://otherinstance.service-now.com
  hostversion: Paris
  loginmethod: basic
  output: yaml
  username: admin

- **text**: The output is formatted as multiple lines of tab-separated string values. Use this output with traditional UNIX text tools such as grep, sed, and awk, and the text processing performed by PowerShell.

<table>
<thead>
<tr>
<th>NAME</th>
<th>HOST</th>
<th>HOSTVERSION</th>
<th>APP VERSION</th>
<th>LOGIN METHOD</th>
<th>USERNAME</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>myinstance</td>
<td>Paris</td>
<td>1.0.8</td>
<td>basic</td>
<td>admin</td>
<td>json</td>
</tr>
<tr>
<td>user1</td>
<td>otherinstance</td>
<td>Paris</td>
<td>1.0.8</td>
<td>basic</td>
<td>admin</td>
<td>yaml</td>
</tr>
</tbody>
</table>

- **table**: The output is formatted as a table which presents the information in a human-readable format.

- **none**: The CLI does not print the output to the console. Success, error, and progress messages still display.

You can specify command output in two ways:

**Use the `output` option in a named profile in the configuration file**

The following example sets the default output format to text.

```
{
  "profiles": {
    "default": {"output": "text"}
  }
}
```

**Use the `--output` argument on the command line**

The following example sets the output of a single command to JSON. This option on the command overrides any currently set value in the configuration file.

```
$ snc record query --table incident --query 'active=true' --output json
```
Install the ServiceNow CLI

Install the ServiceNow CLI on a Mac, Windows, or Linux machine.

**Before you begin**
The latest version of the ServiceNow CLI is 1.0.0. To find your installed version and see if you must update, run `snc version`.

Installing the ServiceNow CLI requires:

- Admin rights to install software on your machine.
- Software to extract or unzip the downloaded package.

Available for the following operating systems:

- Mac
- Windows
- Linux

**Install the ServiceNow CLI on Mac**
Install ServiceNow CLI on a Mac OS using the installer.

**Before you begin**
You must have an Apple-supported version of 64-bit Mac OS.

**Procedure**
1. In your browser, download the installer bundle from the ServiceNow Store.
2. Extract the OS-specific installers.
3. Open the `snc-1.0.0-osx-installer` file.
4. Follow the on-screen instructions.
   - When prompted, ensure the **Add to PATH** option is selected. This option requires root account access.
5. To verify that the installation succeeded, use the following commands.

```bash
$ which snc
/usr/local/bin/snc

$ snc version
{
  "extensions": {},
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Install the ServiceNow CLI on Windows

Install ServiceNow CLI on a Windows OS using the installer.

Before you begin
You must have a 64-bit version of Windows 10 or later.

Procedure
1. In your browser, download the installer bundle from the ServiceNow Store.
2. Extract the OS-specific installers.
3. Open the snc-1.0.0-windows-x64-installer.exe file.
4. Follow the on-screen instructions. By default, the CLI installs to C:\Program Files\ServiceNow CLI.
5. To confirm the installation, check the version in the command line using the following command.

   ```
   C:\>snc version

   {
     "extensions": {},
     "snc": "1.0.0"
   }
   ```

   If Windows is unable to find the program, close and reopen the command prompt window to refresh the path.

Install the ServiceNow CLI on Linux

Install ServiceNow CLI on a Linux machine using the installer.

Before you begin
You must have a 64-bit version of a recent distribution of CentOS or Ubuntu.

Procedure
1. In your browser, download the installer bundle from the ServiceNow Store.
2. Extract the OS-specific installers.
3. Open the snc-1.0.0-linux-x64-installer.run file.
4. Follow the on-screen instructions.
a. When prompted, ensure the **Add to PATH** option is selected. This option requires root account access.

5. To verify that the installation succeeded, use the following commands.

```bash
$ which snc
~/ServiceNow CLI/bin/snc

$ snc version
{
  "extensions": {},
  "snc": "1.0.0"
}
```

**Configure and manage your ServiceNow CLI connection profiles**

Create a connection profile to connect with your instance, view connection profiles, refresh your connection and available commands, or delete profiles you no longer need.

**About this task**
The ServiceNow CLI stores profile information in a `config.json` file which, by default, is stored in your home directory at the following path:

- **Linux and Mac:** `~/.snc/config.json`
- **Windows:** `%USERPROFILE%\snc\config.json`

The CLI uses this file to determine what information to use to connect to an instance, and what settings to use to generate output. By default, the ServiceNow CLI uses the settings found in the *default* profile to connect to an instance. To use alternate settings, you can create and reference additional named profiles. For more information, see Configure and manage your ServiceNow CLI connection profiles.

The following example shows a configuration file with a default profile and a named profile. Each profile can use different credentials and specify different hosts and output formats.

```json
{
  "profiles": {
    "default": {
      "host": "https://myinstance.service-now.com",
      "loginmethod": "basic",
      "username": "admin",
```
Create a default profile

Create a connection profile that the ServiceNow CLI uses by default. You must create a default profile to set up the CLI's initial connection with an instance.

About this task

By default, the information in this profile is used when you run a command that does not explicitly specify a profile to use.

ℹ️ Note: Sensitive credential information is only stored in the OS keychain, not in the configuration file.

Procedure

1. Open your system's command-line tool and execute this command.

```bash
$ snc configure profile set
```

2. The CLI prompts you for the following information. Enter the requested values. When prompting for information, the CLI displays current values in brackets []). To keep an existing value, press the Enter key.

<table>
<thead>
<tr>
<th>Requested information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>The host name of the instance to connect to. Supports both the full URL (<a href="https://my-instance.service-now.com">https://my-instance.service-now.com</a>) or just the host name (my-instance).</td>
</tr>
<tr>
<td>Login method</td>
<td>The login method to use to connect to the instance. Supports Basic, OAuth, and OAuth + MFA.</td>
</tr>
</tbody>
</table>
### Requested information

<table>
<thead>
<tr>
<th>Requested information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username</td>
<td>The user name to use to connect to the instance.</td>
</tr>
<tr>
<td>Password</td>
<td>The password to use to connect to the instance.</td>
</tr>
<tr>
<td>Client id</td>
<td>The client ID to use to connect to the instance when the login method is OAuth or OAuth + MFA.</td>
</tr>
<tr>
<td>Client secret</td>
<td>The client secret to use to connect to the instance when the login method is OAuth or OAuth + MFA.</td>
</tr>
<tr>
<td>Authentication code</td>
<td>The authentication code to use to connect to the instance when the login method is OAuth + MFA.</td>
</tr>
<tr>
<td>Default output format</td>
<td>Specifies how to format the command results. Options are json, yaml, text, and table.</td>
</tr>
</tbody>
</table>

### Example

```bash
$ snc configure profile set

Host:
https://myinstance.service-now.com

Login Method (Basic, OAuth, OAuth + MFA):
Basic

Username:
myusername

Password:
mypassword
```

3. Run any command using the values specified in your default profile by omitting the `--profile` attribute from your command.

### Example

```bash
$ snc record create
```

The command creates a record in the instance specified in the default profile with the specified connection options.
Create a named profile

Create a named connection profile to use with specific commands. This allows you to specify a different instance or connection protocol for a specific command.

Before you begin

A default profile must first exist to communicate with an instance.

About this task

Note: Sensitive credential information is only stored in the OS keychain, not in the configuration file.

Procedure

1. Open your system's command-line tool and execute this command. Specify the name of the named profile in the profile-name argument.

   `$ snc configure profile set [--profile profile-name]

2. The CLI prompts you for the following information. Enter the requested values. When prompting for information, the CLI displays current values in brackets [ ]. To keep an existing value, press the Enter key.

<table>
<thead>
<tr>
<th>Requested information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>The host name of the instance to connect to. Supports both the full URL (<a href="https://my-instance.service-now.com">https://my-instance.service-now.com</a>) or just the host name (my-instance).</td>
</tr>
<tr>
<td>Login method</td>
<td>The login method to use to connect to the instance. Supports Basic, OAuth, and OAuth + MFA.</td>
</tr>
<tr>
<td>Username</td>
<td>The user name to use to connect to the instance.</td>
</tr>
<tr>
<td>Password</td>
<td>The password to use to connect to the instance.</td>
</tr>
<tr>
<td>Client id</td>
<td>The client ID to use to connect to the instance when the login method is OAuth or OAuth + MFA.</td>
</tr>
<tr>
<td>Client secret</td>
<td>The client secret to use to connect to the instance when the login method is OAuth or OAuth + MFA.</td>
</tr>
</tbody>
</table>
### Requested information

<table>
<thead>
<tr>
<th>Requested information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication code</td>
<td>The authentication code to use to connect to the instance when the login method is OAuth + MFA.</td>
</tr>
<tr>
<td>Default output format</td>
<td>Specifies how to format the command results. Options are json, yaml, text, and table.</td>
</tr>
</tbody>
</table>

**Example**

```
$ snc configure profile set --profile user1
```

**Host:**
```
https://mytestinstance.service-now.com
```

**Login Method (Basic, OAuth, OAuth + MFA):**
```
Basic
```

**Username:**
```
myusername
```

**Password:**
```
mypassword
```

3. Run any command using the values specified in your named profile by adding the `--profile <profile-name>` attribute to your command.

**Example**

```
$ snc record create --profile user1
```

The command creates a record in the instance specified in the `user1` profile with the specified connection options.

### View profiles

View all connection profiles set in the configuration file, or view information about a specific profile.

### Before you begin

A default profile must first exist to communicate with an instance.

### About this task

For each profile, the CLI displays host information, version details, user name, login method, and preferred output format. It does not display sensitive information such as passwords or client IDs.
**Procedure**

Open your system’s command-line tool and execute this command. Include the `[--profile profile-name]` argument to view the connection information for a single profile. Otherwise, the CLI lists information for all profiles.

```
$ snc configure profile list [--profile profile-name]
```

**Example**

This command displays all configured profiles.

```
$ snc configure profile list
```

The CLI provides the following output.

```
{
  "profiles":{
    "default":{
      "host":"https://myinstance.service-now.com",
      "loginmethod":"basic",
      "username":"admin",
      "output":"json",
      "hostversion":"Paris",
      "appversion":"1.0"
    }
  }
}
```

**Remove a profile**

Remove a named connection profile that you no longer need from the configuration file.

**Before you begin**

A default profile must first exist to communicate with an instance.

**About this task**

You cannot remove the default connection profile using this command. To remove the default connection profile, edit the configuration file manually.

**Procedure**

Open your system’s command-line tool and execute this command. Specify the name of the named profile in the `profile-name` argument.

```
$ snc configure profile remove [--profile profile-name]
```
Results
The CLI removes the specified profile from the configuration file.

Refresh your connection
Update the available commands from the instance for the given profile. Refresh your connection after modifying any of the commands on the corresponding instance in order to keep the CLI up-to-date.

Before you begin
A default profile must first exist to communicate with an instance.

Procedure
Open your system's command-line tool and execute this command.
Include the [--profile profile-name] argument to refresh connection information from a named profile. If you do not include the argument, the CLI refreshes the default profile.

$ snc configure profile refresh [--profile profile-name]

Results
The CLI connects to the instance in the designated connection profile and updates any commands that may have changed.

Get help with ServiceNow CLI
See available commands, command options, and examples, or generate debug logging output.

Before you begin
• Install the ServiceNow CLI
• Configure and manage your ServiceNow CLI connection profiles

About this task
In addition to the debug logging output option, the CLI logs all command executions to a log file at the following locations:

• Linux and Mac: ~/.snc/.logs.
• Windows: %USERPROFILE%\snc\logs.
Procedure

1. To see the available commands and arguments, open your system’s command-line tool and enter the command you need help with, followed by the --help argument.

Example

The following example displays help for the available top-level commands.

```
$ snc --help
```

Example

The following example displays the available profile-specific commands.

```
$ snc configure profile --help
```

The CLI displays help for the requested command divided into eight sections:

**Name**

The name of the command.

<table>
<thead>
<tr>
<th>Name</th>
<th>set</th>
</tr>
</thead>
</table>

**Description**

The description of what the command does and its return values.

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure connection profiles. This command is interactive and will prompt you for each configuration value.</td>
</tr>
</tbody>
</table>

**Synopsis**

The basic syntax for using the command and its options.

```
Synopsis
snc configure profile set [arguments]
```

**Available commands**

The commands available under the current command group.

<table>
<thead>
<tr>
<th>Available Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>list : Lists the configured connection profiles.</td>
</tr>
<tr>
<td>refresh : Updates the available commands from the instance for the given profile.</td>
</tr>
<tr>
<td>remove : Removes the specified connection profile.</td>
</tr>
</tbody>
</table>
set : Configures connection profiles in order to communicate with an instance.

Command groups
The command groups available under the current command group.

Command Groups
  profile : Set, view, and remove connection profiles.

Arguments
A description of each of the arguments the command accepts.

Arguments
  -p, --profile string : Use a specific connection profile when executing a command.

Global arguments
The global arguments that the command accepts.

Global Arguments
  -d, --debug Print logs to console.
  -h, --help Display detailed help information.
  -o, --output string Set the format for printing command output.

Examples
Examples of the requested command.

Examples
  Create a new profile to save as the default:
  $ snc configure profile set
  Host:
  Login method:
  Username:
  Password:
  Client id:
  Client secret:

2. To generate debug logging output, open your system's command-line tool and enter the command you want to debug, followed by the --debug argument.

Example
The following example generates debug logging output when trying to perform the operation.
$ snc record delete --table incident --sysid 552c48888c033300964f4932b03eb092 --debug

The CLI generates debug logging output when executing the command.

**Perform record operations using ServiceNow CLI**

Create, read, update, delete, and query records in your instance using the ServiceNow CLI command-line tool.

**Before you begin**

- Install the ServiceNow CLI
- Configure and manage your ServiceNow CLI connection profiles

⚠️ **Note:** To perform record operations, install the CLI Metadata application from the ServiceNow Store on the instance you want to connect to.

**Create a record**

Inserts a single record in a specified table.

**About this task**

⚠️ **Note:** You cannot insert multiple records using this command.

**Procedure**

Open your system’s command-line tool and execute this command.

```
$ snc record create [--table table --data data]
```

Pass in values for these arguments.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>Required. Name of the table in which to save the record.</td>
</tr>
<tr>
<td>data</td>
<td>Required. Field name and the associated value for each field to define in the specified record in JSON string format.</td>
</tr>
</tbody>
</table>

**Results**

The system creates a record in the designated table and returns field-value pairs of the new record.

**Example:**

```
$ snc record create --table incident, --data "{"short_description": 'Unable to open file', 'impact':'3'}"
```
The system returns the record in JSON format.

```json
{
    "result": {
        "active": "true",
        "activity_due": "",
        "additional_assignee_list": "",
        "approval": "not requested",
        "approval_history": "",
        "approval_set": "",
        "assigned_to": "",
        "assignment_group": "",
        "business_duration": "",
        "business_service": "",
        "business_stc": "",
        "calendar_duration": "",
        "calendar_stc": "",
        "caller_id": "",
        "category": "inquiry",
        "caused_by": "",
        "child_incidents": "0",
        "close_code": "",
        "close_notes": "",
        "closed_at": "",
        "closed_by": "",
        "cmdb_ci": "",
        "comments": "",
        "comments_and_work_notes": "",
        "company": "",
        "contact_type": "",
        "contract": "",
        "correlation_display": "",
        "correlation_id": "",
        "delivery_plan": "",
        "delivery_task": "",
        "description": "",
        "due_date": "",
        "escalation": "0",
        "expected_start": "",
        "follow_up": "",
        "group_list": "",
        "hold_reason": "",
        "impact": "3",
        "incident_state": "1",
    }
}
```
Delete a record

Deletes the specified record from the specified table.

Procedure

Open your system's command-line tool and execute this command.

```bash
$ snc record delete [--table table --sysid sys_id]
```

Pass in values for these arguments.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>Required. Name of the table in which to delete the record.</td>
</tr>
<tr>
<td>sysid</td>
<td>Required. Sys_id of the record to delete.</td>
</tr>
</tbody>
</table>

Example:

```bash
$ snc record delete --table incident --sysid 552c48888c033300964f4932b03eb092
```

Get a record

Retrieves a single record based on the specified sys_id from the specified table.

Procedure

Open your system's command-line tool and execute this command.

```bash
$ snc record get [--table table --sysid sys_id]
```

Pass in values for these arguments.
Parameter | Description
---|---
table | Required. Name of the table from which to retrieve the record.
sysid | Required. Sys_id of the record to retrieve.

Example:

```
$ snc record get --table incident --sysid 552c48888c033300964f4932b03eb092
```

The system returns the record in JSON format.

```json
{
  "result": {
    "active": "true",
    "activity_due": "",
    "additional_assignee_list": "",
    "approval": "not requested",
    "approval_history": "",
    "approval_set": "",
    "assigned_to": "",
    "assignment_group": "",
    "business_duration": "",
    "business_service": "",
    "business_stc": "",
    "calendar_duration": "",
    "calendar_stc": "",
    "caller_id": {
      "link": "https://my-instance.service-now.com/api/now/table/sys_user/005d500b536073005e0addeeff7b12f4",
      "value": "005d500b536073005e0addeeff7b12f4"
    },
    "category": "inquiry",
    "caused_by": "",
    "child_incidents": "0",
    "close_code": "",
    "close_notes": "",
    "closed_at": "",
    "closed_by": "",
    "cmdb_ci": "",
    "comments": "",
    "comments_and_work_notes": "",
    "company": ""
  }
}
```
"number": "INC0010112",
"opened_at": "2019-07-29 18:48:43",
"opened_by": {
    "link": "https://my-instance.service-now.com/api/now/table/sys_user/6816f79cc0a8016401c5a33be04be441",
    "value": "6816f79cc0a8016401c5a33be04be441"
},
"order": "",
"parent": "",
"parent_incident": "",
"priority": "5",
"problem_id": "",
"reassignment_count": "0",
"reopen_count": "0",
"reopened_by": "",
"reopened_time": "",
"resolved_at": "",
"resolved_by": "",
"rfc": "",
"route_reason": "",
"service_offering": "",
"severity": "3",
"short_description": "Assessment : ATF Assessor",
"sla_due": ""
Query records
Retrieves multiple records from a specified table.

Procedure
Open your system's command-line tool and execute this command.

```
$ snc record query [--displayvalue displayValue --fields fields --limit limit --offset offset --query query --table table]
```

Pass in values for these arguments.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>displayValue</td>
<td>Include <code>--displayvalue displayValue</code> to retrieve the display value from the database for reference and choice fields. Do not include this parameter to retrieve the actual values.</td>
</tr>
<tr>
<td>fields</td>
<td>Comma-separated list of field names to return from the database.</td>
</tr>
<tr>
<td>limit</td>
<td>Maximum number of records to return.</td>
</tr>
<tr>
<td>offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval.</td>
</tr>
<tr>
<td>query</td>
<td>Required. Encoded query used to filter the result set in the following format: <code>--query '&lt;column_name&gt;&lt;operator&gt;&lt;value&gt;'</code>.</td>
</tr>
<tr>
<td>table</td>
<td>Required. Name of the table in which to query the records.</td>
</tr>
</tbody>
</table>

**Example:**

```bash
$ snc record query --displayvalue --fields short_description,state --query '123TEXTQUERY321=email' --table incident
```

The CLI returns any records that match the query.

```json
{
   "result": [
      {
         "short_description": "Unable to connect to email",
         "state": "Closed"
      }
   ]
}
```

**Update a record**

Updates the specified record with the given data attributes.

**Procedure**

Open your system's command-line tool and execute this command.

```bash
$ snc record update [--sysid sys_id --table table --data data]
```

Pass in values for these arguments.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>Required. Name of the table in which to save the record.</td>
</tr>
<tr>
<td>sysid</td>
<td>Required. Sys_id of the record to update.</td>
</tr>
<tr>
<td>data</td>
<td>Required. Field name and the associated value for each field to define in the specified record in JSON string format.</td>
</tr>
</tbody>
</table>

**Example:**

```
$ snc record update --sysid 9ef81de2db0d6090d7055268dc961978 --table incident --data
"{ 'short_description': 'Email servers down', 'urgency': '1' }"
```

The system returns field-value pairs for the updated record.

```
{
  "result": {
    "active": "true",
    "activity_due": "",
    "additional_assignee_list": "",
    "approval": "not requested",
    "approval_history": "",
    "approval_set": "",
    "assigned_to": "",
    "assignment_group": {
      "link": "https://my-instance.service-now.com/api/now/table/sys_user_group/287ebd7da9fe198100f92cc8d1d2154e",
      "value": "287ebd7da9fe198100f92cc8d1d2154e"
    },
    "business_duration": "",
    "business_service": "",
    "business_stc": "",
    "calendar_duration": "",
    "calendar_stc": "",
    "caller_id": "",
    "category": "inquiry",
    "caused_by": "",
    "child_incidents": "0",
    "close_code": "",
    "close_notes": "",
    "closed_at": "",
    "closed_by": "",
    "cmdb_ci": ""
  }
}
```
Create a custom command in ServiceNow CLI

Manage your custom application from the command line by creating custom commands in the ServiceNow CLI.

Before you begin
Role required: admin

About this task
A ServiceNow CLI command maps to a scripted REST endpoint in the End Point [sn_cli_metadata_end_point] table. You can define a scripted REST endpoint to
perform a function in your custom application, or use any existing REST endpoint. Then map a CLI command to execute the REST call.

**Procedure**

1. Make a REST endpoint available to a ServiceNow CLI command.

   **a.** Navigate to **Command Line Interface (CLI) > End Points**.

   **b.** Select **New** and complete the form.

<table>
<thead>
<tr>
<th><strong>Field</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource Path</td>
<td>Required. Path to the endpoint on the instance to map a command to. Can be the path to an inbound REST API, or a scripted REST API. For example, <code>api/now/table/{table}/{sysid}</code>. For more information, see Available REST APIs and Scripted REST APIs.</td>
</tr>
<tr>
<td>HTTP Method</td>
<td>Required. HTTP method to use when the user runs the associated command.</td>
</tr>
<tr>
<td>Application</td>
<td>Read-only application scope for the endpoint.</td>
</tr>
</tbody>
</table>

2. Optional: Create a command group. Alternatively, you can add your new command to an existing command group.

   **a.** Navigate to **Command Line Interface (CLI) > Command Groups**.

   **b.** Select **New** and complete the form.

<table>
<thead>
<tr>
<th><strong>Field</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required. Name of the command group.</td>
</tr>
<tr>
<td>Parent Group</td>
<td>Parent command group.</td>
</tr>
<tr>
<td>Reference Group</td>
<td>Command group to reference. For example, you can create a new command group as an alias for an existing command group. When the user calls a referenced command using the new command group, the original command executes. This enables you to create a command group specific to your custom application that includes both new and existing commands.</td>
</tr>
</tbody>
</table>
### Field | Description
--- | ---
Short Description | Required. Short description for the command group.
Description | Description of the command group used as help text when the user runs the \texttt{--help} command on the command group.
Application | Read-only application scope for the command group.
Active | When selected, the command group is active.

3. Create a command.

**a.** Navigate to **Command Line Interface (CLI) > Commands**.

**b.** Select **New** and complete the form.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required. Name of the command.</td>
</tr>
<tr>
<td>Command Group</td>
<td>Required. Command group that the command is a part of.</td>
</tr>
</tbody>
</table>
| Reference Command | Command to reference. For example, you can create a new command as an alias for an existing command. When the user calls the new command, the original command executes. This enables you to create a command specific to your custom application that executes existing functionality.  

\[ \textbf{Note:} \text{ A command cannot reference a command that references another command, or reference a descendant command, an ancestor command, or a callback command.} \]| |
<p>| API Endpoint | Required. API call to execute when the user runs the command. |
| Short Description | Required. Short description of the command. |
| Application | Read-only application scope for the command. |
| Active | When selected, the command is active. |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is Callback Command</td>
<td>When true, designates the command as a callback command. Select this option to hide the command from the CLI client and prevent users from calling it from the command line. Use this field with the <strong>Callback</strong> section of a primary command. The callback command executes when the primary command is complete.</td>
</tr>
</tbody>
</table>

**Help Text**

<table>
<thead>
<tr>
<th>Description</th>
<th>Description of the command used as help text when the user runs the <code>--help</code> command.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td>Examples of the command used as help text when the user runs the <code>--help</code> command.</td>
</tr>
</tbody>
</table>

**Expressions**

<table>
<thead>
<tr>
<th>Success Expression</th>
<th>Expression used to evaluate the response from the server and determine if the command succeeded. For example, <code>result.code = 1</code>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failure Expression</td>
<td>Expression used to evaluate the response from the server and determine if the command failed.</td>
</tr>
</tbody>
</table>

**Messages**

<table>
<thead>
<tr>
<th>Success Message</th>
<th>Message displayed on the CLI client when the command is successful.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Progress Message</td>
<td>Message displayed on the CLI client when the command is in progress.</td>
</tr>
<tr>
<td>Failure Message</td>
<td>Message displayed on the CLI client when the command fails.</td>
</tr>
</tbody>
</table>

**Callback**

<table>
<thead>
<tr>
<th>Callback Expression</th>
<th>Expression used to determine whether to execute the callback command. For example, you can write an expression that checks on a long-running process. If the expression produces a certain result, the callback executes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Callback Command</td>
<td>Command to execute when the <strong>Callback Expression</strong> is satisfied. Must be a command with the <strong>Is Callback Command</strong> field selected.</td>
</tr>
</tbody>
</table>
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Callback Interval</td>
<td>The interval between callback command executions.</td>
</tr>
<tr>
<td></td>
<td>Unit: Milliseconds</td>
</tr>
<tr>
<td></td>
<td>Default: 1,000</td>
</tr>
<tr>
<td>Max Retries</td>
<td>Maximum number of times the callback command executes.</td>
</tr>
<tr>
<td></td>
<td>Default: 10</td>
</tr>
</tbody>
</table>

#### c. In the Command Arguments related lists, create any command arguments needed for the command.

Create a command argument to allow users to set options when running a command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required. Name of the command argument, for example data.</td>
</tr>
<tr>
<td>Short Name</td>
<td>Short name of the command argument, for example d.</td>
</tr>
<tr>
<td>Data Type</td>
<td>Required. Type of data expected for the command argument. Options include:</td>
</tr>
<tr>
<td></td>
<td>• String: Allows users to enter a string as input.</td>
</tr>
<tr>
<td></td>
<td>• Integer: Allows users to enter a number as input.</td>
</tr>
<tr>
<td></td>
<td>• Boolean: Allows users to enter true or false as input.</td>
</tr>
<tr>
<td></td>
<td>• File Input: Allows users to upload a file as input. Map this argument to a</td>
</tr>
<tr>
<td></td>
<td>Body Parameter Type in the API Endpoint Arguments related list in the next</td>
</tr>
<tr>
<td></td>
<td>step. Users can pass plain text, YAML, JSON, or another file type accepted</td>
</tr>
<tr>
<td></td>
<td>by the REST API. When the user submits a YAML file, the CLI client converts</td>
</tr>
<tr>
<td></td>
<td>to file to JSON format by default. Use the Skip Pre-processing field to</td>
</tr>
<tr>
<td></td>
<td>disable this behavior.</td>
</tr>
<tr>
<td></td>
<td>• Password: Allows users to securely enter a password as input.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> For File Input data types, the file size limit is 10 MB by default. However, you change this limit using the glide.rest.scripted.max_inbound_content_length_mb system property.</td>
</tr>
<tr>
<td>Skip Pre-processing</td>
<td>Set this flag to prevent the CLI client from converting YAML files into JSON format before sending them in the request. When <strong>Data Type</strong> is set to <strong>File Input</strong> and the user submits a YAML file, the CLI client converts the file to JSON before executing the command by default.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Only applies when <strong>Data Type</strong> is set to <strong>File Input</strong>.</td>
</tr>
<tr>
<td>Default Value</td>
<td>Default value to use when the user does not pass a value.</td>
</tr>
<tr>
<td>Short Description</td>
<td>Required. Short description of the command argument. Used as help text when the user runs the <strong>--help</strong> command.</td>
</tr>
<tr>
<td>Application</td>
<td>Read-only application scope for the argument.</td>
</tr>
<tr>
<td>Command</td>
<td>Required. Command that the argument applies to.</td>
</tr>
<tr>
<td>Mandatory</td>
<td>When selected, the user must provide a value for the argument when running the associated command.</td>
</tr>
<tr>
<td>Prompt</td>
<td>Prompt to request information from the user. The CLI prompts for information when the user does not include a required argument in a command.</td>
</tr>
<tr>
<td>Visibility Expression</td>
<td>Expression used to determine whether the CLI should prompt for the argument. Typically used to display an argument based on the value provided to a previous argument.</td>
</tr>
<tr>
<td>Order</td>
<td>Order in which to prompt for the argument.</td>
</tr>
</tbody>
</table>

d. In the API Endpoint Arguments related lists, create any API endpoint arguments needed for the command.

Map command arguments from the Command Arguments related list to parameters in your REST endpoint.
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required. Name of the argument.</td>
</tr>
<tr>
<td>Value</td>
<td>Required. The value from the command that you want to pass to the REST endpoint. You can pass a static value, or an expression such as {flags.table}. Use the flags global variable to access the command arguments.</td>
</tr>
<tr>
<td>Parameter Type</td>
<td>Required. The type of parameter in the REST endpoint that you want to pass the Value to. For example, if you select Body, the CLI passes the value of the Value field to the REST endpoint body. Options include:</td>
</tr>
<tr>
<td></td>
<td>• Body</td>
</tr>
<tr>
<td></td>
<td>• Header</td>
</tr>
<tr>
<td></td>
<td>• Path</td>
</tr>
<tr>
<td></td>
<td>• Query</td>
</tr>
<tr>
<td>Application</td>
<td>Read-only application scope for the argument.</td>
</tr>
<tr>
<td>Command</td>
<td>Required. Command that the argument applies to.</td>
</tr>
</tbody>
</table>

**e. In the Return Values related lists, create any return values needed for the command.**

Create return values to only return certain keys from the response.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Required. Expression representing the path to the key that you want to return.</td>
</tr>
<tr>
<td>Expression</td>
<td></td>
</tr>
<tr>
<td>Alias</td>
<td>Variable name to assign the return value to.</td>
</tr>
<tr>
<td>Application</td>
<td>Read-only application scope for the command.</td>
</tr>
<tr>
<td>Command</td>
<td>Required. Command that the return value applies to.</td>
</tr>
</tbody>
</table>

**Results**

When the user runs the ServiceNow CLI command, the system executes the associated REST API call and returns the result to the ServiceNow CLI.
Manage ServiceNow CLI extensions

Add extensions to the ServiceNow CLI to load additional functionality and commands, update existing extensions, or remove extensions you no longer need.

Procedure

1. Find available extensions.
   a. Open your system’s command-line tool and execute this command.

   $ snc extension list-available --o table

   The system lists available extensions and associated details.

<table>
<thead>
<tr>
<th>NAME</th>
<th>DESCRIPTION</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ui-component</td>
<td>Build and deploy Now Experience Components</td>
<td>19.0.0-alpha.15</td>
</tr>
<tr>
<td>false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. Install an extension.
   a. Open your system’s command-line tool and execute this command.

   $ snc extension add --name <extension-name>

   a. Open your system’s command-line tool and execute this command.

   $ snc extension update --name <extension-name>

4. Optional: Remove an extension.
   a. Open your system’s command-line tool and execute this command.

   $ snc extension remove --name <extension-name>

Creating custom components using the ui-component extension

Develop custom components using the Now® Experience UI Framework and the ui-component extension.

Components are reusable building blocks that you use to create a custom user interface. The Now® Experience Design System comes with a set of customizable components that you can drag into your custom UI. Develop
your own components if you can't find what you need in the Now® Experience Design System library.

To see the Now® Experience Design System library, visit the ServiceNow® Developer Site.

**Benefits of creating custom components**

Developing custom components lets you:

- Personalize a UI according to your agent, customer, and company needs.
- Make your employees more effective and reduce context switching with quick access to important data and information.
- Accommodate your company's unique omni-channel environment using APIs to consolidate your data.

For example, you might want to create a component that displays the cases associated with an SLA, or that tracks the active chats in a particular queue. You can use the Now® Experience UI Framework and the ui-component extension to develop the component you need, and access data from your platform using the Http Effect API. You can also query platform data using GraphQL by creating a custom schema. For more information, see Scripted GraphQL.

**What to know before you begin**

Before you start designing and building your component, make sure you have:

- Some general knowledge of web component concepts, development, and design.
- JavaScript knowledge to define the component behavior.
- Knowledge of Node Package Manager (npm).
- The most recent version of Node.js installed on your local machine. For more information, see, Node.js.
- The ServiceNow CLI installed on your machine.

**Now® Experience UI Framework**

The Now® Experience UI Framework is a JavaScript framework that lets you extend your apps and build web components that are reusable throughout your applications. Using the Now® Experience UI Framework lets you:
• Create a single component to use in multiple places across your applications.
• Encapsulate the component's scope to prevent conflicts with other code.
• Add properties, slots, and actions to your component, allowing users to customize the component every time they use it in a Workspace.

For more information, see the ServiceNow® Developer Site.

ui-component extension and development flow
The ui-component extension is an extension to the ServiceNow CLI that lets you develop custom components using the Now® Experience UI Framework. With the ui-component extension, you can:

• Create the set of files required to develop a component, or project scaffolding.
• Start a local development server to test your component.
• Build a component project and deploy it to a ServiceNow instance.

Application scope
When you deploy a Now® Experience UI Framework component, it deploys into a scoped application on the instance. You can provide an application scope for the component to use as a namespace identifier. Use the namespace identifier guidelines for application development on the instance. For more information, see Application scope.

When reserving an application scope, follow these requirements:

• Maximum: 18 characters.
• Case: snake case.
• Format: x_customerprefix_componentname, where:
  ◦ customerprefix is the value in the glide.appcreator.company.code system property on your instance.
  ◦ componentName is the value provided in the component's name parameter when you created the project.

If you do not provide an application scope when creating your component project, the Now CLI creates one for you.

Alternatively, you can add a value to the scopeName parameter in the now-ui.json file. For more information, see Change a component’s application scope.
Reference guide

To see the Now CLI reference guide, visit the Developer Site.

Set up your environment

Prepare your local environment by installing the Node.js JavaScript runtime environment, Node package manager (npm), and the ui-component extension. These tools enable you to develop and build components locally and deploy them to your instance.

Before you begin

The ui-component extension only supports these operating systems.

- Windows 10
- MacOS Yosemite and later
- Linux CentOS v7.5

Procedure

1. Install Node.js version 12.16.1 or greater. npm automatically installs as part of the Node.js package.
   Node.js is a JavaScript runtime environment that enables you to execute JavaScript locally. npm is the default JavaScript package manager for Node.js. For more information, see Node.js and npmjs.com.

2. Add the ui-component extension to the ServiceNow CLI.

   a. Open your system’s command-line tool and execute this command.

   ```
   $ snc extension add --name ui-component
   ```

3. Verify the installation by running an extension command with the **--help** argument.

   ```
   $ snc ui-component --help
   ```
   The CLI displays a list of available commands. See Get help with ServiceNow CLI.

What to do next

Set up your project.

Related information

- Creating custom components using the ui-component extension
- ServiceNow CLI
Set up your project

Create the component project and the set of files required to develop a component. You can connect to your instance and create an application scope for your component, or you can reserve a scope to verify later.

Before you begin
Set up your environment.

Procedure

1. Create a folder for your project and point your system's command-line tool to the folder.
2. Create the component project and all the files required to build a component.

a. From the folder you created, execute this command.

```bash
$ snc ui-component project [--name name --description description --scope scope --offline]
```

Pass in values for these arguments.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Required. The project name. Must be a valid and unique npm package name.</td>
</tr>
<tr>
<td>description</td>
<td>The project description to be available in the npm registry and the plugins list in your instance.</td>
</tr>
</tbody>
</table>
| scope        | Suggested application scope to assign to this project and its components. If provided, the instance validates the name. Use the namespace identifier guidelines for application development on the instance. For more information, see Application scope. Maximum: 18 characters. Case: snake case. Default: x_customerprefix_componentname, where: 
  - customerprefix is the value in the glide.appcreator.company.code system property on your instance.
  - componentname is the value provided in the component's name parameter when you created the project. |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Alternatively, you can add a value to the <code>scopeName</code> parameter in the <code>now-ui.json</code> file. For more information, see <a href="#">Change a component’s application scope</a>.</td>
</tr>
</tbody>
</table>
| offline | When true, creates and scaffolds a component while disconnected from your instance. Skips validation of the given scope name.  
|        | Default: false.                                                               |

**Example**

```bash
$ snc ui-component project --name @myorg/movie-quotes --description 'A web component that prints movie quotes.'
```

**Example**

```bash
$ snc ui-component project --name @myorg/hello-world --scope x_myorg_helloWorld --offline
```

3. Run the following command to install the npm dependencies.

```bash
npm install
```

**What to do next**

 Develop a component.

**Related information**

[Creating custom components using the ui-component extension](#)

**Use a proxy server with ui-component extension**

Use a proxy server with the Now CLI to define a separate data source for your component. When using a proxy server, the instance forwards Now CLI requests to the proxy server.

**Before you begin**

 Set up your environment.

**Procedure**

1. In your component directory, open the `now-cli.json` file.
2. Update the proxy object.
Update the following parameters with your proxy information.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>development.proxy.headers</td>
<td>Headers to append to your proxied request.</td>
</tr>
<tr>
<td>development.proxy.headers.Authorization</td>
<td>Authentication header to add to the proxied request. Enter your Base 64-encoded username:password here.</td>
</tr>
<tr>
<td>development.proxy.origin</td>
<td>Host to open a proxy with (your instance host address).</td>
</tr>
<tr>
<td>development.proxy.port</td>
<td>The port the proxy runs on.</td>
</tr>
<tr>
<td>development.proxy.proxies</td>
<td>URL glob patterns to pass to the proxy server.</td>
</tr>
</tbody>
</table>

3. Save the file.

**What to do next**

Develop a component.

**Related information**

Creating custom components using the ui-component extension

**Develop a component**

Add your component code and test it using a local development server.

**Before you begin**

- Set up your environment
- Set up your project
About this task
For a tutorial on developing a counter component, visit the Developer Site.

Procedure
1. In your project directory, navigate to src/<component-name>/index.js. This is the primary code file for a component.
2. Add your component code.

Example
This example shows a sample Hello World component.

```javascript
import {createCustomElement} from '@servicenow/ui-core';
import snabbdom from '@servicenow/ui-renderer-snabbdom';
import styles from './styles.scss';

const view = (state, {updateState}) => {
  return (
    <div>Hello World!</div>
  );
};

createCustomElement('hello-world', {
  renderer: {type: snabbdom},
  view,
  styles
});
```

To develop a component for Virtual Agent, add specific properties and actions to interact with the Virtual Agent client interface. The properties required depend on the type of component you are creating. For more information, see Develop a component for Virtual Agent for more information.

3. Optionally add inner components.
An inner component is a component included in another component. For example, a card component might include a separate button component. The button component becomes an inner component that you must import, install, and define in the card component project.

   a. Install the inner component package by executing the install command using your system's command-line tool:

```
  npm install @servicenow/<now-inner-button>
```

   b. Add an import statement to the top of your `index.js` file:
c. Use the inner component in your `index.js` file:

```javascript
<now-inner-button label="Click Me" />
```

d. List the inner component in the `components.[component-name].innerComponents` array in the `now-ui.json` configuration file.

**Example**

```json
{
  "components": {
    "now-chart-timeseries": {
      "innerComponents": [
        "now-inner-button"
      ]
    }
  }
}
```

4. Run the development server command to view your component in a test browser.

```
$ snc ui-component develop [--entry entry --open --port port --host host]
```

Pass in values for these arguments.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| entry | Path to the test module in your component project.  
Default: `example/index.js`. |
| open | Opens the default browser and navigates to the test page.  
Default: false. |
| port | Port where the development server runs.  
Default: 8081. |
| host | Host address to use if you want your local development server to be accessible externally by others. Typically set to `0.0.0.0`. |

**Example**

```
$ snc ui-component develop --entry example/hello.js --open --port 3000
```
What to do next
Deploy a component to an instance.

Related information

Creating custom components using the ui-component extension

Develop a component for Virtual Agent
Create a custom Virtual Agent component to gather input or display information in the Virtual Agent client interface.

Types of Virtual Agent components
To develop a component for Virtual Agent, add specific properties and actions to interact with the Virtual Agent client interface. The properties required depend on the type of component you are creating.

Response component
A response component only provides information to the user, and does not gather input or handle user interaction. For example, a card control that does not require user input and is only in the conversation once.

Add a property to your response component to handle the data sent by the Virtual Agent server.

### Output component properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>controlData</td>
<td>Initial data that the Virtual Agent server sends to your component as the topic runs. Data type: JSON Object</td>
</tr>
</tbody>
</table>

Input component
An input component displays information and/or gathers user input. It includes the same property as the output component to handle data sent by the server, but has more possible states and requires user interaction.

### Input component properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>controlData</td>
<td>Initial data that the Virtual Agent server sends to your component as the topic runs. Data type: JSON Object</td>
</tr>
<tr>
<td>responseValue</td>
<td>Data sent to the component from the user's response, either from the client directly, or from the server if there is a refresh. Only use in components that require user input. Data type: JSON Object</td>
</tr>
<tr>
<td>forceCloseControl</td>
<td>Flag that indicates whether the component can accept input. When true, the control closes and the user cannot interact with it. Monitor changes on the client to update this value. Only use in components that require user input. Data type: Boolean</td>
</tr>
</tbody>
</table>

Use this action to emit data from user interaction.

### Input component actions

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VA_CONTROL#VALUE_SENT</td>
<td>Response data from the client to send to the server. Only use in components that require user input. Data type: JSON Object</td>
</tr>
</tbody>
</table>

### Input component states

Because they accept data, input components must handle multiple states. The state flow is generally as follows:

1. Virtual Agent shows the custom component in the waiting for input state.
2. The user interacts with the component to provide input.
3. The component closes and sends the responseValue property to the server.
4. The server runs the server-side logic and sends the component with the user's
input back to the client.
Waiting for Input
The initial state of a component waiting for user interaction. The
controlData property is set, but the forceControlClosed property is
false. This example shows a slider component in the waiting for input
state.

In this example, if the user has not provided a responseValue and the
control is not closed, the slider and the input button display.
const {controlClosed, sliderVal, sliderMin, sliderMax} = state;
return (<div class={{"slider-chat": true}}>
{responseValue ? null :

<Fragment>

<div class={{"slider-label": true}}>{label}</div>
{controlClosed ? null : <Fragment>
<div class={{"slider-container": true}}>
<input on-change={onSliderChange} type="range"
min={sliderMin} max={sliderMax} value={sliderVal} class={{"slider":
true}} />
<div class={{"slider-value": true}}>
{unitIcon &&
<div class={{"unit-icon": true}}><img
src={unitIcon} /></div>
}
{sliderVal} {unitName}
</div>
<div class={{"button-container":true}}>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.

998


Handling Input

In the previous example, the slider uses a now-button component, which the user clicks to confirm input and send it to the server. When the user clicks the button, the `VA_CONTROL#VALUE_SENT` action fires with the `responseValue` payload.

```javascript
'NOW_BUTTON#CLICKED': (data) => {
    const {updateState, dispatch, state: {sliderVal}} = data;
    updateState({controlClosed: true});
    dispatch('VA_CONTROL#VALUE_SENT', {
        value: {
            sliderVal
        }
    });
}
```

Closed

A component that is closed can no longer accept user input. Components are generally closed because:

- The user responded to the component. The component closes and the conversation continues.
- The user ended the chat. The server does not wait for a response.

For example, the slider component only renders the original prompt when in the closed state.

How many penguins do you want to buy?

Sending response

After the user responds, the control is rendered again on the user's side of the conversation with the value of the `responseValue` property.
For example, the slider control uses this snippet to render the response.

```javascript
{responseValue && <div className="response-container slider-value">
  {unitIcon &&
    <div className="unit-icon"><img src={unitIcon} /></div>
  }
  {returnVal} {unitName}
</div>}
```

Adding the component to Virtual Agent Designer

After developing the component and deploying it to your instance, add it to Virtual Agent Designer using a custom control and definition. For more information, see Virtual Agent custom controls.

Add properties to communicate with Virtual Agent

To develop a component for Virtual Agent, add specific properties and actions to interact with the Virtual Agent client interface. The properties required depend on the type of component you are creating.

Before you begin

- Set up your environment
- Set up your project
- Develop a component
Procedure

1. In your component code, add the Virtual Agent properties to interact with the Virtual Agent client interface.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>controlData</td>
<td>Initial data that the Virtual Agent server sends to your component as the topic runs. Data type: JSON Object</td>
</tr>
<tr>
<td>responseValue</td>
<td>Data sent to the component from the user's response, either from the client directly, or from the server if there is a refresh. Only use in components that require user input. Data type: JSON Object</td>
</tr>
<tr>
<td>forceCloseControl</td>
<td>Flag that indicates whether the component can accept input. When true, the control closes and the user cannot interact with it. Monitor changes on the client to update this value. Only use in components that require user input. Data type: Boolean</td>
</tr>
</tbody>
</table>

2. If creating an input component, use the VA_CONTROL#VALUE_SENT action to send user values to the server.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VA_CONTROL#VALUE_SENT</td>
<td>Data from the client to send to the server. Only use in components that require user input. Data type: JSON Object</td>
</tr>
</tbody>
</table>

What to do next
Test a component for Virtual Agent and Deploy a component to an instance.

Related information
Creating custom components using the ui-component extension

Test a component for Virtual Agent
Test your Virtual Agent custom component before deploying it to your instance.
Before you begin

- Set up your environment
- Set up your project
- Develop a component
- Add properties to communicate with Virtual Agent

About this task
Virtual Agent components must be tested within the Virtual Agent client chat to ensure that the component responds correctly to user input. You can set properties in your component project to test your component in a mock Virtual Agent test client tool.

Procedure

1. Add a dependency on the test client tool to your component project.
   a. Open the `<component-name>/package.json` file in your component project.
   b. Add `"@servicenow/ci-sdk": "1.0.8"` and `"@servicenow/library-translate": "^18.0.0"` to the `devDependencies` object.

   Example
   Here is an example `package.json` file with the correct dependencies.

   ```json
   "dependencies": {
     "@servicenow/ui-renderer-snabbdom": "rome",
     "@servicenow/library-translate": "^18.0.0",
     "@servicenow/now-button": "rome",
     "@servicenow/now-dropdown": "rome",
     "@servicenow/sass-generic": "rome",
     "@servicenow/cli-archetype": "rome",
     "@servicenow/sdk-ci": "1.0.8"
   }
   ```

2. Add sample properties to use as the initial state of the component in your test.
   a. Add a `<component-name>/example/sampleProps.json` file with initial properties to render in your test.

   Example
   Here is an example `sampleProps.json` file with initial properties set for a slider component.

   ```json
   {
     "label": "How many penguins do you want to buy?",
     "defaultValue": 10,
   }
   ```
3. Update the `example.js` file to open the component through the test client tool using the sample properties that you created.

   a. Open the `<component-name>/example/element.js` file in your component project.

   b. Add statements to import `@servicenow/ci-sdk` and the sample properties file you created.

   c. Add the following statement, replacing `<component-name>` with your component's name to create the test tool with initial data from your sample properties.

   ```javascript
   const el = document.createElement('tool-ci-custom-control-tester');
   el.componentTagName = `<component-name>`;
   el.initialExampleData = sampleProps;
   document.body.appendChild(el);
   ```

**Example**

Here is an example `example.js` file that opens the component using the test client tool.

```javascript
import '../src/now-chat-control-slider';
import '@servicenow/sdk-ci';
import sampleProps from './sampleProps.json';

const el = document.createElement('tool-ci-custom-control-tester');
el.componentTagName = `<component-name>`;
el.initialExampleData = sampleProps;
document.body.appendChild(el);
```

4. Run the development server command to view your component in a test browser.

   `$ snc ui-component develop [--entry entry --open --port port --host host]`

Pass in values for these parameters.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| entry | Path to the test module in your component project.  
Default: example/index.js. |
| open | Opens the default browser and navigates to the test page.  
Default: false. |
| port | Port where the development server runs.  
Default: 8081. |
| host | Host address to use if you want your local development server to be accessible externally by others. Typically set to 0.0.0.0 |

**Example**

```
$ snc ui-component develop --entry example/hello.js --open --port 3000
```
The component opens in the test client tool. You can see the initial data provided in the **Custom Control JSON Input Data** field and the component’s output in the **Custom Control Return Data** field.

**What to do next**
If your component is working as expected, deploy it to your instance. See [Deploy a component to an instance](#).

After developing the component and deploying it to your instance, add it to Virtual Agent Designer using a custom control and definition. For more information, see [Virtual Agent custom controls](#).

**Related information**
- [Creating custom components using the ui-component extension](#)

**Add a component to Agent Workspace**
Use custom components to create a custom Workspace interface to fulfill the specific need of your company’s agents.
Communicating with customers through multiple channels can be time consuming. To be efficient in these omni-channel interactions, your agents need a single view of customer information to reduce context switching between multiple tools. By developing custom components for Workspace, your team can bring communications from multiple channels into one interface.

Adding components to a Workspace

Once deployed to your instance, you can add components to Workspace in these ways.

Add a component to a Workspace modal

Use a UI action to launch a custom Now component in a modal so an agent doesn’t have to navigate to a different screen to accomplish a task. For more information, see Render a Now component in a modal.

Add a component to a Workspace landing page using UI Builder

Use the UI Builder to create custom landing pages for your agents. UI Builder is a drag-and-drop tool that lets you visually arrange workspace components. For more information, see Creating custom landing pages for workspaces.

Configure properties in the now-ui.json file to use deployed components in the UI Builder. See Add a component to UI Builder.

Add a component to a Workspace record view

You can add custom or standard components to the component area in the Workspace record view. For more information, see Setting up Record View in workspace.

Add a component to UI Builder

Set properties in a configuration file to add your component to the UI Builder in your instance.

Before you begin

- Set up your environment
- Set up your project
- Develop a component
Procedure

1. In your project directory, open `now-ui.json`.

2. Add the `components.[component-name].uiBuilder` object to the file. This object adds the component to the UI Builder. This object includes these key-value pairs:

<table>
<thead>
<tr>
<th>Key</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>components.[component-name].uiBuilder</td>
<td>Object</td>
<td>Object that adds the component to UI Builder.</td>
</tr>
<tr>
<td>components.[component-name].uiBuilder.label</td>
<td>String</td>
<td>Required. The display name of the component in UI Builder.</td>
</tr>
<tr>
<td>components.[component-name].uiBuilder.icon</td>
<td>String</td>
<td>Required. The name of the icon that appears in UI Builder.</td>
</tr>
<tr>
<td>components.[component-name].uiBuilder.description</td>
<td>String</td>
<td>Required. A short description of the functionality of the component.</td>
</tr>
</tbody>
</table>

Example

```
{
    "components": {
        "card": {
            "uiBuilder": {
                "label": "Card",
                "icon": "chat-fill",
                "description": "A visual card format for a record.",
                "associatedTypes": ["global.core"]
            }
        }
    }
}
```

3. If your component includes properties, add the `components.[component-name].properties` array to the file. This adds the properties as configuration options for the component in UI Builder. This object includes these key-value pairs:
<table>
<thead>
<tr>
<th>Key</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>components. [component-name].properties</td>
<td>Array &lt;Object&gt;</td>
<td>An array of objects that includes all the properties of the component and all relevant information about those properties.</td>
</tr>
<tr>
<td>components. [component-name].properties[].name</td>
<td>String</td>
<td>Name of the property in your component's code.</td>
</tr>
<tr>
<td>components. [component-name].properties[].label</td>
<td>String</td>
<td>Display name of the property to display in UI Builder, if applicable.</td>
</tr>
<tr>
<td>components. [component-name].properties[].description</td>
<td>String</td>
<td>A short description of what the property does or how to use it.</td>
</tr>
<tr>
<td>components. [component-name].properties[].readOnly</td>
<td>Boolean</td>
<td>When true, prevents a user from configuring the property in UI Builder. Default: false.</td>
</tr>
<tr>
<td>components. [component-name].properties[].required</td>
<td>Boolean</td>
<td>When true, the user must configure the property. Default: false.</td>
</tr>
<tr>
<td>components. [component-name].properties[].defaultValue</td>
<td>String</td>
<td>The default value when the user does not provide one.</td>
</tr>
<tr>
<td>components. [component-name].properties[].associatedTypes</td>
<td>Array</td>
<td>Describes where in the UI Builder toolbox the component appears. Value must be &quot;global.core&quot;.</td>
</tr>
<tr>
<td>components. [component-name].properties[].typeMetadata</td>
<td>Object</td>
<td>Extra configuration data that some data types require, such as reference properties and choice lists.</td>
</tr>
</tbody>
</table>
Example

```json
{
    "components": {
        "properties": [
            {
                "name": "backgroundColor",
                "label": "Background Color",
                "description": "Background Color",
                "readOnly": false,
                "required": false,
                "defaultValue": "lightgray"
            },
            {
                "name": "cardType",
                "label": "Type of card",
                "description": "Type of card",
                "readOnly": false,
                "required": false,
                "defaultValue": ""
            }
        ]
    }
}
```

4. Save the file.

What to do next
Deploy a component to an instance.

Related information
Creating custom components using the ui-component extension

Deploy a component to an instance
Deploy your component to display in your instance as an application plugin.

Before you begin
- Set up your environment
- Set up your project
- Develop a component

Procedure
Deploy the component to your instance.
a. Open your system's command-line tool and execute this command.

```bash
$ snc ui-component deploy [--open --force]
```

Pass in values for these arguments.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>open</td>
<td>When true, opens the default browser and navigates to UI Builder in your instance. Default: false</td>
</tr>
<tr>
<td>force</td>
<td>Deploys component changes and overwrites any existing component records. Default: false.</td>
</tr>
</tbody>
</table>

**Example**

```bash
$ snc ui-component deploy --open
```

**What to do next**

Navigate to your instance and add your component to a modal or a landing page in Agent Workspace, or to Virtual Agent. For Agent Workspace, see rendering a Now component in a modal or creating custom landing pages for workspaces. For Virtual Agent, see Virtual Agent custom controls.

**Related information**

- Creating custom components using the ui-component extension

**Change a component’s application scope**

Change a component’s application scope if you encounter scope issues when deploying your component to an instance. For example, you may have set an application scope that is already in use or invalid while creating a component in offline mode.

**Before you begin**

- Set up your environment
- Set up your project
About this task

Procedure

1. In your project directory, open now-ui.json.

2. Update the `scopeName` key with the desired scope name.

   Use the namespace identifier guidelines for application development on the instance. For more information, see Application scope.

   When reserving an application scope, follow these requirements:
   
   - Maximum: 18 characters.
   - Case: snake case.

Example

```json
"scopeName": "x_customerprefix_componentname"
```

3. Deploy your component. See Deploy a component to an instance.

Related information

Creating custom components using the ui-component extension

ServiceNow CLI available commands

Commands and command arguments available to the base system ServiceNow CLI.

Global command arguments

Use command arguments to set options for any CLI commands.

---help

Provides help information for the specified command, which includes the description, supported arguments, and examples. For more information, see Get help with ServiceNow CLI.

```bash
$ snc --help
```

---debug

Provides debug logging output when executing a command. For more information, see Get help with ServiceNow CLI.

```bash
$ snc record delete --table incident --sysid 552c48888c03330964f4932b03eb092 --debug
```
Specifies the named profile to use for a command. For more information, see Configure and manage your ServiceNow CLI connection profiles.

```bash
$ snc configure profile set --profile <profilename>
```

**--output**

Specifies the output format to use for a command. The ServiceNow CLI supports the following output formats.

- **json**: The output is formatted as JSON. This is the default.

```json
{
  "default": {
    "appversion": "1.0.8",
    "host": "https://myinstance.service-now.com",
    "hostversion": "Paris",
    "loginmethod": "basic",
    "output": "json",
    "username": "admin"
  },
  "user1": {
    "appversion": "1.0.8",
    "host": "https://otherinstance.service-now.com",
    "hostversion": "Paris",
    "loginmethod": "basic",
    "output": "yaml",
    "username": "admin"
  }
}
```

- **yaml**: The output is formatted as YAML. Use YAML to handle the output with services and tools that emit or consume YAML-formatted strings.

```
default:
  appversion: 1.0.8
  host: https://myinstance.service-now.com
  hostversion: Paris
  loginmethod: basic
  output: json
  username: admin
user1:
  appversion: 1.0.8
  host: https://otherinstance.service-now.com
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
• **text**: The output is formatted as multiple lines of tab-separated string values. Use this output with traditional UNIX text tools such as grep, sed, and awk, and the text processing performed by PowerShell.

```text
<table>
<thead>
<tr>
<th>default</th>
<th><a href="https://myinstance.service-now.com">https://myinstance.service-now.com</a></th>
<th>Paris</th>
<th>1.0.8</th>
<th>basic</th>
</tr>
</thead>
<tbody>
<tr>
<td>admin</td>
<td>json</td>
<td>admin</td>
<td>json</td>
<td></td>
</tr>
<tr>
<td>user1</td>
<td><a href="https://otherinstance.service-now.com">https://otherinstance.service-now.com</a></td>
<td>Paris</td>
<td>1.0.8</td>
<td>basic</td>
</tr>
<tr>
<td>admin</td>
<td>yaml</td>
<td>admin</td>
<td>yaml</td>
<td></td>
</tr>
</tbody>
</table>
```

• **table**: The output is formatted as a table which presents the information in a human-readable format.

```plaintext
NAME    HOST            HOST       VERSION APP VERSION LOGIN METHOD USERNAME OUTPUT
--------- ---------------- -------------------------- -------------------------- --------------- --------------- --------------- --------------- ---------------
---------- ---------------- -------------------------- -------------------------- --------------- --------------- --------------- ---------------
default  myinstance  Paris   1.0.8   basic   admin   json
user1     otherinstance  Paris   1.0.8   basic   admin   yaml
```

• **none**: The CLI does not print the output to the console. Success, error, and progress messages still display.

```bash
$ snc record query --table incident --query 'active=true' --output json
```

---

**--no-interactive**

Prevents the CLI from prompting the user for argument values. If the user does not pass a value for a required argument, the system uses the default value. If no default value is defined, the system throws an error.

---

**--no-verbose**

The command executes silently without messages. Use this argument in automated testing scenarios.

### Configure profile

Create a connection profile to connect with your instance, view connection profiles, refresh your connection and available commands, or delete profiles you no longer need. For more information about configuring your profile, see Configure and manage your ServiceNow CLI connection profiles.
Set up a default profile

Create a connection profile that the ServiceNow CLI uses by default. You must create a default profile to set up the CLI's initial connection with an instance.

```
$ snc configure profile set
```

The system prompts you for the following information:

<table>
<thead>
<tr>
<th>Requested information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>The host name of the instance to connect to. Supports both the full URL (<a href="https://my-instance.service-now.com">https://my-instance.service-now.com</a>) or just the host name (my-instance).</td>
</tr>
<tr>
<td>Login method</td>
<td>The login method to use to connect to the instance. Supports Basic, OAuth, and OAuth + MFA.</td>
</tr>
<tr>
<td>Username</td>
<td>The user name to use to connect to the instance.</td>
</tr>
<tr>
<td>Password</td>
<td>The password to use to connect to the instance.</td>
</tr>
<tr>
<td>Client id</td>
<td>The client ID to use to connect to the instance when the login method is OAuth or OAuth + MFA.</td>
</tr>
<tr>
<td>Client secret</td>
<td>The client secret to use to connect to the instance when the login method is OAuth or OAuth + MFA.</td>
</tr>
<tr>
<td>Authentication code</td>
<td>The authentication code to use to connect to the instance when the login method is OAuth + MFA.</td>
</tr>
<tr>
<td>Default output format</td>
<td>Specifies how to format the command results. Options are json, yaml, text, and table.</td>
</tr>
</tbody>
</table>

Set up a named profile

Create a named connection profile to use with specific commands. This allows you to specify a different instance or connection protocol for a specific command.
$ snc configure profile set [--profile profile-name]

The system prompts you for the following information:

<table>
<thead>
<tr>
<th>Requested Information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>The host name of the instance to connect to. Supports both the full URL (<a href="https://my-instance.service-now.com">https://my-instance.service-now.com</a>) or just the host name (my-instance).</td>
</tr>
<tr>
<td>Login method</td>
<td>The login method to use to connect to the instance. Supports Basic, OAuth, and OAuth + MFA.</td>
</tr>
<tr>
<td>Username</td>
<td>The user name to use to connect to the instance.</td>
</tr>
<tr>
<td>Password</td>
<td>The password to use to connect to the instance.</td>
</tr>
<tr>
<td>Client id</td>
<td>The client ID to use to connect to the instance when the login method is OAuth or OAuth + MFA.</td>
</tr>
<tr>
<td>Client secret</td>
<td>The client secret to use to connect to the instance when the login method is OAuth or OAuth + MFA.</td>
</tr>
<tr>
<td>Authentication code</td>
<td>The authentication code to use to connect to the instance when the login method is OAuth + MFA.</td>
</tr>
<tr>
<td>Default output format</td>
<td>Specifies how to format the command results. Options are json, yaml, text, and table.</td>
</tr>
</tbody>
</table>

View profiles

View all connection profiles set in the configuration file, or view information about a specific profile.

$ snc configure profile list [--profile profile-name]

Remove a profile

Remove a named connection profile that you no longer need from the configuration file.

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
$ snc configure profile remove [--profile profile-name]

**Refresh a profile**

Update the available commands from the instance for the given profile. Refresh your connection after modifying any of the commands on the corresponding instance in order to keep the CLI up-to-date.

$ snc configure profile refresh [--profile profile-name]

**Perform record operations**

Create, read, update, delete, and query records in your instance using the ServiceNow CLI command-line tool. For more information about performing record operations, see [Perform record operations using ServiceNow CLI](#).

**Create a record**

Inserts a single record in a specified table.

$ snc record create [--table table --data data]

Pass in values for these arguments.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>Required. Name of the table in which to save the record.</td>
</tr>
<tr>
<td>data</td>
<td>Required. Field name and the associated value for each field to define in the specified record in JSON string format.</td>
</tr>
</tbody>
</table>

**Delete a record**

Deletes the specified record from the specified table.

$ snc record delete [--table table --sysid sys_id]

Pass in values for these arguments.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>Required. Name of the table in which to delete the record.</td>
</tr>
<tr>
<td>sysid</td>
<td>Required. Sys_id of the record to delete.</td>
</tr>
</tbody>
</table>
Get a record

Retrieves a single record based on the specified sys_id from the specified table.

$ snc record get [--table table --sysid sys_id]

Pass in values for these arguments.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>Required. Name of the table from which to retrieve the record.</td>
</tr>
<tr>
<td>sysid</td>
<td>Required. Sys_id of the record to retrieve.</td>
</tr>
</tbody>
</table>

Query records

Retrieves multiple records from a specified table.

$ snc record query [--displayvalue displayValue --fields fields --limit limit --offset offset --query query --table table]

Pass in values for these arguments.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>displayValue</td>
<td>Include --displayvalue displayValue to retrieve the display value from the database for reference and choice fields. Do not include this parameter to retrieve the actual values.</td>
</tr>
<tr>
<td>fields</td>
<td>Comma-separated list of field names to return from the database.</td>
</tr>
<tr>
<td>limit</td>
<td>Maximum number of records to return.</td>
</tr>
<tr>
<td>offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval.</td>
</tr>
<tr>
<td>query</td>
<td>Required. Encoded query used to filter the result set in the following format: --query '&lt;column_name&gt;&lt;operator&gt;&lt;value&gt;'.</td>
</tr>
<tr>
<td>table</td>
<td>Required. Name of the table in which to query the records.</td>
</tr>
</tbody>
</table>

Update a record
Updates the specified record with the given data attributes.

```bash
$snc record update [--sysid sys_id --table table --data data]
```

Pass in values for these arguments.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>Required. Name of the table in which to save the record.</td>
</tr>
<tr>
<td>sysid</td>
<td>Required. Sys_id of the record to update.</td>
</tr>
<tr>
<td>data</td>
<td>Required. Field name and the associated value for each field to define in the specified record in JSON string format.</td>
</tr>
</tbody>
</table>

**Work with extensions**

Add extensions to the ServiceNow CLI to load additional functionality and commands, update existing extensions, or remove extensions you no longer need. For more information, see Manage ServiceNow CLI extensions.

**Find available extensions**

```bash
$snc extension list-available -o table
```

**Install an extension**

```bash
$snc extension add --name <extension-name>
```

**Update an extension**

```bash
$snc extension update --name <extension-name>
```

**Remove an extension**

```bash
$snc extension remove --name <extension-name>
```

**Use the ui-component extension**

**Add the ui-component extension**

Add the ui-component extension to the ServiceNow CLI.

```bash
$snc extension add --name ui-component
```

**Set up your project**

Create the component project and the set of files required to develop a component. You can connect to your instance and
create an application scope for your component, or you can reserve a scope to verify later.

```
$ snc ui-component project [--name name --description description --scope scope --offline]
```

Pass in values for these arguments.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Required. The project name. Must be a valid and unique npm package name.</td>
</tr>
<tr>
<td>description</td>
<td>The project description to be available in the npm registry and the plugins list in your instance.</td>
</tr>
</tbody>
</table>
| scope   | Suggested application scope to assign to this project and its components. If provided, the instance validates the name. Use the namespace identifier guidelines for application development on the instance. For more information, see Application scope. Maximum: 18 characters. Case: snake case. Default: $customerprefix$componentname, where:
  - $customerprefix$ is the value in the glide.appcreator.company.code system property on your instance.
  - $componentname$ is the value provided in the component's name parameter when you created the project. Alternatively, you can add a value to the scopeName parameter in the now-ui.json file. For more information, see Change a component's application scope. |
| offline | When true, creates and scaffolds a component while disconnected from your instance. Skips validation of the given scope name. Default: false. |

Run the development server
Add your component code and test it using a local development server.

$ snc ui-component develop [--entry entry --open --port port --host host]

Pass in values for these arguments.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entry</td>
<td>Path to the test module in your component project. Default: example/index.js.</td>
</tr>
<tr>
<td>open</td>
<td>Opens the default browser and navigates to the test page. Default: false.</td>
</tr>
<tr>
<td>port</td>
<td>Port where the development server runs. Default: 8081.</td>
</tr>
<tr>
<td>host</td>
<td>Host address to use if you want your local development server to be accessible externally by others. Typically set to 0.0.0.0.</td>
</tr>
</tbody>
</table>

Deploy a component to an instance

Deploy your component to display in your instance as an application plugin.

$ snc ui-component deploy [--open --force]

Pass in values for these arguments.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>open</td>
<td>When true, opens the default browser and navigates to UI Builder in your instance. Default: false</td>
</tr>
<tr>
<td>force</td>
<td>Deploys component changes and overwrites any existing component records. Default: false.</td>
</tr>
</tbody>
</table>

Commands installed with CMDB Application CLI and API

Commands and command groups available to the ServiceNow CLI when the app-cmdb-api-cli plugin is installed.
Request apps on the Store

Visit the ServiceNow Store website to view all the available apps and for information about submitting requests to the store. For cumulative release notes information for all released apps, see the ServiceNow Store version history release notes.

To script critical operations which support automation across the enterprise, you can leverage APIs or run command line operations that the CMDB Application CLI and API store app provide instead of using the user interface. The CMDB Application CLI and API store app provides a robust framework which consolidates all the APIs that are related to application services and the command lines that let you access the interface to those APIs.

CMDB Application CLI and API commands enable the following tasks:

• Registering and creating an application service and establishing upstream relationships
• Getting details of a given application service and its upstream relationships
• Connecting higher level constructs such as business applications and business service offerings
• Populating an application service with a given population type
• Changing the state of an application service

For the REST API solution, see SG Services API.

Convert Application Service

Converts a manual or empty type application service to a calculated application service. During conversion, the application service record moves into the [cmdb_ci_service_calculated] table with the newly assigned class.

Command group:
• Parent group: service-graph
• Child group: app-service

Roles required

app_service_admin

If using a service mapping related service, the user must have the required roles for that service mapping related service.

Command structure
snc service-graph app-service convert --data '{JSON}'

**Arguments**

This command passes a JSON object using the `data` parameter.

The following properties for identifying a CI take precedence as follows:

1. **sys_id** – If `sys_id`, the system only uses the `sys_id` and ignores any additional values.
2. **number** – If provided without the `sys_id`, the system only uses the number and ignores any additional values.
3. **<IRE field name>** – The system only uses these values if the `sys_id` or `number` are not provided.

**JSON object properties**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;IRE field name&gt;</td>
<td>One or more IRE fields identifying the application service. For example, name or version.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>levels</td>
<td>Number of levels to include in the conversion.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example**

```
snc service-graph app-service convert --data '{"name":"Test Register","environment":"Test","version":"1.0","levels":8}'
```

**Return value**

```
{
  "result": {
    "status": "success"
  }
}
```
Create Application Service Relationship

Constructs upstream relations such as business applications, business service offerings, and other application services. Running this command creates a relationship, taking input with a single parent and a corresponding child object.

Command group:
- Parent group: service-graph
- Child group: app-service

Roles required
app_service_admin

If using a service mapping related service, the user must have the required roles for that service mapping related service.

Command structure

```
snc service-graph app-service create-relationship --data '{JSON}'
```

Arguments

This command passes a JSON object using the `data` parameter.

The following properties for identifying a CI take precedence as follows:

1. sys_id – If sys_id, the system only uses the sys_id and ignores any additional values.
2. number – If provided without the sys_id, the system only uses the number and ignores any additional values.
3. <IRE field name> – The system only uses these values if the sys_id or number are not provided.

JSON object properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>child</td>
<td>Information identifying the child application service with which to create a relationship. The child is located in the Application Service [cmdb_ci_service_auto] table. A dynamic CI group can be added as a child but cannot be parent. Data type: Object</td>
</tr>
</tbody>
</table>
### JSON object properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;child&quot;: {&lt;IRE field name&gt;: &quot;String&quot;, number: &quot;String&quot;, sys_id: &quot;String&quot; }</td>
<td>One or more IRE fields identifying the child application service. For example, name or version. Data type: String</td>
</tr>
<tr>
<td>child.&lt;IRE field name&gt;</td>
<td>One or more IRE fields identifying the child application service. For example, name or version. Data type: String</td>
</tr>
<tr>
<td>child.number</td>
<td>Unique number that identifies the child application service. Data type: String</td>
</tr>
<tr>
<td>child.sys_id</td>
<td>Sys_id of the child application service listed in the Application Service [cmdb_ci_service_auto]. Data type: String</td>
</tr>
<tr>
<td>parent</td>
<td>Details identifying the parent application service with which to create a relationship. Data type: Object</td>
</tr>
<tr>
<td>&quot;parent&quot;: {&lt;IRE field name&gt;: &quot;String&quot;, number: &quot;String&quot;, sys_id: &quot;String&quot;, class_name: &quot;String&quot; }</td>
<td>One or more IRE fields identifying the application service. For example, name or version. Data type: String</td>
</tr>
<tr>
<td>parent.&lt;IRE field name&gt;</td>
<td>One or more IRE fields identifying the application service. For example, name or version. Data type: String</td>
</tr>
<tr>
<td>parent.number</td>
<td>Unique number that identifies the application service. Data type: String</td>
</tr>
</tbody>
</table>
JSON object properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent.sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto]. Data type: String</td>
</tr>
</tbody>
</table>
| parent.class_name     | Name of the class that contains the application service. The parent class name should be from one of the following tables:  
  • cmdb_ci_service_auto  
  • cmdb_ci_service_discovered  
  • cmdb_ci_service_by_tags  
  • cmdb_ci_service_calculated  
  • service_offering  
  • cmdb_ci_business_app  
  Default: cmdb_ci_service_auto  
  Data type: String |

Example

snc service-graph app-service create-relationship --data
'
  "child":{"name":"wdfsdf","environment":"Test","version":"1.0"},"parent":{
  "sys_id":"abcdefg","name":"business App1","class_name":"service_offering"}}'

Return value

```json
{
  "result": {
    "status": "success"
  }
}
```

Delete Application Service

Deletes an application service.

Command group:
• Parent group: service-graph
• Child group: app-service

Roles required
app_service_admin

If using a service mapping related service, the user must have the required roles for that service mapping related service.

Command structure
```
snc service-graph app-service delete --data '{JSON}'
```

Arguments
This command passes a JSON object using the `data` parameter.
The following properties for identifying a CI take precedence as follows:
1. `sys_id` – If `sys_id`, the system only uses the `sys_id` and ignores any additional values.
2. `number` – If provided without the `sys_id`, the system only uses the number and ignores any additional values.
3. `<IRE field name>` – The system only uses these values if the `sys_id` or `number` are not provided.

JSON object properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;IRE field name&gt;</code></td>
<td>One or more IRE fields identifying the application service. For example, name or version. Data type: String</td>
</tr>
<tr>
<td><code>number</code></td>
<td>Unique number that identifies the application service. Data type: String</td>
</tr>
<tr>
<td><code>sys_id</code></td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table. Data type: String</td>
</tr>
</tbody>
</table>

Example
snc service-graph app-service delete --data '{"name":"Test Register","environment":"Test","version":"1.0"}'

Return value

```json
{
    "result": {
        "status": "success"
    }
}
```

Delete Application Service Relationship

Deletes an application service upstream relationship.

Command group:
- Parent group: service-graph
- Child group: app-service

Roles required

app_service_admin

If using a service mapping related service, the user must have the required roles for that service mapping related service.

Command structure

```
snc service-graph app-service delete-relationship --data '{JSON}'
```

Arguments

This command passes a JSON object using the `data` parameter.

The following properties for identifying a CI take precedence as follows:

1. `sys_id` – If `sys_id`, the system only uses the `sys_id` and ignores any additional values.
2. `number` – If provided without the `sys_id`, the system only uses the `number` and ignores any additional values.
3. `<IRE field name>` – The system only uses these values if the `sys_id` or `number` are not provided.
## JSON object properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type:</th>
</tr>
</thead>
<tbody>
<tr>
<td>child</td>
<td>Information describing the child relationship to be deleted from the service application.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>child.&lt;IRE field name&gt;</td>
<td>One or more IRE fields identifying the child application service. For example, name or version.</td>
<td>String</td>
</tr>
<tr>
<td>child.number</td>
<td>Unique number that identifies the child application service.</td>
<td>String</td>
</tr>
<tr>
<td>child.sys_id</td>
<td>Sys_id of the child application service listed in the Application Service [cmdb_ci_service_auto].</td>
<td>String</td>
</tr>
<tr>
<td>parent</td>
<td>Details identifying the parent application service from which to remove a relationship.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>parent.&lt;IRE field name&gt;</td>
<td>One or more IRE fields identifying the application service. For example, name or version.</td>
<td>String</td>
</tr>
</tbody>
</table>
### JSON object properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent.number</td>
<td>Unique number that identifies the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>parent.sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>parent.class_name</td>
<td>Name of the class that contains the application service.</td>
</tr>
<tr>
<td></td>
<td>The parent class name should be from one of the following tables:</td>
</tr>
<tr>
<td></td>
<td>• cmdb_ci_service_auto</td>
</tr>
<tr>
<td></td>
<td>• cmdb_ci_service_discovered</td>
</tr>
<tr>
<td></td>
<td>• cmdb_ci_service_by_tags</td>
</tr>
<tr>
<td></td>
<td>• cmdb_ci_service_calculated</td>
</tr>
<tr>
<td></td>
<td>• service_offering</td>
</tr>
<tr>
<td></td>
<td>• cmdb_ci_business_app</td>
</tr>
<tr>
<td></td>
<td>Default: cmdb_ci_service_auto</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Example

```bash
snc service-graph app-service delete-relationship --data
'"child":{"name":"Test Register","environment":"Test","version":"1.0"},"parent":{"name":"business Service Offering1","class_name":"service_offering"}"
```

### Return value

```json
{
  "result": {
    "status": "success"
  }
}
```
Find Application Service
Finds the details of a given application service and its upstream relationships.

Command group:
- Parent group: service-graph
- Child group: app-service

Roles required
- app_service_admin – This role provides unlimited viewing of application services.
- app_service_user – This role only provides viewing application services in Operational status.

If using a service mapping related service, the user must have the required roles for that service mapping related service.

Command structure
```
snc service-graph app-service find --data '{JSON}'
```

Arguments
This command passes a JSON object using the `data` parameter.
The following properties for identifying a CI take precedence as follows:
1. sys_id – If sys_id, the system only uses the sys_id and ignores any additional values.
2. number – If provided without the sys_id, the system only uses the number and ignores any additional values.
3. `<IRE field name>` – The system only uses these values if the sys_id or number are not provided.

JSON object properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;IRE field name&gt;</code></td>
<td>One or more IRE fields identifying the application service. For example, name or version. Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service. Data type: String</td>
</tr>
</tbody>
</table>
JSON object properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| sys_id   | Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table.  
Data type: String                               |

**Example**

```bash
snc service-graph app-service find --data '{"name" : "Test App Service1"}'
```

**Return value**

```json
{
  "result": {
    "aliases": null,
    "asset": null,
    "asset_tag": null,
    "assigned": "",
    "assigned_to": null,
    "assignment_group": null,
    "attestation_score": null,
    "attested": "0",
    "attested_by": null,
    "attested_date": "",
    "attributes": null,
    "bucket": null,
    "business_contact": null,
    "business_need": null,
    "business_relation_manager": null,
    "business_unit": null,
    "busines_criticality": "4 - not critical",
    "can_print": "0",
    "category": null,
    "change_control": null,
    "checked_in": "",
    "checked_out": "",
    "checkout": null,
    "comments": null,
    "company": null,
    "compatibility_dependencies": null,
    "consumer_type": "internal",
    "correlation_id": null,
    "cost": null,
```
Populate Application Service

Populates an application service with a service population method.

Command group:

- Parent group: service-graph
- Child group: app-service

Roles required

- app_service_admin

If using a service mapping related service, the user must have the required roles for that service mapping related service.

Command structure

```
snc service-graph app-service populate --data '{JSON}'
```

Arguments

This command passes a JSON object using the data parameter. The following properties for identifying a CI take precedence as follows:
1. `sys_id` – If `sys_id`, the system only uses the `sys_id` and ignores any additional values.

2. `number` – If provided without the `sys_id`, the system only uses the number and ignores any additional values.

3. `<IRE field name>` – The system only uses these values if the `sys_id` or `number` are not provided.

### JSON object properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;IRE field name&gt;</code></td>
<td>One or more IRE fields identifying the application service. For example, name or version.</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>number</code></td>
<td>Unique number that identifies the application service.</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>population_method</code></td>
<td>Required. Identifies the population method and its accompanying property to identify the content for population. Only one accompanying object is valid per type.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>population_method.group_id</code></td>
<td>Group ID of the CMDB group configured with the <code>cmdb_group</code> population type.</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>population_method.levels</code></td>
<td>Number of levels to use in building the service. If the level value is</td>
<td></td>
</tr>
</tbody>
</table>
### JSON object properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>not provided, the system checks the sys_property for the value. If svc.manual.convert.levels.default_value is not populated, a default value of 3 is used. <strong>Data type:</strong> Number</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Number</td>
</tr>
<tr>
<td>population_method.service_candidate</td>
<td>Unique identifier of the service candidate. <strong>Data type:</strong> String</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> String</td>
</tr>
<tr>
<td>population_method.service_relations</td>
<td>List of objects containing hierarchy data for the CIs within the application service. All CIs form pairs with a parent and child CI. The top-level CI, referred to as the entry point of an application service, does not have a parent CI. <strong>Data type:</strong> Array</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Array</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### JSON object properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;parent&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>&quot;type&quot;: &quot;service_hierarchy&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
</tbody>
</table>

Associated population type: `service_hierarchy`

<table>
<thead>
<tr>
<th>population_method.service_relations.child</th>
<th>Name of a child CI related to the CI.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>population_method.service_relations.parent</th>
<th>Name of a parent CI related to the CI.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>population_method.tags</th>
<th>List of objects containing tags to associate with the CI. This information is located in the Key Values [cmdb_key_value} table.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
</tbody>
</table>

```json
"population_method": {
    "tags": [
        {
            "tag": "String",
            "value": "String"
        }
    ],
    "type": "tag_list"
}
```

Associated population type: `tag_list`

<table>
<thead>
<tr>
<th>population_method.tags.tag</th>
<th>Tag name.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>population_method.tags.value</th>
<th>Tag value.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>
## Register Application Service

Creates an application service, tags and constructs upstream relationships such as business applications, business service offerings, and other application services.

**Command group:**
- Parent group: `service-graph`
- Child group: `app-service`

### Example

```bash
snc service-graph app-service populate --data '{"name":"Test Register","environment":"Test","version":"1.0","population_method":{"group_id":"String","type":"cmdb_group"}}'
```

### Return value

```json
{
  "result": {
    "status": "success"
  }
}
```
Roles required

app_service_admin

If using a service mapping related service, the user must have the required roles for that service mapping related service.

Command structure

```
snc service-graph app-service register --data '{JSON}'
```

Arguments

This command passes a JSON object using the `data` parameter.

The following properties for identifying a CI take precedence as follows:

1. `sys_id` – If `sys_id`, the system only uses the `sys_id` and ignores any additional values.
2. `number` – If provided without the `sys_id`, the system only uses the number and ignores any additional values.
3. `<IRE field name>` – The system only uses these values if the `sys_id` or `number` are not provided.

JSON object properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;IRE field name&gt;</code></td>
<td>One or more IRE fields identifying the application service. For example, name or version.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>relationships</td>
<td>Upstream relationships categorized by type.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### JSON object properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| "business_service_offering"  | ```json
  : [Array],
  "parent_app_service": [Array],
  "technical_service_offering": [Array]
} ```                                                                                                                                                      |
| Maximum number of relationships is 25.                                                                                                                 |
| relationships.business_app    | List of objects representing Business Application relationship types. These values can be defined using one of the following items as key-value pairs.  
  - `<IRE field name>`  
  - number  
  - sys_id  
  Data type: Array                                                                                                                                         |
| relationships.business_service_offering | List of objects representing Business Service Offering relationship types. These values can be defined using the following items as key-value pairs.  
  - `<IRE field name>`  
  - number  
  - sys_id  
  Data type: Array                                                                                                                                         |
| relationships.parent_app_service | List of objects representing Application relationship types. These values can be defined using the following items as key-value pairs.  
  - `<IRE field name>`  
  - number  
  - sys_id  
  Data type: Array                                                                                                                                         |
### JSON object properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>service relationship types.</td>
<td>Service relationship types. These values can be defined using the following items as key-value pairs.</td>
</tr>
<tr>
<td></td>
<td>- &lt;IRE field name&gt;</td>
</tr>
<tr>
<td></td>
<td>- number</td>
</tr>
<tr>
<td></td>
<td>- sys_id</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>relationships.technical_service_offering</td>
<td>List of objects representing Technical Service Offering relationship types. These values can be defined using the following items as key-value pairs.</td>
</tr>
<tr>
<td></td>
<td>- &lt;IRE field name&gt;</td>
</tr>
<tr>
<td></td>
<td>- number</td>
</tr>
<tr>
<td></td>
<td>- sys_id</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>tags</td>
<td>List of objects containing tag definitions as key-value pairs.</td>
</tr>
<tr>
<td></td>
<td>&quot;tags&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;key&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
JSON object properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tags.key</td>
<td>Tag category name.</td>
</tr>
<tr>
<td>tags.value</td>
<td>Tag value.</td>
</tr>
</tbody>
</table>

Data type: Array

Example

```
snc service-graph app-service register --data '{"name": "Test Register", "environment": "Test", "version": "1.0", "number": "SNSVC0001014", "relationships": {"business_application": [{"sys_id": "0250a9400697410f87713b656474250"}, {"number": "APM0001002"}, {"name": "Test Biz App1"}], "business_service_offering": [{"sys_id": "ed32e9800697410f87713b656474259"}], "technical_service_offering": [{"sys_id": "80e12d600697410f87713b65647421c"}, {"number": "BSN0001005"}, {"name": "Tech Service Offering2"}], "parent_app_service": [{"sys_id": "a2f061800697410f87713b656474255"}], "tags": [{"key": "key1", "value": "value1"}, {"key": "key2", "value": "value2"}]}}'
```

Return value

```
{
  "result": {
    "app_service": {
      "sys_id": "99b2a5400697410f87713b6564742ad",
      "name": "Test Register",
      "number": "SNSVC0001014"
    },
    "message": "Service registered successfully",
    "status": "INSERT"
  }
}
```

Update Application Service

Updates an existing application service provided and creates tags for the given application service.
Command group:
- Parent group: service-graph
- Child group: app-service

Roles required

app_service_admin

If using a service mapping related service, the user must have the required roles for that service mapping related service.

Command structure

`snc service-graph app-service update --data '{JSON}'`

Arguments

This command passes a JSON object using the `data` parameter. The following properties for identifying a CI take precedence as follows:

1. sys_id – If sys_id, the system only uses the sys_id and ignores any additional values.
2. number – If provided without the sys_id, the system only uses the number and ignores any additional values.
3. `<IRE field name>` – The system only uses these values if the sys_id or number are not provided.

JSON object properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;fields or tags to update&gt;</td>
<td>Use key-value pairs to identify each field or tag to be updated. Only basic information can be updated, no upstream relationships can be updated. Data type: String</td>
</tr>
<tr>
<td>&lt;IRE field name&gt;</td>
<td>One or more IRE fields identifying the application service. For example, name or version. Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service.</td>
</tr>
</tbody>
</table>
### JSON object properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto]. Data type: String</td>
</tr>
</tbody>
</table>

### Example

```
snc service-graph app-service update --data '{"name": "Test Register", "version": "2.0"}'
```

### Return value

```
{
"result": {
  "sys_id": "99b2a54040697410f87713b6564742ad",
  "name": "Test Register",
  "number": "SNSVC0001014",
  "version": "2.0"
}
}
```

### Update Application Service State

Changes the application service lifecycle state to activate, deactivate, or retire.

**Command group:**

- Parent group: service-graph
- Child group: app-service

**Roles required**

app_service_admin

If using a service mapping related service, the user must have the required roles for that service mapping related service.

**Command structure**

```
snc service-graph app-service update-state --data '{(JSON)}'
```

**Arguments**

This command passes a JSON object using the `data` parameter.
The following properties for identifying a CI take precedence as follows:

1. sys_id – If sys_id, the system only uses the sys_id and ignores any additional values.

2. number – If provided without the sys_id, the system only uses the number and ignores any additional values.

3. <IRE field name> – The system only uses these values if the sys_id or number are not provided.

### JSON object properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;IRE field name&gt;</td>
<td>One or more IRE fields identifying the application service. For example, name or version.</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service.</td>
</tr>
<tr>
<td>state</td>
<td>Required. Lifecycle state of the application service. These values are updated in the Application Services [cmdb_ci_service_auto] table. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• ACTIVATE – Life cycle is operational and in use.</td>
</tr>
<tr>
<td></td>
<td>◦ operational_status=Operational</td>
</tr>
<tr>
<td></td>
<td>◦ life_cycle_stage=Operational</td>
</tr>
<tr>
<td></td>
<td>◦ life_cycle_stage_status=In Use</td>
</tr>
<tr>
<td></td>
<td>• DEACTIVATE – Life cycle is not operational and is in the design stage.</td>
</tr>
<tr>
<td></td>
<td>◦ operational_status=Non-Operational</td>
</tr>
<tr>
<td></td>
<td>◦ life_cycle_stage=Design</td>
</tr>
<tr>
<td></td>
<td>◦ life_cycle_stage_status=Build</td>
</tr>
<tr>
<td></td>
<td>• RETIRE – End of life.</td>
</tr>
<tr>
<td></td>
<td>◦ operational_status=Retired</td>
</tr>
<tr>
<td></td>
<td>◦ life_cycle_stage=End Of Life</td>
</tr>
<tr>
<td></td>
<td>◦ life_cycle_stage_status=Retired</td>
</tr>
</tbody>
</table>
JSON object properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto]. Data type: String</td>
</tr>
</tbody>
</table>

Example

```
snc service-graph app-service update-state --data '{"name":"Test Register","environment":"Test","version":"1.0"}'
```

Return value

```
{
    "result": {
        "status": "success"
    }
}
```

ServiceNow Extensions for Visual Studio Code

Edit your ServiceNow applications in Visual Studio Code with the help of the ServiceNow Extensions for VS Code.

The ServiceNow Extensions for the VS Code editor enables you to edit applications within your ServiceNow instance.
Advantages of using Visual Studio Code

**Edit your applications offline**

Use Visual Studio Code to download and edit a local copy of your application. You can edit the application offline, and then synchronize again when your instance is available.

**Use Visual Studio Code JavaScript features to reduce development time and improve code quality.**

Visual Studio Code IntelliSense includes editing features such as code completion, code suggestions, and quick information. Use these tools to complete your coding tasks quickly and reduce errors. For more details on using IntelliSense in your applications, see [IntelliSense in VS Code](#). The extension also supports Linting using ESLint. The same standard ESLint rules used in the ServiceNow instance are available.

**Functions of ServiceNow Extensions for VS Code**

The ServiceNow Extensions for VS Code extension includes tools for developing on the Now Platform.

The extension adds several functions to your Visual Studio Code (VS Code) implementation. You access these functions through the command palette.
## Available functions in the ServiceNow extension for VS Code

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setup Now Workspace</td>
<td>Create the project folder to work with ServiceNow applications. For details on creating this workspace, see Set up a workspace in VS Code.</td>
</tr>
<tr>
<td>Activate Now Extension</td>
<td>Activates the ServiceNow extension. See Activate ServiceNow Extensions for VS Code.</td>
</tr>
<tr>
<td>Create Now Project</td>
<td>Import existing ServiceNow applications to a Visual Studio Code project. See Create a project in VS Code.</td>
</tr>
<tr>
<td>Sync Current Project</td>
<td>During development, synchronize all files between VS Code and the instance. See Synchronize the current project between a Visual Studio Code workspace and a ServiceNow instance.</td>
</tr>
<tr>
<td>Sync Current File</td>
<td>Synchronize the current file you are working on. See Sync the current file between a Visual Studio Code</td>
</tr>
</tbody>
</table>
### Available functions in the ServiceNow extension for VS Code (continued)

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>workspace and a ServiceNow instance.</td>
<td></td>
</tr>
<tr>
<td>Reset Project</td>
<td>Sets project to its original state. See Reset a project in Visual Studio Code.</td>
</tr>
<tr>
<td>Configure File Types</td>
<td>Modify the options selected in the metadata. Metadata determines which elements of your application you want to synchronize with VS Code. See Import an application into Visual Studio Code.</td>
</tr>
<tr>
<td>Add Custom File Types</td>
<td>Add file types to your project using the Custom File Types wizard. See Add custom file types in Visual Studio Code.</td>
</tr>
<tr>
<td>Select Application</td>
<td>Switch between application within the workspace in VS code. See Import an application into Visual Studio Code.</td>
</tr>
<tr>
<td>Select Update Set</td>
<td>Select an update set. Changes synchronized to your instance are applied to the selected update set. For more information on update sets, see Import an application into Visual Studio Code.</td>
</tr>
<tr>
<td>Create New File</td>
<td>Create records in your application. For details, see Create a file in VS Code.</td>
</tr>
<tr>
<td>Global File Search</td>
<td>Find files within the instance. See Search files on your instance in VS Code.</td>
</tr>
<tr>
<td>Run Background Scripts</td>
<td>Run a background script on your instance. See Run background scripts using VS Code.</td>
</tr>
</tbody>
</table>
Available functions in the ServiceNow extension for VS Code (continued)

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IntelliSense</td>
<td>An IntelliSense code-completion aid available for Glide APIs. See IntelliSense in VS Code.</td>
</tr>
</tbody>
</table>

Install ServiceNow Extensions for VS Code

Download and install ServiceNow extensions to begin editing your applications in VS Code.

Before you begin
Role required: none
In order to install ServiceNow Extensions for Visual Studio Code, you must have VS Code version 1.38.0 or later installed.

Procedure

1. Open VS Code and navigate to the Extensions tab by clicking the Extensions tab icon (️) or pressing Control+Shift+X on Windows or Command+Shift+X on MacOS.
3. Select ServiceNow Extensions for VS Code in the search results and click Install to proceed with installation.
5. Navigate to View > Command Palette in Visual Studio Code. You can also use a keyboard shortcut (Control+Shift+P on Windows and Command+Shift+P on MacOS) to open the command palette.
6. Choose Developer: Reload Window from the list.

What to do next
Once you reload VS Code activate the ServiceNow extension.

Activate ServiceNow Extensions for VS Code

Activate the ServiceNow Extensions for VS Code to be able to edit applications within your ServiceNow instance.
Before you begin
Install ServiceNow Extensions for VS Code to activate the extension and begin editing your applications in VS code.

About this task
Activating the extension is the first step you must perform after installation of the extension to access its functionalities.

Procedure
1. Navigate to View > Command Palette to open the command palette. You can also use a keyboard shortcut, Control+Shift+P on Windows or Command+Shift+P on MacOS, to open the command palette.
2. Choose Now: Activate Now Extension from the list.

The Setup Workspace link appears at the bottom of the VS Code IDE. After the extension is activated, Set up your workspace in VS Code to begin editing ServiceNow applications.

Set up a workspace in VS Code
Using the ServiceNow Extensions for VS Code, create a project work folder to use as a workspace for ServiceNow applications.

Before you begin
Activate the workspace to access the functionalities of ServiceNow Extensions for VS Code. Role required: admin
Procedure

1. Click the **Setup Workspace** on the status bar at the bottom of the VS Code IDE.
   You can also use a keyboard shortcut, Control+Shift+P on Windows or Command+Shift+P on MacOS, to open the command palette and choose **Now: Setup Now Workspace** from the list.

   ![Setup Now Workspace command](image)

2. Do any of the following actions:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
</table>
   | To set up a workspace from a new folder | a. Click **New Folder**.  
b. Enter a name for the folder in the **Name of new Folder** window.  
c. Click **Create**. |

   The workspace is created from the selected folder.

**What to do next**
Create a project in your workspace. You can create multiple projects of different project types within a workspace.
Create a project in VS Code

ServiceNow applications are contained within a project in the VS Code IDE. Import an application from your ServiceNow instance with the help of ServiceNow Extensions for VS Code.

Before you begin
Setup your workspace to create a project and start editing your ServiceNow applications.

Procedure
1. Click **Create Project** in the status bar on the bottom of the VS Code IDE.

You can also use a keyboard shortcut, Control+Shift+P on Windows or Command+Shift+P on MacOS, to open the command palette and choose **Now: Create Now Project** from the list.

The **Create Project** wizard opens.

2. Enter the URL of the ServiceNow instance and press Enter to confirm.
   Include the complete URL for your instance, for example, [https://example.service-now.com](https://example.service-now.com)

3. When prompted, select the Authentication type.
• To create a basic application, choose **Basic** and fill in the following fields, when prompted.

### Basic application

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>Enter the User name of the ServiceNow instance URL.</td>
</tr>
<tr>
<td>Password</td>
<td>Enter the password of the ServiceNow instance URL.</td>
</tr>
</tbody>
</table>

### Project type

• To create an OAuth application, choose **OAuth** application and fill in the following fields, when prompted.

### OAuth application

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>Enter the Username of the ServiceNow instance URL.</td>
</tr>
<tr>
<td>Password</td>
<td>Enter the password of the ServiceNow instance URL.</td>
</tr>
<tr>
<td>Client ID</td>
<td>Client ID is automatically generated by ServiceNow OAuth server.</td>
</tr>
<tr>
<td>Client secret</td>
<td>Client secret for the OAuth application.</td>
</tr>
</tbody>
</table>

4. Select the project type when prompted.
Note: You can enable the Packages and Plugins project types by enabling them from the Settings. Navigate to Preferences > Settings > Extensions > Now > Project Settings to enable them.

5. Select Import Existing.

Import an application into Visual Studio Code

After you create a project, import an application from your instance into the project to begin editing.

Procedure

1. Visual Studio Code displays the list of applications available in your instance for the selected application type. Select an application.

2. At the Configure file types prompt, click OK to select all file types.

By default, Visual Studio Code imports all the file types of your application. You can modify the selection of the file types while editing the application, by choosing Now: Configure File Types from the command palette and choosing the file types from the list.
Your ServiceNow application is imported into your workspace.

**Note:** Even if your application exists on your instance, you must create a project for it in Visual Studio Code.

You can switch applications within the workspace by clicking the name of the application, for example, EmployeeApp in this case, in the status bar at the bottom of the VS code IDE or choosing **Now: Select Application** from the command palette and selecting the application name from the list. Similarly, you can modify the update set by clicking the current update set icon, for example, EmployeeApp in this case, in the status bar at the bottom of the VS code IDE or by choosing **Now: Select Update Set** from the command pallet and selecting the name of the updateset.

**Synchronization between Visual Studio Code and a ServiceNow instance**

You can synchronize your applications between Visual Studio Code and your ServiceNow instance.

**Synchronization**

Changes made to your applications in Visual Studio Code are stored on your local file system only until you synchronise with your instance.

Synchronization accounts for the following changes:

- File modifications on the instance or in the local file system
- File creation on the instance or in the local file system
- File deletion on the instance or in the local file system

You can sync your currently selected record or all records in the current application.

**Sync the current file between a Visual Studio Code workspace and a ServiceNow instance**

Identify conflicting files and merge or overwrite the changes, between the current file in your Visual Studio Code workspace and your instance.

**Before you begin**

Role required: admin
Procedure

1. Click the Sync icon from the menu on the top-right corner of the Visual Studio Code IDE.
   You can also use a keyboard shortcut Control+Shift+P in Windows and Command+Shift+P in MacOS to open the command palette and choose Now: Sync Current File from the list.
   - If the changes occur either on the server or on the client, the extension synchronizes the file successfully without showing up any errors.
   - If there are no differences between the local and server versions, then the message No changes detected appears on the bottom of the page.
   - If the same file is modified both on the client and the server, then the system displays the conflicts in a Conflicting Files dialog box.

2. If a conflict list is displayed in a Conflicting Files dialog box, check the conflicts.
   - Compare the Server and Client version using the Diff window, and evaluate the changes.
   - Override the Server or Client changes.

3. Check for the differences in the Conflicting Files list.

A dialog box displays the following options:
- Open Diff: Check for the differences between two versions and make the necessary changes.
- Mark Resolved: Resolve the differences and sync the versions.
• Overwrite Server: Choose if you want to override the server version with the client version.

• Overwrite Local: Choose **Override Local** to override the local version with the server version.

4. Click **Open Diff**.
   You can see two versions of the file with the differences highlighted.

5. Make changes to the server version from the client version and save the file.

6. Click **Mark as Resolved** under **Conflicting Files**.
   The changes are successfully synchronized upon the next synchronization.
   When the changes are merged successfully, the message **Sync completed successfully** appears at the bottom of the page.

Synchronize the current project between a Visual Studio Code workspace and a ServiceNow instance

Synchronize files between the Visual Studio Code workspace (client/local) and the instance (server) bi-directionally.

**Before you begin**
Create your project and start editing your applications.

**About this task**
The changes made to the application files in the Visual Studio Code workspace are initially stored in the local file system. When you run the **Sync** command, the local changes made to the application files are sent to the server for persistence.
Procedure

1. Click Sync on the status bar on the bottom of the Visual Studio Code IDE. You can also use a keyboard shortcut, Control+Shift+P on Windows or Command+Shift+P on MacOS, to open the command palette and choose Now: Sync Current Project from the list.

![Sync current project](image)

- If the changes occur either on the server or on the client, the extension synchronizes the file successfully without showing up any errors.
- If there are no differences between local and server versions, then the message No changes detected appears on the bottom of the page.
- If the same file is modified both on client and server, then the system shows up the conflicts.

2. If a conflict list is displayed, right-click on the conflicting file to open the context menu and select from the context menu. Alternatively, select the file and click Open Diff menu from the top of the page.

![Conflicting files display](image)
• Choose **Override Server** if you want to overwrite the server version with the client version

• Choose **Override Local** to overwrite the client version with the server version.

• Choose **Mark as resolved** to remove the flag indicating a conflict. The server version will be overwritten in the next synchronization.

3. Retain the desired changes in the client file and save.

4. once you have resolved all file conflicts, click **Mark as Resolved** under **Conflicting Files**.

   The changes are synchronized on the next sync. When the changes are merged successfully, the message **Sync completed successfully** appears in the bottom of the page.

   **Note:** The files are checked for syntax errors at the beginning of synchronization process. Any errors found are presented in the problem tab at the bottom of the screen.

**What to do next**

All the changes are tagged to the update set in the status bar. When a project is loaded or selected, the default update set is displayed in the status bar. Click the update set picker to choose the current update set (valid until the IDE or project is closed). If you do not explicitly select an update set, all writes happen to the default update set.

**Clear instance credentials in Visual Studio Code**

Clear the stored credentials of the current project from the system. User credentials (instance URL, user name, password) of a project are stored in the operating system credentials vault so that the user does not need to log in each time.
About this task
Once you clear credentials from the current instance, the ServiceNow Extensions for Visual Studio Code extension prompts for authentication the next time you open VS Code. You can continue working in the current session.

ℹ️ Note: The password is not stored in the Visual Studio Code settings.

Procedure

1. Navigate to View > Command Palette.
   You can also use a keyboard shortcut, Control+Shift+P on Windows or Command+Shift+P on MacOS to open the command palette.

2. Choose Now: Clear Credentials from the command palette.

   ![Now: Clear credentials command]

   The credentials of the current project are cleared.

3. Optional: Navigate to Code > Preferences > Settings > Workspace > Extensions > ServiceNow to check whether the credentials and the instance URL have been deleted.
Clear credentials

User  Workspace

Commonly Used
  › Text Editor
  › Workbench
  › Window
  › Features
  › Application

Extensions
  ✔ Beautiful config
  ✔ CSS
  ✔ Emmet
  ✔ ESLint
  ✔ Git
  ✔ Grunt
  ✔ Gulp
  ✔ HTML
  ✔ Jake
  ✔ JSON
  ✔ LESS
  ✔ Merge Conflict
  ✔ Node debug
  ✔ Now
  ✔ Npm
  ✔ PHP
  ✔ Reference Search ...
  ✔ SCSS (Sass)
  ✔ TypeScript

Now

Now: Enable Verbose Logging

Now: Enable verbose logging

Now > Instance > Authentication: Type (Modified in: Workspace)

Basic

Now > Instance > Host: Url (Also modified in: Workspace)

Instance URL

http://exampleinstance.servicenow.com

Now > Instance > OAuth > Client: ID (Modified in: Workspace)

Client Id

Now > Instance > OAuth > Client: Secret (Modified in: Workspace)

Client Secret

Now > Instance > User: Name (Modified in: Workspace)

User Name

xyz

Now > Instance > User: Password (Also modified in: Workspace)

Password

.........

Note: To re-enter your credentials, navigate to CodePreferencesSettingsWorkspaceExtensionsNow. Enter your credentials in Username and Password fields.

Reset a project in Visual Studio Code

Reset the project to the state on the server by discarding all the local changes if you encounter any serious sync issues.

About this task

When you reset a project to its server state, all unsynchronized changes are lost.
Procedure

1. Navigate to View > Command Palette in.
   You can also use a keyboard shortcut, Control+Shift+P on Windows or Command+Shift+P on MacOS, to open the command palette.

2. Choose Now: Reset Project from the drop-down list.

3. Click Yes in the dialog box.

Results

The project is reset to its original state.

Note: This option should be used only when something goes wrong, for example, if the project is corrupted or you are facing sync issues.

Related information

Create a project in VS Code

Add custom file types in Visual Studio Code

If you have work with file types other than the default types provided, you can add additional file types to your instance and edit them in ServiceNow extension for Visual Studio Code.

Before you begin

Role required: admin

Procedure

1. Create a new file type in your ServiceNow instance and inherit it from the application file.

2. Add custom columns of the type Script and String, and enter some data.

   You can also use a keyboard shortcut, Control+Shift+P on Windows or Command+Shift+P on MacOS, to open the command palette.

4. Choose Now: Add Custom File Types from the command palette.
   The command fetches all the existing tables in the current application and opens the Add new file type wizard.

5. Select the file type you created in your ServiceNow instance from the list.

6. At the Select tags prompt, select the tags you created in the instance.
   All tags are selected by default. Click a tag’s check box to deselect it.
The new file type is successfully added to the app.config.json file. You can also add the desired file types manually to the app.config.json file under your project. See the example below.

7. Absorb this new file type to the current project using the **Now: Configure File Types** command from the command palette.

8. Select the file type you created from the list of file types and click **OK**. The selected file type is ready for editing in the ServiceNow Extensions for Visual Studio Code extension.

**Example:**
The following is an example for app.config.json

```json
"CustomFileTypes": {
   "sc_ic_aprvl_type_defn": {
      "superCoverName": "Miscellaneous",
      "coverName": "Approval Type Definition",
      "tags": {
         "approver_script": "js",
         "approver_html": "html",
      }
   }
}
```

- superCoverName is a name of the super parent directory (Should always point to Miscellaneous).
- coverName is a descriptive name of the table.
- sc_ic_aprvl_type_defn is a table identifier.
• tags represent the set of scriptable columns in the table
• approver_script is a name of the table column / xml tag
• js is an extension of the file (js | html | css | json)

Create a file in VS Code

Use the ServiceNow extension to create a new file for your application with your project.

Before you begin
Import an application in your VS Code project to create a file within your project.

Procedure

1. Open the VS Code project containing your application.

2. Click the Create File on the status bar on the bottom of the VSCode IDE.
   You can also use a keyboard shortcut, Control+Shift+P on Windows or Command+Shift+P on MacOS, to open the command palette and choose Now: Create New File from the list.

3. Select the file group from Select File Group list.
4. Select the file type from the **Select File Type** list from the command palette at the top of the screen.

5. Enter a name for the file in the Create New File dialog box. The extension creates a file in your application on your local file system. The new file is added to your instance when you synchronize your workspace.

**Search files on your instance in VS Code**

Search and download any script files on your instance using the ServiceNow Extensions for VS Code extension.
Procedure

1. Open a project in VS Code containing your application.

2. Click the **File Search** on the status bar on the bottom of the VSCode IDE. You can also use a keyboard shortcut, Control+Shift+P in Windows or Command+Shift+P on MacOS, to open the command palette and choose **Now: Global Search** from the list.

3. Select the file group from the **Select File Group** list in the command palette on the top of the screen.
Select File Group list

4. Select the file type from the Select a File Type list in the command palette on the top of the screen.

Select a file type list

5. Select files from the Select files to download list.
Select files to download list

You can select and download multiple files at a time.
The selected files are downloaded to the scratch folder of your application.
You can edit the selected files.

Related information
Create a file in VS Code
Run background scripts using VS Code
Run scripts from your ServiceNow instance in VS Code using the ServiceNow Extensions for VS Code.

Before you begin
Role required: admin
**Procedure**

1. While writing the script in the VS Code IDE, select the script in the text file and right-click on the script.

2. Select **Run Background Script - Current** or **Run Background Script - Global** from the context menu. The background scripts runner tab opens in VS Code and shows the location where the script is executed in the server.

**IntelliSense in VS Code**

The IntelliSense code editing feature in the ServiceNow Extensions for VS Code extension provides intelligent code completions which provide auto-completion options to make writing code faster and more efficient.

When you start scripting in the VS Code, the IntelliSense feature shows auto-completion options suitable for your code context at the top of the list.

**IntelliSense for GlideSystem**

<table>
<thead>
<tr>
<th>gs</th>
</tr>
</thead>
<tbody>
<tr>
<td>getSelection</td>
</tr>
<tr>
<td>GlideSchedule</td>
</tr>
<tr>
<td>GlideScopedEvaluator</td>
</tr>
<tr>
<td>GlideScriptableInputStream</td>
</tr>
<tr>
<td>GlideScriptedProcessor</td>
</tr>
<tr>
<td>GlideServletRequest</td>
</tr>
<tr>
<td>GlideServletResponse</td>
</tr>
<tr>
<td>GlideSession</td>
</tr>
<tr>
<td>g_user</td>
</tr>
<tr>
<td>g_list</td>
</tr>
<tr>
<td>getMessage</td>
</tr>
</tbody>
</table>

The auto-completion options available are:

- Auto-completion of methods and properties for client and server scripts.
- Auto-completion of methods and properties for standard JS APIs
- Auto-completion of Jelly tags and attributes.
- Code completion in commonly used standards.

The extension supports snippets (templates) for commonly used code patterns and loads snippets defined by the user in Macros table of the Syntax Editor on an instance. Snippets appear in IntelliSense when you type `Space` along with other suggestions. Alternatively, snippets can also be inserted using the **Insert Snippet** command from the command palette in VS Code.
Telemetry in ServiceNow Extensions for VS Code

ServiceNow Extensions for VS Code collects information on its various usage patterns such as the commands run with the help of Now Telemetry.

Now Telemetry helps in the following:

- Debug issues
- Analyse most used functions or features
- Capture user actions
- Improve the extension

ℹ️ Note: Telemetry collects anonymous information related to usage and does not capture any personal information such as name and email ID of the users. Data is accessible to ServiceNow only and is not shared with anyone.

Telemetry is enabled by default for the ServiceNow extension. If you do not wish to send usage data, you can set the `now.telemetry.enableTelemetry` user setting to false. Alternatively, select **File** (Windows) or **Code** (Mac) and navigate to **Preferences > Settings > Extensions > Now** and clear **Now Telemetry** check box to disable the feature.

ℹ️ Note: The usage data will be sent until you disable the setting. The extension abides by the global telemetry setting `telemetry.enableTelemetry`. If that is set to false, Now telemetry is disabled.
Legacy application creator

Application developers have options when creating applications.

⚠️ **Note:** As of the New York release, Guided Application Creator is the default way to set up applications on the Now Platform. There is no migration path from the legacy application creator to Guided Application Creator.

You can add a property to force legacy application creator to open instead of Guided Application Creator. For more information, see Activate the legacy application creator.

Sample application creation options

To start creating an application, navigate to System Applications > Studio and click Create Application. Select one of the available options.

- Start from scratch.
- Create custom application.
- Start from an existing service (available only when the Service Creator plugin is activated).
- Start from a template.
- Start from global.

The application creation method determines the application scope.
After you create an application, it appears on the applications list (System Applications > Applications) and is set as the current application in the application picker.

**Activate the legacy application creator**

The legacy application creator is replaced by Guided Application Creator in the New York release. You can add a property to force the legacy application creator to open instead of Guided Application Creator.

**Before you begin**
Role required: admin

**About this task**
The system property `sn_g_app_creator.use.legacy.appcreator` is set to `false` by default. If you set the property to `true`, the legacy application creator opens when you create an application in the Application Manager or Studio; Guided Application Creator is no longer available.

> Note: The legacy application creator platform opens by default if you are using Internet Explorer 11 or older.

**Procedure**

1. Add a property with the following settings.
   - Name: `sn_g_app_creator.use.legacy.appcreator`
   - Type: true | false
   - Value: true

2. Click Submit.

**Legacy: Start from scratch**

You can create an empty application using the Start from scratch option.

**Before you begin**
Complete the steps in Activate the legacy application creator.

Role required: admin

**About this task**
This option is good for applications that only use one configuration record, such as a workflow or script-based application.
Note: As of the New York release, Guided Application Creator is the default way to set up applications on the Now Platform. There is no migration path from the legacy application creator to Guided Application Creator.

Procedure
1. From the Start from scratch row, click Create.
2. Enter a Name for the application.
3. Create Create.
4. In the Confirm Application dialog box, click OK.

Legacy: Create a custom application
You can create an application with UI and data elements using the Create custom application option.

Before you begin
Complete the steps in Activate the legacy application creator.
Role required: admin

About this task
This option is good for applications that require UI elements and tables.

Note: As of the New York release, Guided Application Creator is the default way to set up applications on the Now Platform. There is no migration path from the legacy application creator to Guided Application Creator.

Procedure
1. From the Create custom application row, click Create.
2. Enter a Name for the application.
3. Optional: Modify any of these fields if the default values do not meet your needs.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Menu</td>
<td>The application menu users can use to access the application.</td>
</tr>
<tr>
<td>User Role</td>
<td>The role a user must have to access the application.</td>
</tr>
</tbody>
</table>
4. **Optional:** Select **Create Table** and provide table details.
   
   (Optional) If you do not create a table using this option, you must manually define a table before you can store any application data.

5. Click **Create**.

6. On the Confirm Application dialog box, click **OK**.
   The system creates the application and provides an **Edit App** option.

---

**Legacy: Start from an existing service**

You can convert an existing service to an application if service creator is active.

**Before you begin**

Complete the steps in **Activate the legacy application creator**.

**Role required:** admin

**About this task**

⚠️ **Note:** As of the New York release, Guided Application Creator is the default way to set up applications on the Now Platform. There is no migration path from the legacy application creator to Guided Application Creator.

You may want to convert successful services into full applications to take advantage of features only available to applications such as contextual security, data or UI policies, UI actions, and auditing.

When converting a service that exists on a production instance, you should pull the service to a non-production instance and convert the service to an application in that environment. After completing the conversion, you can push the new application to the production environment.

**Procedure**

1. From the **Start from a service** row, click **Create**.

2. In the **Create from service** field, select a published service.

3. **Optional:** Change the application details and catalog item details if the default values do not meet your needs.

4. **Optional:** Select **Replace original service** to deactivate the catalog item for the service and replace it with an equivalent catalog item for the application.
   
   (Optional) When replacing an existing service with an application, the application preserves the same end-user experience for submitting and
fulfilling requests. Any service requests that were made prior to the conversion are converted to requests within the application.

5. Click **Create**.

6. Click **OK**.

**Legacy: Conversion mapping**

The conversion process uses the following service creator elements to create application records.

### Service Creator conversion mapping

<table>
<thead>
<tr>
<th>This Service Creator element</th>
<th>Becomes this application element</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service category table</td>
<td>Application table</td>
</tr>
<tr>
<td>Service name</td>
<td>Table label</td>
</tr>
<tr>
<td>Service form layout</td>
<td>Application table form layout</td>
</tr>
<tr>
<td>Service catalog policies</td>
<td>Application table UI policies</td>
</tr>
<tr>
<td>Service catalog item</td>
<td>Application catalog item</td>
</tr>
<tr>
<td>Service catalog variable</td>
<td>Fields on the application table</td>
</tr>
<tr>
<td>Category form layout</td>
<td>Table form layout</td>
</tr>
<tr>
<td>Service task workflow</td>
<td>Application table workflow</td>
</tr>
</tbody>
</table>

**Legacy: Start from a template**

You can create service management applications from a template using the **Start from a template** option if service management is active.

**Before you begin**

Complete the steps in **Activate the legacy application creator**.

Role required: **admin**

**About this task**

This option is good for applications that need to track the status of requests or task-driven records.
Note: As of the New York release, Guided Application Creator is the default way to set up applications on the Now Platform. There is no migration path from the legacy application creator to Guided Application Creator.

Procedure
1. From the Start from a Template row, click Create.
2. Enter a Name for the application.
3. Optional: Click Configure from the section containing the process you want to use, and select the configuration options you want to use.
4. Click Create from the section containing the process you want to use.
5. In the Confirm Application dialog box, click OK.
   The system creates the application and provides an Edit App option.

Legacy: Start from global
You can create a legacy application in the global scope using the Start from global option.

Before you begin
Complete the steps in Activate the legacy application creator.

Role required: admin

About this task
Applications created in the global scope do not benefit from application scope protections. Use this option to support legacy applications from previous versions.

Note: As of the New York release, Guided Application Creator is the default way to set up applications on the Now Platform. There is no migration path from the legacy application creator to Guided Application Creator.

Procedure
1. From the Start from global row, click Create.
2. Enter a Name for the application.
   The default value for the Scope field is global and cannot be changed.
3. Optional: Modify any of these fields if the default values do not meet your needs.
### Option Description

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Menu</strong></td>
<td>The application menu users can use to access the application.</td>
</tr>
<tr>
<td><strong>User Role</strong></td>
<td>The role a user must have to access the application.</td>
</tr>
</tbody>
</table>

**4. Optional:** Select **Create Table** and provide table details.

(Optional) If you do not create a table using this option, you must manually define a table before you can use the application.

**5. Click Create.**

**6. On the Confirm Application dialog box, click OK.**

The system creates the application and provides an **Edit App** option.

### Related information

- Add a property using `sys_properties.list`

### APIs and scripts

APIs are used in scripts to change functionality and to provide new features and applications to your instance.

<table>
<thead>
<tr>
<th>Explore</th>
<th>Set up</th>
<th>Administer</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Domain separation and Script Debugger</td>
<td>• No set up is required to use scripts.</td>
<td>• Some APIs require a plug in to be activated. See the specific API description.</td>
</tr>
<tr>
<td><strong>Use</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Script information</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Script Debugger and Session Log</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• JavaScript API reference</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• REST API reference</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Using extension points to extend application functionality</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Troubleshoot and get help</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Develop</th>
<th>Integrate</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Developer training</td>
<td>• Web services</td>
</tr>
<tr>
<td>• Developer documentation</td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Using extension points to extend application functionality

Use extension points to extend the functionality of an application without altering the original application code. You can use pre-existing extension points available in selected Now Platform applications, or you can add extension points when you develop custom applications in your own instance.

Using extension points

By using extension points, you can integrate customizations without actually altering the core components in the application code. Extension points can prevent your custom code interactions from breaking, which often occurs after an upgrade if you directly embed the custom code into the application code.

Extension points that are embedded in the application code act as out-points, where data passes to the custom code, and as in-points that handle the returned results. When creating an application, the returned data or objects must conform to the requirements that you define for the extension point.

Types of extension points

You can create extension points to process the custom code that uses the following types of artifacts:

**Scripted extension points**

Extension points in server-side script includes that store JavaScript functions and object classes. To learn about scripts includes and how to implement scripted extension points in the application and custom code, see:

- Script includes
- Creating and adding a scripted extension point
- Registering custom script includes against the scripted extension points

**UI extension points**

Extension points that are used in server-side UI macros such as HTML extensions.
UI macros are discrete scripted components that you can add to the user interface. You use them to add custom content to a UI page, without having to directly modify the page. For example, you can use UI macros to add headers and footers to the standard Knowledge Base View (kb_view) UI page in which KB articles appear.

To learn about UI macros and how to implement UI extension points in the application and custom code, see:

- UI Macros
- Creating and adding a UI extension point
- Registering custom UI macros against the UI extension points

Client extension points

Extension points that are used in client-side UI scripting, typically for modifying forms.

UI scripts enable you to package client-side JavaScript into a reusable form, which is similar to how script includes store server-side JavaScript. You can create UI scripts and run them from client scripts, from other client-side script objects, and from HTML code.

To learn about UI scripts and how to implement client extension points in application and custom code, see:

- UI scripts
- Creating and adding a client extension point
- Registering custom UI scripts against the client extension points

When you use extension points to process customizations, you create a defined structure for integrating custom data or functionality into an application. Custom server-side script includes, UI macros such as HTML extensions, and client-side UI scripts are all external to the application code and only interact with it at specified extension points.

Application code

The term application code refers to:

Standard application code

Standard, or base, application code that comprises the Now Platform.

Pre-defined extension points are already embedded in certain applications, such as Customer Service Management and Field Service Management. To learn more about the Now Platform...
applications that contain pre-defined extension points, see the following:

<table>
<thead>
<tr>
<th>Application</th>
<th>Extension point topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coaching</td>
<td>Coaching troubleshooting</td>
</tr>
<tr>
<td>Continual Improvement Management</td>
<td>Improvement with other applications</td>
</tr>
<tr>
<td>Customer Service Management</td>
<td>Extension points in Customer Service Management</td>
</tr>
<tr>
<td></td>
<td>Creating custom user roles</td>
</tr>
<tr>
<td></td>
<td>CSM integration with Change Management</td>
</tr>
<tr>
<td></td>
<td>CSM integration with Incident Management</td>
</tr>
<tr>
<td></td>
<td>CSM integration with Problem Management</td>
</tr>
<tr>
<td></td>
<td>CSM integration with Request Management</td>
</tr>
<tr>
<td>Field Service Management</td>
<td>Extension points in Field Service Management</td>
</tr>
<tr>
<td>Knowledge Management</td>
<td>Use extension points for Knowledge Management</td>
</tr>
<tr>
<td>Orchestration</td>
<td>Client software distribution extension network Installed with client software distribution</td>
</tr>
<tr>
<td>Password Reset</td>
<td>Password Reset script includes</td>
</tr>
</tbody>
</table>

**Internally developed custom applications for your enterprise**

You can add extension points to handle the registration of custom artifacts that are used to modify or extend the functionality of an application. When developers create custom code, they register, or pair, specific custom artifacts with specific extension points. Adding extension points enables integration of future customizations without having to change your base code.
Creating an extension point in the application code

You want to ensure that the internally developed applications that you create for your enterprise can be properly customized, combined, and upgraded as needed. Extension points enable you to modify the functionality and user interface for an application without editing its core components. Use of extension points also creates a highly defined structure for functionality extensions.

When you create an extension point, you can restrict its use to the application scope in which it is defined or specify that the extension point can run in all scopes globally. If the application scope is restricted, customizations that are registered against the extension point can only run in the designated application scope. Before designing and building an application that includes extension points, you should:

• Identify where to place extension points in the application code to accommodate custom script includes, UI macros, or UI scripts.
• Determine the content and structure for an extension point. This definition should describe how a customization should be structured, how it interacts with the application code, and how the data that is returned by a customization at the extension point is processed.

To create extension points and embed them in the application code, perform the following tasks:
1. Create an extension point and define its custom interface. This is an example of a scripted extension point.

2. Designate specific places in the application code where data or objects can be sent to a customization, and where data is returned.
3. Create an artifact, such as script includes, UI macros, or UI scripts, with the appropriate API call. The API call identifies the extension point at which registered custom artifacts execute in the application code.

Enabling debugging and logging

When you create an extension point, you should also enable debugging and logging. Debugging and logging help you to identify and fix issues that are related to the use of custom artifacts with an extension point. When you enable the debugger, you can set break points in script includes, UI macros, or UI scripts. You can check the logs to see the following details:

- When an extension point finds active extension instances, including the total number of extension instances found and the time each extension instance was found.
- When an extension point executes extension instances, including the total number of custom artifacts executed and the time of each executed. It also includes the total number that failed to execute and the time at which each extension instance failed.
Note: To learn more about how to enable debugging and how it works, see:

- Script Debugger and Session Log
- Debugging applications
- Writing to the debug log

Registering a custom artifact against an extension point

When you are customizing an application, you extend the base functionality by using custom artifacts, such as server-side script includes, UI macros, or client-side UI scripts. To design and build custom artifacts, perform the following tasks:

1. Review the listing of available extension points that are appropriate to the specific type of custom artifact that you are creating.
   - These listings include extension points that you created and any pre-defined extension points that are embedded in the Now Platform functions.
   - Each listing includes information about an extension point in the application code that calls a custom artifact and what data or objects should be returned to it.

2. Select an extension point.

3. Determine how to structure the custom artifact. The structure should be based on the extension point descriptions. The descriptions include the requirements for using the artifact with the custom code and where the artifact will located in the base application code.

4. Create the custom artifact and code when you register it against the selected extension point. Through registration, you create an extension instance record that links the extension point definition to its implementation in the custom artifact.

How registered custom artifacts are processed

When the application code executes and finds an embedded API call containing an extension point, it:

1. Uses the extension point in the API call to determine which custom artifacts are registered against it.

2. Sends the appropriate data or objects to the registered custom artifacts.

3. Collects the returned output from each custom artifact.

4. Processes and incorporates the returned results into the base application.
Using scripted extension points in server-side scripts

Use the scripted extension points in the server-side script includes that store JavaScript functions and object classes. By using extension points, you can integrate customizations without actually altering the core components in the application code.

You create the scripted extension points and add them to the script includes in the base application code. When customizing a base application, you implement the scripted extension points by creating the custom script includes and registering them against the scripted extension points.

⚠️ Note: For an example of a scripted extension point and to learn more about this process, see Using extension points to extend application functionality.

Creating and adding a scripted extension point

When developing an application, create scripted extension points and add them to the script includes in the application code. Use a scripted extension point to designate the specific location where data or objects can be sent to a registered custom script include and where returned results are processed.

⚠️ Note: For an example of a scripted extension point, see Using extension points to extend application functionality.

Create a scripted extension point

Create a scripted extension point that can be placed in a script include in the base application code. By placing the scripted extension point, you designate the specific location in the application code where data or objects can be sent to a customization and where data is returned.

Before you begin

Role required: You must have the specific role for the developer or administrator of the application, or you must have the admin role.

⚠️ Note: To learn about application-specific administrator roles and delegated development, see Access control rules in application administration apps and Delegated development and deployment.

About this task

Define the content and structure for an extension point. This definition should describe how a customization should be structured, how it interacts with the application code, and how the data that is returned by a customization at the extension point is processed.

By creating a good definition, you provide a structure for the extensions.
Procedure

1. Navigate to System Extension Points > Scripted Extension Points.
2. Click New.
3. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Unique name for the extension point. The name can be up to 100 alphanumeric characters, including special characters.</td>
</tr>
<tr>
<td>API Name</td>
<td>Name of the extension point API that is pre-pended with the application scope to which it applies. This is a system-assigned name and cannot be changed.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope against which the extension point is assigned. This is system-assigned and cannot be changed. For more information about the protections that are offered by the use of scoping, see Application scope.</td>
</tr>
<tr>
<td>Restrict to this scope</td>
<td>Option for restricting the extension point to the application scope only.</td>
</tr>
<tr>
<td>Description</td>
<td>Requirements for the custom script include, such as how the UI script should be structured and how it should operate with the application code.</td>
</tr>
<tr>
<td>Example</td>
<td>Example of how a custom script include that interacts with this extension point should be structured so it operates with the application code.</td>
</tr>
</tbody>
</table>

4. Click Submit.
Add a scripted extension point in the base application code

Add the scripted extension point into a script include in the base application code. To add the extension point, place an API call that identifies the location at which registered custom artifacts execute.

Before you begin

You must first create a scripted extension point before you can add it to a script include in the application code. Role required: You must have the specific role for the developer or administrator of the application, or you must have the admin role.

Note: To learn about application-specific administrator roles and delegated development, see Access control rules in application administration apps and Delegated development and deployment.

Procedure

1. In the application code, access the existing script include that you want to add a scripted extension point to, or create a new script include. To learn more about creating script includes, see Script includes.

2. In the script include, add a line of code at the location that you expect to collect and process the custom script output.

The code must contain the `getExtensions` command that identifies the extension point against which the custom script includes are registered. For example:

```java
var canReserve = true;
try {
    var eps = new GlideScriptedExtensionPoint().getExtensions("CanReserve");
    eps.forEach(function(ep) {
        if (ep.handles(thing)) {
            canReserve = ep.canReserve(thing, from, until, by);
        }
    });
} catch (ex) {
    gs.error("Error running ShouldAllow extension points!");
}
```

Note: It is also a good practice to deliver error messages for application customizers when their custom scripts run in an extension instance. If something fails when the function returns arguments to the extension point, the error messages include information about how to troubleshoot the custom script.
Registering custom script includes against the scripted extension points

Register custom script includes against a selected scripted extension point in the application code.

By registering the custom script, you create an extension instance record that links the scripted extension point definition to its implementation in the custom script include.

Register a custom script include

Create a custom script include and code when you register it against a selected scripted extension point. Through registration, you create an extension instance record that links the extension point definition to its implementation in the custom artifact.

Before you begin

Role required: You must have the specific role for the developer or administrator of the application, or you must have the admin role.

Note: To learn about application-specific administrator roles and delegated development, see Access control rules in application administration apps and Delegated development and deployment.

Procedure

1. Navigate to System Extension Points > Scripted Extension Points.
2. In the Extension Point [sys_extension_point] table, select the scripted extension point that you want to register a custom script include against.

Note: This listing includes the scripted extension points that you created and any pre-defined scripted extension points that are embedded in Now Platform functions (for example, Customer Service Management and Field Service Management).

To learn more about the functions that contain the pre-defined extension points, see the Related Topics links in Using extension points to extend application functionality.

3. In the Extension Point record, click the Create Implementation related link to create a new custom script include and register it against the selected scripted extension point.
4. In the Script Include form, create the custom code.
   To learn more about creating script includes, see Script includes.
5. Click Update.
Results
The custom script include is created and registered against the scripted extension point.

Review scripted extension instances
Review the listing of custom script includes that are registered against a scripted extension point. Extension instances indicate the application scope in which registered custom script include are used, and provides a link to each. You can also activate or inactive an extension instance.

Before you begin
Role required: You must have the specific role for the developer or administrator of the application, or you must have the admin role.

Note: To learn about application-specific administrator roles and delegated development, see Access control rules in application administration apps and Delegated development and deployment.

Procedure
1. Navigate to System Extension Points > Scripted Extension Points.
2. In the Extension Point [sys_extension_point] table, select the scripted extension point.
3. In the Implementations related list of the Extension Point record, select a scripted extension instance.
   The system displays the Extension Instance record.

Extension Instance record

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point</td>
<td>Unique name for the scripted extension point. The name can be up to 100 alphanumeric characters, including special characters.</td>
</tr>
<tr>
<td>Class</td>
<td>Script include in the application code in which the scripted extension point is implemented.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope against which the extension point is assigned. This is system-assigned and cannot be changed. For more information about the protections that are offered by the use of scoping, see Application scope.</td>
</tr>
<tr>
<td>Active</td>
<td>Option for activating an extension point in the current instance.</td>
</tr>
</tbody>
</table>

4. Click Submit.
Related information

Script includes

Using extension points to extend application functionality

Using UI extension points in server-side UI macros

Use UI extension points in the server-side UI macros, such as HTML extensions, to add custom content to a UI page without having to directly modify the page. By using UI extension points, you can integrate customizations without actually altering the core components in the application code.

You create the UI extension points and add them to the UI macros in the base application code. When customizing a base application, implement the UI extension points by creating the custom UI macros and registering them against the UI extension points.

Note: You create and register the UI extension points in a manner that is similar to how you implement the scripted extension points. For a detailed example of a scripted extension point, and to learn more about this process, see Using extension points to extend application functionality.

Creating and adding a UI extension point

When developing an application, create UI extension points and add them to the server-side UI macros, such as HTML extensions, or the UI pages in the base application code.

Use a UI extension point to designate the specific location where data or objects can be sent to a registered custom UI macro and where returned results are processed.

Create a UI extension point

Create a UI extension point that you can place in the UI macros, such as HTML extensions, in the application code. By placing the UI extension point, you designate the specific location in the application code where data or objects can be sent to a customization and where data is returned.

Before you begin

Role required: You must have the specific role for the developer or administrator of the application, or you must have the admin role.

Note: To learn about application-specific administrator roles and delegated development, see Access control rules in application administration apps and Delegated development and deployment.
About this task
Define the content and structure for an extension point. This definition should
describe how a customization should be structured, how it interacts with the
application code, and how the data that is returned by a customization at the
extension point is processed.

By creating a good definition, you provide a structure for the functionality of the
extensions.

Procedure
1. Navigate to System Extension Points > UI Extension Points.
2. Click New.
3. On the form, fill in the fields.

UI Extension Point form

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Unique name for the extension point. The name can be up to 100 alphanumeric characters, including special characters.</td>
</tr>
<tr>
<td>API Name</td>
<td>Name of the extension point API that is pre-pended with the application scope to which it applies. This is a system-assigned name and cannot be changed.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope against which the extension point is assigned. This is system-assigned and cannot be changed. For more information about the protections that are offered by the use of scoping, see Application scope.</td>
</tr>
<tr>
<td>Restrict to this scope</td>
<td>Option for restricting the extension point to the application scope only.</td>
</tr>
<tr>
<td>Allow access over AJAX/REST</td>
<td>Option for enabling or disabling access to the UI extension point over AJAX or REST.</td>
</tr>
</tbody>
</table>
4. Click **Submit**.

### Add a UI extension point in the base application code

Add the UI extension point into a UI macro or HTML extension in the base application code. To add the extension point, place an API call that identifies the location at which registered custom artifacts execute.

**Before you begin**

You must first create a UI extension point before you can add it to a UI macro or UI page in the application code. Role required: You must have the specific role for the developer or administrator of the application, or you must have the admin role.

**Note:** To learn about application-specific administrator roles and delegated development, see Access control rules in application administration apps and Delegated development and deployment.

**Procedure**

1. In the application code, access the existing UI macro that you want to add a scripted extension point to or create a new one. To learn more about creating UI macros with jelly tags, see UI Macros and Jelly tags.

2. In the UI macro, add a jelly tag with a line of code at the location that you expect to collect and process custom UI macro output. Typically, this line of code contains:
• A call_extension command that identifies the name of the UI extension point (for example, extension="global.KMArticle.ViewHeader") against which the custom UI macros are registered.

This line of code is similar to what is in an application code script include for scripted extension points. For an example, see Register a custom script include.

• Arguments that are passed into the UI macro when it is rendered. For example, for a Knowledge Base article, knowledgerecord="${knowledgeRecord}" passes in the current knowledge record.
• (Optional) If there are multiple implementations of this extension point, using a `limit` command specifies that x number of implementations should be returned (for example, `limit="1"`). This figure shows a UI macro that contains these commands.
<-- MB Extension point call to customize footer -->
<call_extension extension="global.KWArticleViewFooter" knowledgeRecord="#knowledgeRecord" limit="1" />

<!--  -->
<if test="#isMobileView = true">  
  <div class="mobile">  
    <inline template="kb_view_common_body"/>
  </div>
  </if>  
  </if>  
  </if>  
  </if>

Protection policy: None
Registering custom UI macros against the UI extension points

Register custom UI macros against a selected UI extension point in the application code.

By registering custom UI macros, you create an extension instance record that links the UI extension point definition to its implementation in the custom UI macro.

Register a custom UI macro

Create a custom UI macro and code when you register it against a selected UI extension point.

Before you begin

Role required: application-specific developer, application-specific admin, or admin

Note: To learn about application-specific administrator roles and delegated development, see Access control rules in application administration apps and Delegated development and deployment.

Procedure

1. Navigate to System Extension Points > UI Extension Points.

2. In the UI Extension Point [sys_ui_extension-point] table, select the extension point that you want to register a custom UI macro against.

   Note: This listing includes UI extension points that you created and any pre-defined UI extension points that are embedded in Now Platform functions (for example, Customer Service Management and Field Service Management).

   To learn more about functions that contain pre-defined extension points, see the Related Topics links in Using extension points to extend application functionality.

3. In the UI Extension Point record, click the Create Implementation related link to create a new custom UI macro and register it against the selected UI extension point.

4. In the UI Macro form, create the custom code.

   To learn more about creating UI macros, see UI Macros.

5. Click Update.

Results

The custom UI macro is created and registered against the UI extension point.
Review UI extension instances

Review the listing of custom UI macros that are registered against a UI extension point. Extension instances indicate the application scope in which registered custom UI macros are used and provides a link to each. You can also activate or inactivate an extension instance.

Before you begin
Role required: You must have the specific role for the developer or administrator of the application, or you must have the admin role.

⚠️ Note: To learn about application-specific administrator roles and delegated development, see Access control rules in application administration apps and Delegated development and deployment.

Procedure

1. Navigate to System Extension Points > UI Extension Points.
2. In the UI Extension Point [sys_ui_extension_point] table, select the UI extension point that you want to view extensive instances for.
3. In the Implementations related list of the UI Extension Point record, select a UI extension instance.
   The system displays the UI Extension Instance record.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point</td>
<td>Unique name for the UI extension point. The name can have up to 100 alphanumeric characters, including special characters.</td>
</tr>
<tr>
<td>Class</td>
<td>UI macro in the application code in which the UI extension point is implemented.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope against which the extension point is assigned. This is system-assigned and cannot be changed. For more information about the protections that are offered by the use of scoping, see Application scope.</td>
</tr>
<tr>
<td>Active</td>
<td>Active extension point in the current instance. To inactivate an extension point, clear the check box.</td>
</tr>
</tbody>
</table>

4. Click Submit.
Related information

UI Macros
Using extension points to extend application functionality

Using client extension points in client-side UI scripting

Use client extension points in client-side UI scripting to modify forms, so that you do not have to directly modify the form. By using client extension points, you can integrate customizations without actually altering the core components in the application code.

You create client extension points and add them to the UI scripts in the base application code. When customizing a base application, implement the client extension points by creating the custom UI scripts and registering them against the client extension points.

Note: You create and register the client extension points in a manner that is similar to how you implement the scripted and UI extension points. For a detailed example of a scripted extension point, and to learn more about this process, see Using extension points to extend application functionality and Using UI extension points in server-side UI macros.

Creating and adding a client extension point

Create client extension points and add them to the client-side UI scripts in the base application code.

Use a client extension point to designate the specific location where data or objects can be sent to a registered custom UI script and where returned results are processed.

Create a client extension point

Create a client extension point that you can place in the UI scripts in the application code. By placing the client extension point, you designate the specific location in the application code where data or objects can be sent to a customization, and where data is returned.

Before you begin

Role required: You must have the specific role for the developer or administrator of the application, or you must have the admin role.

Note: To learn about application-specific administrator roles and delegated development, see Access control rules in application administration apps and Delegated development and deployment.
About this task
Define the content and structure for an extension point. This definition should describe how a customization should be structured, how it interacts with the application code, and how the data that is returned by a customization at the extension point is processed.

By creating a good definition, you provide a structure for your extension points.

Procedure
1. Navigate to **System Extension Points > Client Extension Points**.
2. Click **New**.
3. On the form, fill in the fields.

### Client Extension Point form

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Unique name for the extension point. The name can be up to 100 alphanumeric characters, including special characters.</td>
</tr>
<tr>
<td>API Name</td>
<td>Name of the extension point API that is pre-pended with the application scope to which it applies. This is a system-assigned name and cannot be changed.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope against which the extension point is assigned. This is system-assigned and cannot be changed. For more information about the protections that are offered by the use of scoping, see Application scope.</td>
</tr>
<tr>
<td>Restrict to this scope</td>
<td>Option for restricting the extension point to the application scope only.</td>
</tr>
<tr>
<td>Allow access over AJAX/REST</td>
<td>Option for enabling access to the client extension point over AJAX or REST. To learn more about these development architectures and techniques, see AJAX and REST API.</td>
</tr>
</tbody>
</table>
### Field | Description
--- | ---
Description | Requirements for the custom UI script, such as how the UI script should be structured and how it should operate with the application code.
Example | Example of how a custom UI script that interacts with this extension point should be structured so it operates with the application code.

4. Click **Submit**.

### Add a client extension point in the base application code

Add the UI extension point into a UI script in the base application code. To add the extension point, place an API call that identifies the location at which registered custom artifacts execute.

#### Before you begin

You must first create a UI extension point before you can add it to a UI script in the application code. Role required: You must have the specific role for the developer or administrator of the application, or you must have the admin role.

⚠️ **Note:** To learn about application-specific administrator roles and delegated development, see Access control rules in application administration apps and Delegated development and deployment.

#### Procedure

1. In the application code, access the existing UI script that you want to add a scripted extension point to, or create a new one. To learn more about creating UI scripts, see UI scripts.

2. In the UI script, add a line of code at the location that you expect to collect and process the custom UI script output. The code must contain the `getExtension` command that identifies the client extension point against which the custom UI scripts are registered. This line of code is similar to what is in an application code script include for scripted extension points. For an example, see Register a custom script include.

### Registering custom UI scripts against the client extension points

Register custom UI scripts against a selected UI extension point in the application code.
By registering custom UI scripts, you create an extension instance record that links the client extension point definition to its implementation in the custom UI script.

Register a custom UI script
Create a custom UI script and code when you register it against a selected client extension point.

Before you begin
Role required: You must have the specific role for the developer or administrator of the application, or you must have the admin role.

Note: To learn about application-specific administrator roles and delegated development, see Access control rules in application administration apps and Delegated development and deployment.

Procedure
1. Navigate to System Extension Points > Client Extension Points.
2. In the Client Extension Point [sys_client_extension_point] table, select the client extension point that you want to register the custom UI script against.

Note: This listing includes the client extension points that you define and the pre-defined client extension points that are embedded in Now Platform® functions (for example, Customer Service Management and Field Service Management).

To learn more about functions that contain the pre-defined extension points, see the Related Topics links in Using extension points to extend application functionality.

3. In the Client Extension Point record, click the Create Implementation related link to create a new custom UI script and register it against the selected client extension point.
4. In the UI Script form, create the custom code.
   To learn more about creating UI scripts, see UI scripts.
5. Click Update.

Results
A custom UI script is created and registered against the client extension point.

Review client extension instances
Review the listing of custom UI scripts that are registered against a client extension point. Extension instances indicate the application scope in which
registered custom UI scripts are used, and provides a link to each. You can also activate or inactive an extension instance.

**Before you begin**
Role required: You must have the specific role for the developer or administrator of the application, or you must have the admin role.

ℹ️ **Note:** To learn about application-specific administrator roles and delegated development, see Access control rules in application administration apps and Delegated development and deployment.

**About this task**
A listing of the custom script UI scripts that are registered against the client extension point appear at the bottom of the Extension Point record. It includes links to each extension instance, custom UI script, and application scope in which the registered custom UI script is used.

**Procedure**

1. Navigate to System Extension Points > Client Extension Points.
2. In the Client Extension Point [sys_client_extension_point] table, select the client extension point.
3. In the Implementations related list of the Client Extension Point record, select a client extension instance.

**Client Extension Instance record**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point</td>
<td>Unique name for the client extension point. The name can have up to 100 alphanumeric characters, including special characters.</td>
</tr>
<tr>
<td>Class</td>
<td>Base code UI script in which the client extension point is implemented.</td>
</tr>
<tr>
<td>Application</td>
<td>Application scope against which the extension point is assigned. This is system-assigned and cannot be changed. For more information about the protections that are offered by the use of scoping, see Application scope.</td>
</tr>
<tr>
<td>Active</td>
<td>Option that indicates an active extension point in the current instance.</td>
</tr>
</tbody>
</table>

4. Click Submit.
Related information

UI scripts
Using extension points to extend application functionality

JavaScript API reference

Use JavaScript APIs in scripts you write to change the functionality of applications, or when you create new applications.

AbstractDBObject - Global

Provides common methods for classes based on records in the database.
Use this script include as a base class to create your own database object class.

AbstractDBObject - isValid()

Determines if the current database record is valid.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the database record is valid, false otherwise.</td>
</tr>
</tbody>
</table>

The following example is from the AbstractDBObject definition in the Script Includes [sys_script_include] table.

```javascript
AbstractDBObject.prototype = {
    /*
    * Gets the record specified by the given source and table name. If the source is an
    * instance of GlideRecord, it is
    * simply returned. If the source is a string, then it is used as a sysID to find the
    * correct record, which is then
    * returned. If anything is wrong, a null will be returned.
    */
    _getRecord: function(source, tableName) {
        if (!source)
            return null;
    }
};
```
else if (source instanceof GlideRecord)
    return (source.getTableName() == tableName) ? source : null;
else {
    var gr = new GlideRecord(tableName);
    return gr.get('sys_id', source) ? gr : null;
}

isValid: function() {
    return this.valid;
},

/*AbstractDBObject*/

AbstractReconciler - Global

Use this script include as a base class to create your own Discovery reconciler.

This is an abstract base class for reconciling newly discovered data in an
XMLMemoryTable to a related list in the database. AbstractReconciler
implements the common logic and helper methods for reconciling related
lists for a particular CMDB item against discovered information present in a
MemoryTable instance.

Most of this logic is very straightforward, but one bit may not be: the resolution
of references to previously reconciled records. This feature works by collecting
information while reconciling one related list, then passing that information on
to the reconciler for another related list. For example, while switch partitions
are being reconciled, information is gathered (in a map instance) that maps
an interface number for a partition to the sys_id of the partition's record in
the partition table. Then later when the switchports are being reconciled, a
reference to the switch partition that contains the switchport can be resolved by
using this map.

AbstractReconciler - getReconciliationKey()

Gets the string value of the field to be used as a key to look up a record in the
newly discovered data.

This value will be used to attempt to find a record in the discovered data, using
the column returned by the getReconciliationKey() method.

Mandatory method, must be overridden in all concrete subclasses.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value to use as a key into the discovered data.</td>
</tr>
</tbody>
</table>

AbstractReconciler - getReconciliationField()

Determines if the newly discovered data is different than the data already in the database for the current item.

This method is only invoked for items that have been rediscovered.
Mandatory method, must be overridden in all concrete subclasses.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the data has changed.</td>
</tr>
</tbody>
</table>

AbstractReconciler - hasChanged()

Returns true if the newly discovered data is different than the data already in the database for this item.

This method is only invoked for items that have been rediscovered.
Mandatory method, must be overridden in all concrete subclasses.
AbstractReconciler - process()

Reconciles the newly discovered data with the data already present in the database and resolves references to previously reconciled data.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the discovered data is different than the data already in the database.</td>
</tr>
</tbody>
</table>

AbstractReconciler - markAbsent()

Marks CIs as absent by setting the install status to "Absent"; other tables are marked by setting the (presumed present) field "absent" to true.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
AbstractReconciler - markPresent()

Marks CIs as present by setting the install status to "Installed"; other tables are marked by setting the (presumed present) field "absent" to false.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

AbstractReconciler - readDatabaseFields()

Reads the current memory table record of newly discovered data and checks its validity.

Mandatory method, must be overridden in all concrete subclasses.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Boolean  True if the data is valid.
AbstractReconciler - readDiscovered()

Reads the current memory table record of newly discovered data and checks its validity.

Mandatory method, must be overridden in all concrete subclasses.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the data is valid.</td>
</tr>
</tbody>
</table>

AbstractReconciler - resolveReferenceFields()

Reads the fields in the current database record into instance member fields.

Mandatory method, must be overridden in all concrete subclasses.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

AbstractReconciler - track(String sys_id)

Optional method to override in concrete subclasses that need to resolve reference fields.

This method is invoked after valid discovered data is read. Generally implementations will use some of this discovered data as a key into a map (passed into the concrete reconciler class when it was instantiated) that will
return the sys_id of the database record holding the referenced information. However, implementations are not required to use this approach.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The sys_id of the current database record.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### AbstractReconciler - setDatabaseFields()

Sets the database fields for the current record to the values in the newly discovered information.

This method is invoked both for new database records and for existing database records when the discovered data has changed.

Mandatory method, must be overridden in all concrete subclasses.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### AbstractReconciler - setup()

Optional override for concrete subclasses that require special setup.
AbstractScriptProcessor - Global
Base JavaScript processor class that other JavaScript processors extend. Extend this class to create your own JavaScript processors.

AbstractScriptProcessor - initialize(String request, String response, String process)
Called by the Prototype JavaScript Framework during object construction.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request</td>
<td>String</td>
<td>The request</td>
</tr>
<tr>
<td>response</td>
<td>String</td>
<td>The response</td>
</tr>
<tr>
<td>process</td>
<td>String</td>
<td>The process</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

AbstractScriptProcessor - process()
Called by the Prototype JavaScript Framework during object processing. Do not call this method directly.
### AbstractTimelineSchedulePage - Global

A helper class to facilitate working with the data displayed on timelines.

Processing the data displayed within a timeline starts by executing the corresponding function of the specified script include. Like other script includes, the language syntax is JavaScript and follows the default security constraints of this type of resource. However, because the different types of display options are complex, a helper class was created. To display a timeline, extend the `AbstractTimelineSchedulePage` class and pair it with its corresponding schedule page. At a minimum, extending classes should override the `getItems()` method, as this is the primary event handler for returning items to be displayed on the client.

The client processes the data that `AbstractTimelineSchedulePage` returns in two phases. The first phase makes the actual updates to the timeline. Immediately after (if configured), the second phase displays a success message box, error message box, or dialog message prompt. In phase one, the available options for manipulating data include:

- **Do Not Update Any Items** - This is the default behavior. Do not perform any of the remaining steps in phase one.
- **Update With Specific Items** - This is done using: `add()`.
- **Render The Timeline Using the `getItems()` Function** - This is done using:
  ```javascript
  setDoReRenderTimeline(true);
  ```

**Note:** If both `TimelineItems` are returned and `setDoReRenderTimeline` is set to `true`, the system will ignore the `setDoReRenderTimeline` property and explicitly show only the `TimelineItems` that were added via the `add()` function.

The available options in phase two include:
• Do Not Display Any Message Boxes - This is the default behavior.
• Display a Success Dialog Box - This is done using: `setStatusSuccess()`.
• Display an Error Dialog Box - This is done using: `setStatusError()`.
• Display a Dialog Confirm Box - This is done using: `setStatusPrompt()`.

**Note:**

A script include class that extends `AbstractTimelineSchedulePage` automatically receives all Uri parameters from the original Url whose prefix begins with `sysparm_timeline_`. To access the values of these, use: `this.getParameter("sysparm_timeline_VARIABLE");` inside your extended class.

This is useful if you need to display a schedule page from a dynamic element, such as from a context menu from a list. By passing in dynamic data via the Url the schedule page will auto-include these parameters inside the Ajax calls and therefore will be accessible inside the `AbstractTimelineSchedulePage` Script includes.

### AbstractTimelineSchedulePage - add(Object ObjTimeline)

Adds a `TimelineItem` object that will be returned to the client and appropriately displayed on the timeline.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ObjTimeline</td>
<td>Object</td>
<td>The <code>TimelineItem</code> object to add to the timeline.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### AbstractTimelineSchedulePage - addSeparator()

Adds a horizontal frame separator into the list of timeline items. All future items added via `add()` will be added into the subsequent timeline frame.

All future items added via `add()` will be added into the subsequent timeline frame.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

// Inside of a script include that extends AbstractTimelineSchedulePage
   this.addSeparator();

### AbstractTimelineSchedulePage - elementMoveX(String spanSysId, String newStartDateTimeMs)

Event handler for when a timeline span moves horizontally.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spanSysId</td>
<td>String</td>
<td>The sys ID of the current span being adjusted.</td>
</tr>
<tr>
<td>newStartDateTimeMs</td>
<td>String</td>
<td>The new start time of the span in milliseconds. Make sure to parse the string using parseInt() before performing any numerical comparisons.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### AbstractTimelineSchedulePage - elementMoveY(String spanSysId, String itemSysId, String newItemSysId)

Event handler for when a timeline span moves vertically.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spanSysId</td>
<td>String</td>
<td>The sys ID of the current span being adjusted.</td>
</tr>
<tr>
<td>itemSysId</td>
<td>String</td>
<td>The sys ID of the timeline item associated with the current span.</td>
</tr>
<tr>
<td>newItemSysId</td>
<td>String</td>
<td>The sys ID of the timeline item (a row) that the current span was dragged into.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

AbstractTimelineSchedulePage - elementMoveXY(String spanSysId, String itemSysId, String newItemSysId, String newItemSysId, String newStartDateTimeMs)

Event handler for when a timeline span moves both horizontally and vertically.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spanSysId</td>
<td>String</td>
<td>The sys ID of the current span being adjusted.</td>
</tr>
<tr>
<td>itemSysId</td>
<td>String</td>
<td>The sys ID of the timeline item associated with the current span.</td>
</tr>
<tr>
<td>newItemSysId</td>
<td>String</td>
<td>The sys ID of the timeline item (a row) that the current span was dragged into.</td>
</tr>
<tr>
<td>newStartDateTimeMs</td>
<td>String</td>
<td>The new start time of the span in milliseconds. Make sure to parse the string using parseInt() before performing any numerical comparisons.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
AbstractTimelineSchedulePage - elementSuccessor(String spanSysId, String newSuccSpanId)

Event handler for when a timeline relationship has been created between two spans.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spanSysId</td>
<td>String</td>
<td>The sys ID of the current span which will be a predecessor for the newly created relationship.</td>
</tr>
<tr>
<td>newSuccSpanId</td>
<td>String</td>
<td>The sys ID of the successor span from the relationship created.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>ReturnValue</td>
</tr>
</tbody>
</table>

AbstractTimelineSchedulePage - elementTimeAdjustStart(String spanSysId, String newStartDateTimeMs)

Event handler for when a timeline span's start date was modified.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spanSysId</td>
<td>String</td>
<td>The sys ID of the current span that is being adjusted.</td>
</tr>
<tr>
<td>newStartDateTimeMs</td>
<td>String</td>
<td>The new start time of the span in milliseconds. Make sure to parse the string using parseInt() prior to performing any numerical comparisons.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
AbstractTimelineSchedulePage - elementTimeAdjustEnd(String spanSysId, String newEndDateTimeMs)

Event handler for when a timeline span's end date was modified.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spanSysId</td>
<td>String</td>
<td>The sys ID of the current span that is being adjusted.</td>
</tr>
<tr>
<td>newEndDateTimeMs</td>
<td>String</td>
<td>The new end time of the span in milliseconds. Make sure to parse the string using parseInt() prior to performing any numerical comparisons.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

AbstractTimelineSchedulePage - getItems()

Event handler for returning schedule items to display on the timeline.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

AbstractTimelineSchedulePage - inputBox(String strInputText)

Event handler for when a string was typed into the left pane input box.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strInputText</td>
<td>String</td>
<td>The text that was entered in the input box in the left pane.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

AbstractTimelineSchedulePage - itemMove(String itemSysId, String newItemSysId)

Event handler for when a timeline row item was moved and dragged into another row item.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>itemSysId</td>
<td>String</td>
<td>The sys ID of the timeline item associated with the current span.</td>
</tr>
<tr>
<td>newItemSysId</td>
<td>String</td>
<td>The sys ID of the timeline item (a row) that the current span was dragged into.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

AbstractTimelineSchedulePage - setDoReRenderTimeline(Boolean b)

Specifies whether or not to re-render all of the timeline items using the `getItems()` function.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>If true, re-render the timeline by making a new event call to the server's <code>getItems()</code> handler.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

AbstractTimelineSchedulePage - setPageTitle(String strTitle)

Specifies the text to display as the title of the timeline.

The page title can be set (and updated) from any interactive event; however, is recommended to be set during the `getItems()` event.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strTitle</td>
<td>String</td>
<td>The text to be displayed on the timeline title.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

AbstractTimelineSchedulePage - setStatusError(String strTitle, String strMessage)

Sets the current event request to an error status with a specified title and message to display in phase two of the `GlideTimeline` event processing.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strTitle</td>
<td>String</td>
<td>The text to be displayed in the dialog box title.</td>
</tr>
<tr>
<td>strMessage</td>
<td>String</td>
<td>The text to be displayed within the dialog box. The text can contain HTML formatting.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
AbstractTimelineSchedulePage - setStatusPrompt(String strTitle, String strOkFunction, String strCancelFunction, String strCloseFunction)

Sets the current event request to a prompt error status with a specified title and message to display in a confirmation dialog box during phase two of the GlideTimeline event processing. The confirmation box displays an **OK** and **Cancel** button that each generate new events that will call the functions specified the parameter arguments. The custom defined functions for **OK**, **Cancel**, and **Close** receive the same parameter arguments as those for the current event.

The confirmation box displays an **OK** and **Cancel** button that each generate new events that will call the functions specified the parameter arguments.

⚠️ **Note:** The custom defined functions for **OK**, **Cancel**, and **Close** receive the same parameter arguments as those for the current event.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>strTitle</td>
<td>String</td>
<td>The text to display in the dialog box. The text can contain HTML formatting.</td>
</tr>
<tr>
<td>strOkFunction</td>
<td>String</td>
<td>The function to call in the current script include class if the users clicks the <strong>OK</strong> button.</td>
</tr>
<tr>
<td>strCancelFunction</td>
<td>String</td>
<td>The function to call in the current script include class if the user clicks the <strong>Cancel</strong> button.</td>
</tr>
<tr>
<td>strCloseFunction</td>
<td>String</td>
<td>The function to call in the current script include class if the user clicks the <strong>Close</strong> button.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var MyTimelineScriptIncludeClass = Class.create();
    MyTimelineScriptIncludeClass.prototype =
    Object.extendsObject(AbstractTimelineSchedulePage, { }

    getItems: function() {
        //...
    },
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
elementTimeAdjustEnd: function(spanSysId, newEndDateTimeMs) {
  // Display a status prompt dialog box
  this.setStatusPrompt('Confirm Action', 'Are you sure you want to do that?',
    'this._myOkHandlerFunction',
    'this._myCancelHandlerFunction',
    'this._myCloseHandlerFunction');
},

_myOkHandlerFunction: function(spanSysId, newEndDateTimeMs) { // ... },

_myCancelHandlerFunction: function(spanSysId, newEndDateTimeMs) { // ... },

_myCloseHandlerFunction: function(spanSysId, newEndDateTimeMs) { // ... }

AbstractTimelineSchedulePage - setStatusSuccess(String strTitle,
String strMessage)

Sets the current event request to a success status with a specified title and
message to display in phase two of the GlideTimeline event processing.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strTitle</td>
<td>String</td>
<td>The text to be displayed in the dialog box title.</td>
</tr>
<tr>
<td>strMessage</td>
<td>String</td>
<td>The text to be displayed within the dialog box. The text can contain HTML formatting.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

AccAgentsAPI API - Scoped

The AccAgentsAPI Script include enables performing management actions on available agents.
This API requires the Agent Client Collector Framework (sn_agent) store application and is provided within the \textit{sn_agent} namespace. For more information, refer to Agent Client Collector.

For the REST API solution, refer to Agent Client Collector API.

This API includes methods that enable the following:

- Getting extensive information of one or more agents.
- Submitting a request to grab an agent log and retrieving information about the request progress.
- Starting or stopping data collection.
- Restarting an agent.
- Running discovery on an agent.

\textbf{AccAgentsAPI - AccAgentsAPI()}

Creates an AccAgentsAPI instance.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

The following example shows how to initialize AccAgentsAPI.

```javascript
var agentsApi = new sn_agent.AccAgentsAPI();
```

\textbf{AccAgentsAPI - checkGrabLogRequestProgress(String requestId)}

Checks the status of a grab log request.

Run the \texttt{submitGrabLogRequest()} method to get a request ID.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>requestId</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;</code></td>
<td>JSON object containing the grab log request status.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;output&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>status</td>
<td>Number indicating status of the grab log request. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Grab log request is complete.</td>
</tr>
<tr>
<td></td>
<td>• 1: Grab log request in progress.</td>
</tr>
<tr>
<td></td>
<td>• 2: Grab log request timed out.</td>
</tr>
<tr>
<td></td>
<td>• 3: Grab log request has an error.</td>
</tr>
<tr>
<td></td>
<td>• 4: Grab log request was not found.</td>
</tr>
<tr>
<td>output</td>
<td>Information describing the status.</td>
</tr>
</tbody>
</table>

The following example shows how to use a request ID to get the status of a grab log request.

```javascript
var agentsApi = new sn_agent.AccAgentsAPI();
var logRequestStatus = agentsApi.checkGrabLogRequestProgress("<request_ID>");
gs.info(JSON.stringify(logRequestStatus, null, 2));
```

Output:

```javascript
{
   "status": 2,
   "output": "Grab Log Request Timed Out"
}
```

**AccAgentsAPI - getAgent(String agentID)**

Gets the information of a specified agent.

To get a list of agent IDs:
• Run the `getAgentsList()` method.
• Check the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table.
• Run the Agent Client Collector GET list REST API.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentID</td>
<td>String</td>
<td>Unique ID of an agent listed in the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;</td>
<td>Object containing extended agent information.</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>error</td>
<td>Error message. Null if there is no error.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>agent</td>
<td>```</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>agent.agent_id</td>
<td>ID of the agent as submitted.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Properties</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>agent.data_collection</td>
<td>Data collection indicates whether scheduled checks are to be run. These checks are a part of the policies scheduled for this agent to run. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: On – Checks run as scheduled.</td>
</tr>
<tr>
<td></td>
<td>• 1: Off (manual) – Checks have been disabled manually.</td>
</tr>
<tr>
<td></td>
<td>• 2: Off (auto) – Checks have been disabled automatically due to high CPU consumption by the agent.</td>
</tr>
<tr>
<td>Data type: Number</td>
<td></td>
</tr>
<tr>
<td>agent.ip_address</td>
<td>Agent IP address.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>agent.is_duplicate</td>
<td>Flag that indicates whether this agent is a duplicate of another. There should be only a single agent on a given host. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: The agent has the same host as an Alive/Up agent with a different agent ID. Turn off or uninstall the duplicate</td>
</tr>
<tr>
<td></td>
<td>• false: This agent has no duplicates in the Alive/Up state.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>agent.is_restart_enabled</td>
<td>Flag that indicates whether restart is enabled. Agent restart is not configurable. It depends on the OS and the version of the OS the agent is running on. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Restart is enabled for this agent.</td>
</tr>
<tr>
<td></td>
<td>• false: Restart is disabled for this agent.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>agent.name</td>
<td>Name of the agent.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent.number_of_running_checks</td>
<td>The number of checks the agent is scheduled to run. These checks are a part of the policies scheduled for this agent to run. Data type: Number</td>
</tr>
<tr>
<td>agent.status</td>
<td>Status of the agent. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Alive/Up – The agent is active.</td>
</tr>
<tr>
<td></td>
<td>• 1: Warning – The agent has not received a keep-alive message in the past few minutes.</td>
</tr>
<tr>
<td></td>
<td>• 2: Down – The agent has not received a keep-alive message in a long time.</td>
</tr>
<tr>
<td></td>
<td>• 3: Restarting – The agent is restarting.</td>
</tr>
<tr>
<td>agent.up_since</td>
<td>UTC time since the agent’s status became alive/up. The value is in GlideDateTime format. Data type: String</td>
</tr>
<tr>
<td>agent.version</td>
<td>Version of Agent Client Collector the agent is running. Data type: String</td>
</tr>
</tbody>
</table>

The following example shows how to display an agent’s status.

```javascript
var agentsApi = new sn_agent.AccAgentsAPI();
var agentInfo = agentsAPI.getAgent("<agent_ID>");

if (!gs.nil(agentInfo.error))
gs.error(agentInfo.error);
else
  gs.info("agent status: " + agentInfo.agent.status);
```

Output:

agent status: 2

The following example shows how to get all agent details.
var agentsApi = new sn_agent.AccAgentsAPI();
var agentInfo = agentsAPI.getAgent("<agent_ID>" );
gs.info(JSON.stringify(agentInfo, null, 2));

Output:

```json
{
  "error": null,
  "agent": {
    "name": "win2016-dc-64bit",
    "status": 0,
    "agent_id": "<agent_ID>",
    "ip_address": "10.222.333.42",
    "number_of_running_checks": 1,
    "data_collection": 0,
    "is_restart_enabled": true,
    "is_duplicate": false,
    "up_since": "2021-03-24 11:04:38",
    "version": "2.4.0"
  }
}
```

AccAgentsAPI – getAgentsList(String encodedQuery, Number limit)

Gets a list of agents with related information.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>encodedQuery</td>
<td>String</td>
<td>Encoded query string in standard Glide format. See Encoded query strings.</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td>Optional. Restricts results to a maximum number of agents. Use null or undefined for both if they are not required. Default/Max: 20,000</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Array&gt;</td>
<td>Array of JSON objects containing extended agent information.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;agent_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;data_collection&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;ip_address&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;is_duplicate&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;is_restart_enabled&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;number_of_running_checks&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;up_since&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;version&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

agent_id          | ID of the agent as submitted.                                               |
|                   | Data type: String                                                           |

data_collection   | Data collection indicates whether scheduled checks are to be run. These checks are a part of the policies scheduled for this agent to run. Possible values: |
|                   | • 0: On – Checks run as scheduled.                                          |
|                   | • 1: Off (manual) – Checks have been disabled manually.                     |
|                   | • 2: Off (auto) – Checks have been disabled automatically due to high CPU consumption by the |
|                   | Data type: Number                                                           |

ip_address         | Agent IP address.                                                           |
|                   | Data type: String                                                           |

is_duplicate       | Flag that indicates whether this agent is a duplicate of another. There should be only a single agent on a given host. Possible values: |
## Returns (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
|                           | • true: The agent has the same host as an Alive/Up agent with a different agent ID. Turn off or uninstall the duplicate  
|                           | • false: This agent has no duplicates in the Alive/Up state.  
|                           | Data type: Boolean                                                                                                                                                                                          |
| is_restart_enabled        | Flag that indicates whether restart is enabled. Agent restart is not configurable. It depends on the OS and the version of the OS the agent is running on.  
|                           | Possible values:  
|                           | • true: Restart is enabled for this agent.  
|                           | • false: Restart is disabled for this agent.  
|                           | Data type: Boolean                                                                                                                                                                                          |
| name                      | Name of the agent.  
|                           | Data type: String                                                                                                                                                                                          |
| number_of_running_checks  | The number of checks the agent is scheduled to run. These checks are a part of the policies scheduled for this agent to run.  
|                           | Data type: Number                                                                                                                                                                                          |
| status                    | Status of the agent.  
|                           | Possible values:  
|                           | • 0: Alive/Up – The agent is active.  
|                           | • 1: Warning – The agent has not received a keep-alive message in the past few minutes.  
|                           | • 2: Down – The agent has not received a keep-alive message in a long time.  
|                           | • 3: Restarting – The agent is restarting.  
|                           | Data type: Number                                                                                                                                                                                          |
| up_since                  | UTC time since the agent’s status became alive/up. The value is in GlideDateTime format.  
                                                                                                                                                         |
## Returns (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>Version of Agent Client Collector the agent is running. Data type: String</td>
</tr>
</tbody>
</table>

The following example shows how to restrict results by query and number. The query returns all agents that are not in the down state with a maximum of two results.

```javascript
var agentsApi = new sn_agent.AccAgentsAPI();
var agentList = agentsApi.getAgentsList("agent_extended_info.status!=2", 2);

gs.info(JSON.stringify(agentList, null, 2));
```

**Output:**

```
[
  {
    "name": "007-175",
    "status": 0,
    "agent_id": "007-175",
    "ip_address": "11.222.63.66",
    "number_of_running_checks": 0,
    "data_collection": 0,
    "is_restart_enabled": false,
    "is_duplicate": false,
    "up_since": "2021-03-24 14:36:45",
    "version": "2.4.0"
  },
  {
    "name": "win2016-dc-64bit",
    "status": 0,
    "agent_id": "007-64",
    "ip_address": "10.222.333.42",
    "number_of_running_checks": 1,
    "data_collection": 0,
    "is_restart_enabled": true,
    "is_duplicate": false,
    "up_since": "2021-03-24 11:04:38",
    "version": "2.4.0"
  }
]```
The following example shows how to list every agent in the system. This example uses no query and no maximum number of results.

```javascript
var agentsApi = new sn_agent.AccAgentsAPI();
var agentList = agentsApi.getAgentsList(null, 0);
gs.info(JSON.stringify(agentList, null, 2));
```

The following example shows how to iterate over the results provided and displays each agent ID.

```javascript
var agentsApi = new sn_agent.AccAgentsAPI();
var agentsList = agentsApi.getAgentsList(null, 0);
for (var i = 0; i < agentsList.length; i++)
    gs.info("agent with id: " + agentsList[i].agent_id);
```

Output:

```
sn_agent: agent with id: 000a00e0aa1aa3a4
sn_agent: agent with id: 000a00e1aa1aa3a4
sn_agent: agent with id: 000a00e2aa1aa3a4
```

**AccAgentsAPI - restartAgent(String agentID)**

Restarts a specified agent with alive/up status.

If Agent Client Collector performance issues occur, you can restart the agent. Manual restart is supported in the following environments:

- Linux-based agents using **systemd**
- Windows agents

To get a list of agent IDs:

- Run the `getAgentsList()` method.
- Check the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table.
- Run the **Agent Client Collector GET list** REST API.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentID</td>
<td>String</td>
<td>Unique ID of an agent listed in the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Error message if applicable, null otherwise.</td>
</tr>
</tbody>
</table>

The following example shows how to restart an agent.

```javascript
var agentsApi = new sn_agent.AccAgentsAPI();

var err = agentsApi.restartAgent("<agent_ID>");
if (!gs.nil(err))
gs.error(err);
```

**AccAgentsAPI - runDiscovery(String agentID)**

Runs a discovery check to locate CIs related to an agent. The specified agent must be in alive/up status.

To get a list of agent IDs:
- Run the `getAgentsList()` method.
- Check the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table.
- Run the Agent Client Collector GET list REST API.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentID</td>
<td>String</td>
<td>Unique ID of an agent listed in the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Error message if applicable, null otherwise. For example, Agent With ID: &lt;agentID&gt; Is Not Up: no thrown error.</td>
</tr>
</tbody>
</table>

The following example shows how to run discovery on an agent with alive/up status.

```
var agentsApi = new sn_agent.AccAgentsAPI();

var err = agentsApi.runDiscovery("<agent_ID>");

if (!gs.nil(err))
  gs.error(err);
```

AccAgentsAPI - setDataCollectionStatus(String agentID, Boolean status)

Set the given data collection status (true/false if enabled or not) for a specified agent.

To get a list of agent IDs:
- Run the `getAgentsList()` method.
- Check the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table.
- Run the Agent Client Collector GET list REST API.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentID</td>
<td>String</td>
<td>Unique ID of an agent listed in the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table.</td>
</tr>
</tbody>
</table>
| status | Boolean      | Flag that indicates whether data collection is enabled for the agent. Valid values:  
  • true: Enables data collection for this agent.  
  • false: Disables data collection for this agent.  
  Default: true |
### AccAgentsAPI - submitGrabLogRequest(String agentId)

Requests the log of a specified agent with alive/up status.

**Note:** To retrieve the log and check its progress, pass the request ID returned to the `checkGrabLogRequestProgress()` method.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentID</td>
<td>String</td>
<td>Unique ID of an agent listed in the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;</td>
<td>JSON object containing the request ID and any error information.</td>
</tr>
</tbody>
</table>

```javascript
var agentsApi = new sn_agent.AccAgentsAPI();
var err = agentsApi.setGrabLogRequest("<agentID>";' true);
if (!gs.nil(err))
    gs.error(err);
```

The following example shows how to turn on agent data collection.

```javascript
var agentsApi = new sn_agent.AccAgentsAPI();
var err = agentsApi.setDataCollectionStatus("<agentID>", true);
if (!gs.nil(err))
    gs.error(err);
```

The following example shows how to turn off agent data collection.

```javascript
var agentsApi = new sn_agent.AccAgentsAPI();
var err = agentsApi.setDataCollectionStatus("<agentID>", false);
if (!gs.nil(err))
    gs.error(err);
```
Returns (continued)

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message. Null if there is no error. Data type: String</td>
</tr>
<tr>
<td>request_id</td>
<td>Sys_id of a request in the Agent Client Collector Requests [sn_agent_request] table. You can use this ID to get the status of the request using GET /agents/{request_id}/. Data type: String</td>
</tr>
</tbody>
</table>

The following example shows how to get a log request ID.

```javascript
var agentsApi = new sn_agent.AccAgentsAPI();
var submittedRequest = agentsApi.submitGrabLogRequest("<agentID>");
if (!gs.nil(submittedRequest.error))
    gs.error(submittedRequest.error);
else
    gs.info("Request ID: " + submittedRequest.request_id);
```

Output:

Request ID: <sys_id>

AccCheckDefsAPI API - Scoped

The AccCheckDefsAPI script include enables managing check definitions and associated parameters.

This API requires the Agent Client Collector Framework (sn_agent) store application and is provided within the sn_agent namespace. This API requires the agent_client_collector_admin role. For more information, refer to Agent Client Collector.

This API does not have a constructor for creating an instance. Call each method using the AccCheckDefsAPI static class in the following format:

```javascript
sn_agent.AccCheckDefsAPI.<method>
```

For the REST API solution, refer to Agent Client Collector API.

This API includes methods that enable the following:
• Get a specified check definition
• Get a list of check definitions
• Update checks and check parameters

AccCheckDefsAPI - getCheck(String checkDefId, Boolean withParams)

Gets a specified check definition with details.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>checkDefId</td>
<td>String</td>
<td>Sys_id of the check definition listed in the Check Definitions [sn_agent_check_def] table.</td>
</tr>
<tr>
<td>withParams</td>
<td>Boolean</td>
<td>Flag that indicates whether existing check parameter details are returned. Information for each standard and secure check parameter is included in a JSON object. Valid values: • true: Return check parameter details. • false: Do not return check parameter details. Default: False</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check</td>
<td>Details of the specified check definition.</td>
</tr>
</tbody>
</table>

"check ": {
    "background": Boolean,
    "check_group": "String",
    "check_type": "String",
    "command": "String",
    "error": "String",
    "name": "String",
    "params": [Array],
    "plugins": [Array],
    "proxy_valid": Boolean,
    "secure_params": [Array],
    "sys_id": "String",}
<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;timeout&quot;: Number</td>
<td></td>
</tr>
</tbody>
</table>
| check.background         | Flag that indicates whether this check definition is a background check. A background check is a check which the agent starts execution of and doesn’t wait for it to finish running. Valid values:
- true: This check definition is a background check.
- false: This check definition is not a background check.
Data type: Boolean |
| check.check_group        | Group specified for this check definition. Data type: String |
| check.check_type         | Type of check. Possible values:
- Events – Check results are transformed into an Event Management event.
- Metrics – Values from the check result are transformed to metrics.
Data type: String |
| check.command            | Command that the Agent Client Collector executes. Data type: String |
| check.error              | Message if there is an error. Null otherwise. Data type: String |
| check.name               | Name of the check. Data type: String |
| check.params             | List of parameter definitions associated with the check definition. These results are only included if the withParams parameter is set to true. |

"params": [
]
## Returns (continued)

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
</table>
|              | ```json{language:JavaScript}
{
"active": Boolean,
"default_value": "String",
"mandatory": Boolean,
"name": "String",
"sys_id": "String"
}
``` |
|              | Data type: Array                                                            |
|              | check.params.active Flag that indicates whether the check parameter is active. Valid values:  |
|              | • true: The check parameter is active.                                      |
|              | • false: The check parameter is inactive.                                   |
|              | Data type: Boolean                                                          |
|              | check.params.default_value Specifies the default value for this check parameter. Data type: String |
|              | check.params.mandatory Flag that indicates whether the check parameter is required. Valid values:  |
|              | • true: The check parameter is required.                                    |
|              | • false: The check parameter is optional.                                   |
|              | Data type: Boolean                                                          |
|              | check.params.name Name of the check parameter.                             |
|              | Data type: String                                                           |
|              | check.params.sys_id Sys_id of the check parameter listed in the Check Secure Parameter Definitions [sn_agent_check_param_def] table. Data type: String |
|              | check.plugins List of [Agent Client Collector plugins] associated with this check. Data type: Array |
## Returns (continued)

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check.proxy_valid</td>
<td>Flag that indicates whether the check definition policy is set to work as a proxy. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: This check definition policy is set to work as a proxy.</td>
</tr>
<tr>
<td></td>
<td>• false: This check definition policy is not set to work as a proxy.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>check.secure_params</td>
<td>List of secure parameters assigned to this check. These results are only included if the withParams parameter is set to true.</td>
</tr>
<tr>
<td></td>
<td>&quot;secure_params&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;active&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;order&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>check.secure_params.active</td>
<td>Flag that indicates whether the secure parameter is active. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The secure parameter is active.</td>
</tr>
<tr>
<td></td>
<td>• false: The secure parameter is inactive.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>check.secure_params.name</td>
<td>Name of the secure parameter.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>check.secure_params.order</td>
<td>Order in which the parameter is sent to the check command/script.</td>
</tr>
<tr>
<td>Data type: Number</td>
<td></td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check.secure_params.sys_id</td>
<td>Sys_id of the secure parameter listed in the Check Secure Parameter Definitions [sn_agent_check_secure_param_def] table. Data type: String</td>
</tr>
<tr>
<td>check.sys_id</td>
<td>Sys_id of the check definition listed in the Check Definitions [sn_agent_check_def] table. Data type: String</td>
</tr>
<tr>
<td>check.timeout</td>
<td>Timeout in seconds. Data type: Number</td>
</tr>
</tbody>
</table>

The following example shows how to get information for a specified check definition.

```javascript
var checkDefId = "94436b237f705300f128134f8dfa91a4";
var withParams = true;

var checkDef = sn_agent.AccCheckDefsAPI.getCheck(checkDefId, withParams);

gs.info(JSON.stringify(checkDef, null, 2));
```

Output:

```json
{
  "check": {
    "name": "app.apache.metrics-apache",
    "command": "metrics-apache-graphite.rb -p {{.labels.params_port}} --path
{{.labels.params_path}} -h {{.labels.params_host}}",
    "plugins": [
      "monitoring-plugin-common"
    ],
    "timeout": 60,
    "proxy_valid": true,
    "background": false,
    "check_type": "Metrics",
    "check_group": "Apache",
    "sys_id": "94436b237f705300f128134f8dfa91a4",
    "params": [
      {
        "name": "port",
```
"active": true,
"mandatory": true,
"default_value": "80",
"sys_id": "58436b237f05300f128134f8dafa91a8"
},
{
"name": "path",
"active": true,
"mandatory": true,
"default_value": "/server-status?auto",
"sys_id": "98436b237f05300f128134f8dafa91aa"
},
{
"name": "scheme",
"active": false,
"mandatory": false,
"default_value": null,
"sys_id": "a4e57a96db3bb4035305c55dc9619f6"
},
{
"name": "host",
"active": true,
"mandatory": true,
"default_value": "127.0.0.1",
"sys_id": "d4436b237f05300f128134f8dafa91a6"
},
{
"name": "ssl_secure_connection",
"active": false,
"mandatory": false,
"default_value": null,
"sys_id": "e3b272c4530100106ffeddeeff7b1275"
}
],
"secure_params": [
{
"name": "cred_user_name",
"active": true,
"order": 1,
"sys_id": "2494cd6e531700106ffeddeeff7b1273"
},
{
"name": "cred_password",
"active": true,
AccCheckDefsAPI - getChecksList(String encodedQuery, Number limit, Boolean withParams)

Gets a list of check definitions with details.

See also Checks and policies.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>encodedQuery</td>
<td>String</td>
<td>An encoded query string to filter the check definition result list. Use null for an unfiltered list of check definitions in the system.</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td>Limits the number of returned records. Set to null to use the default value. Default: 20,000</td>
</tr>
</tbody>
</table>
| withParams       | Boolean| Flag that indicates whether existing check parameter details are returned. Information for each standard and secure check parameter is included in a JSON object. Valid values:  
• true: Return check parameter details.  
• false: Do not return check parameter details. Default: False |

**Returns**

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check definitions</td>
<td>List of check definition and details provided as JSON objects.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>[</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;background&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;check_group&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;check_type&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;command&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;params&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;plugins&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;proxy_valid&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;secure_params&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;timeout&quot;: Number</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>background</td>
<td>Flag that indicates whether this check definition is a background check. A background check is a check which the agent starts execution of and doesn’t wait for it to finish running. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: This check definition is a background check.</td>
</tr>
<tr>
<td></td>
<td>• false: This check definition is not a background check.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type</strong>: Boolean</td>
</tr>
<tr>
<td>check_group</td>
<td>Group specified for this check definition.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type</strong>: String</td>
</tr>
<tr>
<td>check_type</td>
<td>Type of check. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Events – Check results are transformed into an Event Management event.</td>
</tr>
<tr>
<td></td>
<td>• Metrics – Values from the check result are transformed to metrics.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type</strong>: String</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| **command** | Command that the Agent Client Collector executes.  
Data type: String |
| **name** | Name of the check.  
Data type: String |
| **params** | List of parameter definitions associated with the check definition. These results are only included if the `withParams` parameter is set to true.  

```
"params": [  
{
    "active": Boolean,
    "default_value": "String",
    "mandatory": Boolean,
    "name": "String",
    "sys_id": "String"
}
]
```

Data type: Array |

| **params.active** | Flag that indicates whether the check parameter is active.  
Valid values:  
• true: The check parameter is active.  
• false: The check parameter is inactive.  
Data type: Boolean |
| **params.default_value** | Specifies the default value for this check parameter.  
Data type: String |
| **params.mandatory** | Flag that indicates whether the check parameter is required.  
Valid values:  
• true: The check parameter is required.  
• false: The check parameter is optional.  
Data type: Boolean |
| **params.name** | Name of the check parameter. |
## Returns (continued)

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>params.sys_id</code></td>
<td>Sys_id of the check parameter listed in the Check Secure Parameter Definitions [sn_agent_check_param_def] table. Data type: String</td>
</tr>
<tr>
<td><code>plugins</code></td>
<td>List of Agent Client Collector plugins associated with this check. Data type: Array</td>
</tr>
</tbody>
</table>
| `proxy_valid`      | Flag that indicates whether the check definition policy is set to work as a proxy. Valid values: Valid values:  
                      | • true: This check definition policy is set to work as a proxy.  
                      | • false: This check definition policy is not set to work as a proxy. Data type: Boolean                                               |
| `secure_params`    | List of secure parameters assigned to this check. These results are only included if the withParams parameter is set to true.  
                      | "secure_params": [  
                      |   {  
                      |     "active": Boolean,  
                      |     "name": "String",  
                      |     "order": Number,  
                      |     "sys_id": "String"  
                      |   }  
                      | }  
                      | Data type: Array                                                              |
| `secure_params.active` | Flag that indicates whether the secure parameter is active. Valid values:  
                           | • true: The secure parameter is active.  
                           | • false: The secure parameter is inactive. |
### Returns (continued)

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>secure_params.name</td>
<td>Name of the secure parameter. Data type: String</td>
</tr>
<tr>
<td>secure_params.order</td>
<td>Order in which the parameter is sent to the check command/script. Data type: Number</td>
</tr>
<tr>
<td>secure_params.sys_id</td>
<td>Sys_id of the secure parameter listed in the Check Secure Parameter Definitions [sn_agent_check_secure_param_def] table. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the check definition listed in the Check Definitions [sn_agent_check_def] table. Data type: String</td>
</tr>
<tr>
<td>timeout</td>
<td>Timeout in seconds. Data type: Number</td>
</tr>
</tbody>
</table>

The following example shows how to retrieve a list of two check definitions with parameter values.

```javascript
var encQuery = "nameSTARTSWITHchecks_";
var limit = 2;
var withParams = true;

var checkDefs = sn_agent.AccCheckDefsAPI.getChecksList(encQuery, limit, withParams);

gs.info(JSON.stringify(checkDefs, null, 2));
```

**Output:**

```
[
  {
    "name": "checks_api_test",
    "command": "echo hello",
    "plugins": [],
    "timeout": 9,
    "proxy_valid": true,
    "background": false,
  }
]```
"check_type": "TestCheck",
"check_group": "computer",
"sys_id": "7f1f9026dba530106f4810284b96194f",
"params": [],
"secure_params": [

{
  "name": "check_api_test_check_secure_param2",
  "active": true,
  "order": 2,
  "sys_id": "2d30a066dba530106f4810284b9619c1"
},

{
  "name": "check_api_test_check_secure_param1",
  "active": true,
  "order": 100,
  "sys_id": "4c20a066dba530106f4810284b9619a8"
}
]
},

{
  "name": "checks_api_test222",
  "command": "echo hello1212121",
  "plugins": [],
  "timeout": 60,
  "proxy_valid": true,
  "background": false,
  "check_type": "TestCheck",
  "check_group": "computer",
  "sys_id": "99e12466dba530106f4810284b961976",
  "params": [
    {
      "name": "check_api_test_check_param_222",
      "active": true,
      "mandatory": false,
      "default_value": "test_test_test",
      "sys_id": "44026466dba530106f4810284b9619b2"
    }
  ],
  "secure_params": []
}
**AccCheckDefsAPI - updateCheck(String checkDefId, Object updateJson)**

Enables changing one or more field values of a specified check definition.

See also Checks and policies.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>checkDefId</td>
<td>String</td>
<td>Sys_id of the check definition listed in the Check Definitions [sn_agent_check_def] table.</td>
</tr>
</tbody>
</table>
| updateJson    | Object  | Map of check definition fields to be updated to their new values. Refer to the data dictionary for a comprehensive list of Check Definition fields and types.  

```json
{
    "background": Boolean,
    "check_group": "String",
    "check_type": "String",
    "command": "String",
    "name": "String",
    "plugins": [Array],
    "proxy_valid": Boolean,
    "timeout": Number
}
```

| updateJson.active | Number | Indicates whether this check definition is active. Valid values:  
|-------------------|--------|--------------------------------------------------------------------------|
|                   |        | • 0: This check definition is inactive.  
|                   |        | • 1: This check definition is active.  

| updateJson.background | Boolean | Flag that indicates whether this check definition is a background check. A background check is a check which the agent starts execution of and doesn’t wait for it to finish running. Valid values:  
|-----------------------|---------|--------------------------------------------------------------------------|
|                       |         | • true: This check definition is a background check.  
|                       |         | • false: This check definition is not a background check.  

| updateJson.check_group | String | Group specified for this check definition.  
|------------------------|--------|--------------------------------------------------------------------------|
| updateJson.check_type   | String | Type of check.  

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>updateJson.command</td>
<td>String</td>
<td>Command that the Agent Client Collector executes.</td>
</tr>
<tr>
<td>updateJson.name</td>
<td>String</td>
<td>Name of the check.</td>
</tr>
<tr>
<td>updateJson.plugins</td>
<td>Array</td>
<td>List of Agent Client Collector plugins associated with this check.</td>
</tr>
<tr>
<td>updateJson.proxy_valid</td>
<td>Boolean</td>
<td>Flag that indicates whether the check definition policy is set to work as a proxy. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: This check definition policy is set to work as a proxy.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: This check definition policy is not set to work as a proxy.</td>
</tr>
<tr>
<td>updateJson.timeout</td>
<td>Number</td>
<td>Timeout in seconds.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Error message if unsuccessful.</td>
</tr>
</tbody>
</table>

The following example shows how to deactivate a check definition.

```javascript
var checkJson = sn_agent.AccCheckDefsAPI.getCheck(activeCheckDefs[0].sys_id, true);
if (!gs.nil(checkJson.error))
  gs.error(checkJson.error);

var updateJson = {active: "0"); // deactivate the check definition
sn_agent.AccCheckDefsAPI.updateCheck(checkJson.check.sys_id, updateJson);
```
AccCheckDefsAPI - updateCheckParam(String checkDefParamId, Object updateJson)
Enables changing one or more field values of a specified check parameter.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>checkDefParamId</td>
<td>String</td>
<td>Sys_id of the check parameter listed in the Check Parameter Definitions [sn_agent_check_param_def] table.</td>
</tr>
<tr>
<td>updateJson</td>
<td>Object</td>
<td>Map of check parameter fields to be updated to their new values. Refer to the data dictionary for a comprehensive list of Check Secure Parameter Definition fields and types.</td>
</tr>
<tr>
<td>updateJson.active</td>
<td>Boolean</td>
<td>Flag that indicates whether the check parameter is active. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: The check parameter is active.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: The check parameter is inactive.</td>
</tr>
<tr>
<td>updateJson.default_value</td>
<td>String</td>
<td>Specifies the default value for this check parameter.</td>
</tr>
<tr>
<td>updateJson.mandatory</td>
<td>Boolean</td>
<td>Flag that indicates whether the check parameter is required. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: The check parameter is required.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: The check parameter is optional.</td>
</tr>
<tr>
<td>updateJson.name</td>
<td>String</td>
<td>Name of the check parameter.</td>
</tr>
</tbody>
</table>
The following example shows how to activate a check parameter.

```javascript
var checkParmSysId = "cd922ce6dba530106f4810284b961966";
var updateJson = {"active": "true"};
sn_agent.AccCheckDefsAPI.updateCheckSecureParam(checkParmSysId, updateJson);
```

**AccCheckDefsAPI - updateCheckSecureParam(String checkDefSecureParamId, Object updateJson)**

Enables changing one or more field values of a specified check secure parameter.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>checkDefSecureParamId</td>
<td>String</td>
<td>Sys_id of the secure parameter listed in the Check Secure Parameter Definitions [sn_agent_check_secure_param_def] table.</td>
</tr>
<tr>
<td>updateJson</td>
<td>Object</td>
<td>Map of check secure parameter fields to be updated to their new values. Refer to the data dictionary for a comprehensive list of Check Secure Parameter Definition fields and types.</td>
</tr>
<tr>
<td>updateJson.active</td>
<td>Boolean</td>
<td>Flag that indicates whether the secure parameter is active. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- true: The secure parameter is active.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- false: The secure parameter is inactive.</td>
</tr>
<tr>
<td>updateJson.name</td>
<td>String</td>
<td>Name of the secure parameter.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Error message if unsuccessful.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>updateJson.order</td>
<td>Number</td>
<td>Order in which the parameter is sent to the check command/script.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Error message if unsuccessful.</td>
</tr>
</tbody>
</table>

The following example shows how to activate a check secure parameter.

```javascript
var checkSecParmSysId = "<sys_id>";
var updateJson = {"active": "true"};
sn_agent.AccCheckDefsAPI.updateCheckSecureParam(checkSecParmSysId, updateJson);
```

### ActivityHistoryRecord - Global

A data object that is a member of the WorkflowModelManager class; ActivityHistoryRecord methods facilitate interacting with the WorkflowModelManager object.

The ActivityHistoryRecord is basically a copy of the data from a GlideRecord on table wf_history. There are some helper methods defined for this class as well. There are no accessors or mutator to the simple data elements of this class.

**ActivityHistoryRecord - addArrivedActivityToJoin(String Array activityIds)**

Changes the `arrivedState` of already known ids to `true`.

As soon as this method changes the `arrivedState` of known ids to `true`, it tests to see if the join is satisfied. If it is, the model walk can continue; if not, it should continue to next appropriate transition. This is called during playback, as expected `wf_activities` flow to the join.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>activityIds</td>
<td>String Array</td>
<td>Array of <code>wf_activity.sys_ids</code> that are headed towards join.</td>
</tr>
</tbody>
</table>
ActivityHistoryRecord - addJoinFromActivityIds(String Array activityIds)

Seeds all the expected wf_activity sys_ids that are expected to pass through this join and sets their arriveState = false.

This state is flipped to **true** as each expected activity transitions to this instance of the Join. This method is called as part of building the cached model in memory, and is not called during playback.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>activityIds</td>
<td>String Array</td>
<td>Array of wf_activity.sys_ids that are headed towards join.</td>
</tr>
</tbody>
</table>

ActivityHistoryRecord - addTransition(Object InTransition)

As model is cached by `_getExecutedTransitions()`, this method adds transitions that have gone FROM this activity towards the .to activity.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InTransition</td>
<td>Object</td>
<td>ExecutedTransition Javascript object to add to this activity.</td>
</tr>
</tbody>
</table>
ActivityHistoryRecord - debugDump( )
Prints debug information.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ActivityHistoryRecord - doesJoinContainActivity( )
Tests the incoming wf_activity sys_id, presumed to be seeded in this history record. If it is, it is tested to see if it is an activity that would come through this join.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ActivityHistoryRecord - getLogString( )
Returns the log string from the history record.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The log string.</td>
</tr>
</tbody>
</table>

**ActivityHistoryRecord - getSatisfiedJoinActivities()**

Returns the sys_ids of the history records that transition to this Join activity that have already come through.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Array of sys_ids of the history records that transition to this Join activity.</td>
</tr>
</tbody>
</table>

**ActivityHistoryRecord - getTransitionCount()**

Returns the transition count for this activity.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>Number</td>
<td>Number of transitions from this activity.</td>
<td></td>
</tr>
</tbody>
</table>

**ActivityHistoryRecord - getUnSatisfiedJoinActivities( )**

Returns the sys_ids of the history records that transition to this Join activity that the join is still waiting for.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

**ActivityHistoryRecord - isARollback( )**

Determines if this activity is one of the Rollback activities.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Boolean</td>
</tr>
</tbody>
</table>

**ActivityHistoryRecord - isIdADestination(String ahrSys_id)**

Determines if the sys_id passed in is a destination of any of the transitions associated with this instance of an ActivityHistoryRecord.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ahrSys_id</td>
<td>String</td>
<td>Activity history sys_id from table wf_history.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True, if the sys_id passed in is a destination of any of the transitions.</td>
</tr>
</tbody>
</table>

### ActivityHistoryRecord - isJoin( )

Determines if this activity is a join.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True, if the activity is &quot;Join&quot;.</td>
</tr>
</tbody>
</table>

### ActivityHistoryRecord - isJoinSatisfied( )

Examines the value of the arriveState.

Called during playback when the transition to sys_id of a transition points to an activity that is a join.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True, if the join is satisfied.</td>
</tr>
</tbody>
</table>

ActivityHistoryRecord - isJoinWaitingForActivity(String Array activityIds)
Tests the incoming wf_activity sys_id, presumed to be seeded in this history record.

If it is, it is tested to see if it has already been through; if it has, then this join is not waiting on this activity and it should look further downstream for the one that is waiting. This test is called in sequence as the model is walked, so it does presume in the instances of rollbacks and loops. If the sys_id is in here and the flag is true, this waiting Join is further up the sequence. This sort of check is required as the history records of Joins are removed from the history table, and so it is possible for a record to be in the transition table with no match in the corresponding history table, but is still an appropriate match for an earlier executed instance of the same Join.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>activityIds</td>
<td>String Array</td>
<td>Array of wf_activity.sys_ids that are headed towards join.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True, if the activity has already been through.</td>
</tr>
</tbody>
</table>

ActivityHistoryRecord - isRolledBack( )
Determines if this workflow was rolled back.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True, if this workflow was rolled back.</td>
</tr>
</tbody>
</table>

**ActivityHistoryRecord** - `isTurnstile()`

Determines if this activity is a turnstile.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True, if this activity is a turnstile.</td>
</tr>
</tbody>
</table>

**action** - Scoped, Global

Enables handling data for URLs in a UI action script.

Use the `action` API to configure UI actions with which users can interact. Use these scripts in the UI Action [sys_ui_action] table. For information, see UI actions. Methods for this API are referred to by the variable name 'action' in any server-side JavaScript. To learn more, see Creating interactions with UI actions.

**action** - `getGlideURI()`

Gets a GlideURI object to determine the user view.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>GlideURI object representing the URI parameter of the user view.</td>
</tr>
</tbody>
</table>
The following example shows how to get the user view and set the redirect URL to the service catalog home page.

```javascript
var uri = action.getGlideURI();
var path = 'catalog_home.do';

uri.set('sysparm_view', 'catalog_default');
action.setRedirectURL(uri.toString(path));
```

### action - getReturnURL()

Gets the URL of the return page in view after a UI action is complete.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>URL of the return page in view after a UI action is complete.</td>
</tr>
</tbody>
</table>

```javascript
action.getReturnURL();
```

### action - getURLParameter(String parameterName)

Gets the value of a URL parameter.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameterName</td>
<td>String</td>
<td>Name of the URL parameter name to be queried for the URL parameter value.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>URL parameter value.</td>
</tr>
</tbody>
</table>

```javascript
action.getURLParameter('sysparm_query');
```

**action - openGlideRecord(Object gr)**

Opens a page with a GlideRecord in the user view.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gr</td>
<td>GlideRecord</td>
<td>GlideRecord of the page to be opened in the user view.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to include the caller name and incident description created from an interaction record. For more details, see Set up custom UI actions in Workspace.

```javascript
if(current.update()){
    var inc = new GlideRecord("incident");
    inc.newRecord();
    inc.caller_id = current.opened_for;
    inc.short_description = current.short_description;
    action.openGlideRecord(inc);
}
```

**action - setNoPop(Boolean noPop)**

Indicates whether to enable or disable pop-up windows on the page in the current view.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>noPop</td>
<td>Boolean</td>
<td>Flag indicating whether to enable or disable pop-up windows on the page:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Disables pop-up windows.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Default. Enables pop-up windows.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to disable pop-up windows for a UI action.

```java
action.setNoPop(true);
```

**action - setRedirectURL(Object URL)**

Sets the redirect URI for this transaction, which determines the next page the user sees.

For use cases:

- Create a UI routing action
- Create knowledge articles from HR cases
- Define task relationships with UI actions

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>Object</td>
<td>URL to set as the redirect. You can provide the URL as a string or a GlideRecord. If you pass the URL as a GlideRecord, this value takes the focus to that record's form.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
The following example shows how to redirect a user to a URL from a UI action using the `current` variable.

```javascript
var fixchg = new GlideRecord("change_request");
fixchg.short_description= current.short_description;
fixchg.comments= current.comments.getHTMLValue();
// fixchg.parent = current.sys_id;
fixchg.insert();
FixChange();

gs.addInfoMessage("Change "+ fixchg.number+" created");
action.setRedirectURL(current);
action.setReturnURL(fixchg);

function FixChange(){
  var m2m = new GlideRecord('task_rel_task');
m2m.initialize();
m2m.child= current.sys_id;
m2m.parent= fixchg.sys_id;
m2m.type.setDisplayValue("Fixes::Fixed by");
m2m.insert();}
```

The following example shows how to create a new incident record and redirect to the new incident after a UI action completes.

```javascript
var reqItem = current.u_item;
var requestedFor = current.u_requested_for;
var location = current.location;

if(current.u_incident_request == 'Incident'){  
  //Create a new incident record and redirect to the new incident
  var rec = new GlideRecord('incident');
  rec.initialize();
  rec.caller_id = requestedFor;
  rec.location = location;
  rec.insert();
  action.setRedirectURL(rec);
}

if(current.u_incident_request == 'Request'){  
  //Build the url and route the user to the request item
  var url = '';
  if(current.u_item.sys_class_name == 'sc_cat_item_guide'){
```
url = 'com.glideapp.servicecatalog_cat_item_guide_view.do?sysparm_initial=true&sysparm_guide=' + reqItem + '&sysparm_user=' + requestedFor + '&sysparm_location=' + location;
}
else{
    url = 'com.glideapp.servicecatalog_cat_item_view.do?sysparm_id=' + reqItem + '&sysparm_user=' + requestedFor + '&sysparm_location=' + location;
}
action.setRedirectURL(url);

**action - setReturnURL(Object URL)**

Sets the return URI for this transaction after a UI action is complete. You can use this method to determine what page the user has in view when they return from submit.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>Object</td>
<td>URI to set as the return location after a UI action is complete. You can provide the URL as a string or a GlideRecord.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example enables the user to select the UI action to create a change record using information from the problem record and the change template. After the change, the user returns to current view. For more information, see [Scripted templates](#). For more use cases, see [Define task relationships with UI actions](#).

```javascript
var change = new GlideRecord("change_request");
change.initialize();
change.short_description = current.short_description;
change.description = current.u_details;
change.cmdb_ci = current.u_service;
change.priority = current.priority;
change.requested_by = current.caller_id;
```
change.assignment_group.setDisplayValue('Change & Release');
change.u_status = 'New';
change.parent = current.number;
change.applyTemplate("standard/rfc");
current.rfc = change.insert();
current.comments = 'Change ' + change.number + ' created';;

var mySysID = current.update();

gs.addInfoMessage("Change " + change.number +" created");

action.setRedirectURL(change);
action.setReturnURL(current);

**action - setURLParameter(String parameterName, String parameterValue)**
Sets a URL parameter name and value.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameterName</td>
<td>String</td>
<td>Name of the URL parameter.</td>
</tr>
<tr>
<td>parameterValue</td>
<td>String</td>
<td>Value of the parameter.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

action.setURLParameter('sysparm_query', 'priority=2^active=true');

**Agent - Global**

Provides methods for getting or setting agent presence and channel availability.
The **Agent** API is provided within the `sn_awa` namespace.

Requires the following:

- Advanced Work Assignment plugin (com.glide.awa)
- Role: awa_integration_user or admin

**Agent - get(String user_sys_id)**

Gets an agent by sys_id.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user_sys_id</td>
<td>String</td>
<td>The sys_id of the user listed in sys_user table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the user exists, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var agent = sn_awa.Agent.get("<user_sys_id>");
var presence = agent.getPresence();
```

Agent – getPresence()

Gets the current presence state and channel availability of a provided agent.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Presence state sys_id, presence state name, and channel information. Error message otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var agent = sn_awa.Agent.get("<user_sys_id>");
var presence = agent.getPresence();
```

Agent – setPresence(Object setRequest)

Sets the state of a provided agent’s presence and sets the agent’s channel availability for that state.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>setRequest</td>
<td>Object</td>
<td>Contains setRequest.sys_id and setRequest.channels.</td>
</tr>
<tr>
<td>setRequest.sys_id</td>
<td>String</td>
<td>Represents presence_sys_id from awa_presence_state table</td>
</tr>
</tbody>
</table>
| setRequest.channels   | name/value pair | Optional  
  - sys_id – channel_sys_id from the awa_service_channel table  
  - available – Agent availability to receive work for this channel (true or false) |

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if successful, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var agent = sn_awa.Agent.get("<user_sys_id>");
var presence = agent.setPresence({  
   sys_id: <presence_sys_id>,  
   channels: [][]  
   sys_id: "<channel_sys_id>",  
   available: true // or false  
 });
```

### AgentNowHandler API - Scoped

The AccCheckDefsAPI script include enables running check definitions on demand, checking the status of on demand executions and test check executions and for stopping background checks that were executed on demand.

This API requires the Agent Client Collector Framework (sn_agent) store application and is provided within the `sn_agent` namespace. For more information, refer to Agent Client Collector.

For the REST API solution, refer to Agent Client Collector API.

#### AgentNowHandler - AgentNowHandler()

Creates an AgentNowHandler instance.
The following example shows how to initialize `AgentNowHandler`.

```javascript
var handler = new sn_agent.AgentNowHandler();
```

**AgentNowHandler - `getRequestStatus(String requestId)`**

Gets status of the request with the given ID.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestId</td>
<td>String</td>
<td>A check request ID generated by calling the <code>runCheckForCis()</code> method.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Status of the request and any applicable error message.</td>
</tr>
<tr>
<td>status</td>
<td>Request status. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• done – Check is successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – Check has failed. See error message for details.</td>
</tr>
<tr>
<td></td>
<td>• mid_flow – Request output is being handled by the MID server.</td>
</tr>
<tr>
<td></td>
<td>• processing – Check is in progress.</td>
</tr>
<tr>
<td></td>
<td>• timeout – Check processing exceeded time limit set in the <code>runCheckForCis()</code> method.</td>
</tr>
<tr>
<td>err_msg</td>
<td>Error message if any. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• No agents found for relevant CIs.</td>
</tr>
<tr>
<td></td>
<td>• No background check request with given ID.</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• No request with given ID.</td>
<td></td>
</tr>
<tr>
<td>• No test result with given ID.</td>
<td></td>
</tr>
<tr>
<td>• Request timeout.</td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to get the status of a request.

```javascript
var handler = new sn_agent.AgentNowHandler();
var check = {checkDefId: "158279505372b30034b8ddeeff7b1270"};
var computerGr = new GlideRecord("cmdb_ci_computer");
computerGr.query();
var requestId = handler.runCheckForCis(computerGr, check, 0, 60);
var reqStatusJson = handler.getRequestStatus(requestId);
gs.info(JSON.stringify(reqStatusJson));
```

### AgentNowHandler - getTestResultStatus(String testResultId)

Gets the test check status of the given test result.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>testResultId</td>
<td>String</td>
<td>A test result ID generated by creating a test check request.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>Status of the test results. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Pending</td>
</tr>
<tr>
<td></td>
<td>• 1: In progress</td>
</tr>
<tr>
<td></td>
<td>• 2: Complete</td>
</tr>
<tr>
<td></td>
<td>• 3: No test result with given ID</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>output</td>
<td>Output describing the status.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

The following example shows how to get result status of a completed test check request:

```javascript
var testCheckStatusJson = handler.getTestResultStatus("testResultId");
gs.info(JSON.stringify(testCheckStatusJson));
```

**AgentNowHandler - runCheckForCis(Object cis, Object check, Number priority, Number timeout)**

Runs a check against the given configuration item.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cis</td>
<td>GlideRecord</td>
<td>GlideRecord of any CMDB table (any application, host, or agent) that the check is working against.</td>
</tr>
<tr>
<td>check</td>
<td>Object</td>
<td>Contains the check ID and optional check parameters.</td>
</tr>
<tr>
<td>check.checkDefId</td>
<td>String</td>
<td>Sys_id of a check definition in the Check Definitions [sn_agent_check_def] table.</td>
</tr>
<tr>
<td>check.params</td>
<td>Object</td>
<td>Optional. Map of parameter names and values. These settings can be used to override the parameter records of the check definition and its specified values.</td>
</tr>
<tr>
<td>priority</td>
<td>Number</td>
<td>Priority of the request to be set on the ECC queue.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0: interactive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1: expedited</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2: standard</td>
</tr>
<tr>
<td>timeout</td>
<td>Number</td>
<td>Value of the timeout for the request in seconds.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the generated background check request.</td>
</tr>
</tbody>
</table>

The following example shows how to run a background check and get its request ID.

```javascript
var handler = new sn_agent.AgentNowHandler();
var check = {checkDefId: "028fcd5067c80010b7b72dbd2685ef4f");
var computerGr = new GlideRecord("cmdb_ci_computer");
computerGr.query();
var requestId = handler.runCheckForCis(computerGr, check, 0, 60);
gs.info(requestId);
```

Output:

```
b9cf14aedb5e30106f4810284b961990
```

AgentNowHandler - stopBackgroundCheck(String requestId)

Stops a background check.

To start a background check, use the runCheckForCis() method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestId</td>
<td>String</td>
<td>The ID of a background check request generated by calling the runCheckForCis() method.</td>
</tr>
</tbody>
</table>
The following example shows how to stop executing a background check.

```java
handler.stopBackgroundCheck(backRequestId);
```

## AgentMetrics - Global

Processes inbound MID server metrics.

Updates these tables:
- ECC Agent Counter Metric
- ECC Agent Memory Metric
- ECC Agent RGR Metric
- ECC Agent Scalar Metric

Use these methods in server scripts.

### AgentMetrics - AgentMetrics()

Creates an instance of AgentMetrics.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### AgentMetrics - handleCounter(String metric, String name, String midId)

Handles counter metrics from the agent by putting them into an array that can be used to update the ECC Agent Counter Metric table (ecc_agent_counter_metric).

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metric</td>
<td>String</td>
<td>The counter object from the agent</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The device name</td>
</tr>
<tr>
<td>midId</td>
<td>String</td>
<td>The MID server sys_id</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var amtc = new AgentMetrics();
amtc.handleCounter('count', 'server_name', '1098888bbb9483abcd89981ffaeff');
```

#### AgentMetrics - handleMemory(String metric, String name, String midId)

Updates the memory metrics for the given agent.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metric</td>
<td>String</td>
<td>The memory object from the agent</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The device name</td>
</tr>
<tr>
<td>midId</td>
<td>String</td>
<td>The MID server sys_id</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var amtc = new AgentMetrics();
amtc.handleMemory('mem_obj', 'server_name', '1098888bbb9483abcd89981ffaeff');
```

#### AgentMetrics - handleRGR(String metric, String name, String midId)

Handles metric information from a remote glide record by putting it into an array.

```javascript
var amtc = new AgentMetrics();
amtc.handleRGR('mem_obj', 'server_name', '1098888bbb9483abcd89981ffaeff');
```
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metric</td>
<td>String</td>
<td>The remote object from the agent</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The device name</td>
</tr>
<tr>
<td>midId</td>
<td>String</td>
<td>The MID server sys_id</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var amtc = new AgentMetrics();
amtc.handleRGR('remote_obj', 'server_name', '109888bbb9483abcd89981ffaeff');
```

**AgentMetrics - handleScalar(String metric, String name, String midId)**

Handles scalar values from the agent by putting them into an array.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metric</td>
<td>String</td>
<td>The counter object from the agent</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The device name</td>
</tr>
<tr>
<td>midId</td>
<td>String</td>
<td>The MID server sys_id</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var amtc = new AgentMetrics();
amtc.handleScalar('scalar', 'server_name', '109888bbb9483abcd89981ffaeff');
```
AgentMetrics - updateMetric(String tableName, String values, String metric, String midID)

Updates the specified metrics table with the new metric information.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>The metrics table to update</td>
</tr>
<tr>
<td>values</td>
<td>String</td>
<td>The object containing the values to add to the table</td>
</tr>
<tr>
<td>metric</td>
<td>String</td>
<td>The metric type to add to the table</td>
</tr>
<tr>
<td>midID</td>
<td>String</td>
<td>The MID server sys_id</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var amtc = new AgentMetrics();
amtc.handleMemory('memory', 'server_name', '1098888bbb9483abcd89981ffaeff');
amtc.updateMetric('ecc_agent_memory_metric', 'm_values', 'memory', '1098888bbb9483abcd89981ffaeff');
```

APCHandler - Global

Handles SNMP classification and identification for UPSs, environmental monitors, and PDUs made by APC.

Use this API for SNMP classification and identification.

APCHandler - classifyAndIdentify()

Classifies and identifies APC devices.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ApcPduOutletReconciler - Global
Reconciles outlets for power distribution units.
Use this API for SNMP-related discovery.

ApcPduOutletReconciler - getReconciliationField()
Returns the reconciliation field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The reconciliation field</td>
</tr>
</tbody>
</table>

ApcPduOutletReconciler - getReconciliationKey()
Returns the reconciliation key.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The reconciliation key</td>
</tr>
</tbody>
</table>
ApcPduOutletReconciler - hasChanged()
Determines if the outlet state has changed.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the outlet state has changed; otherwise, false.</td>
</tr>
</tbody>
</table>

ApcPduOutletReconciler - readDatabaseFields()
Reads the outlet database fields.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

ApcPduOutletReconciler - readDiscovered()
Reads the discovered outlet database fields.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**ApcPduOutletReconciler - setDatabaseFields()**

Sets the outlet database fields.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**api - UI Builder**

The `api` API provides methods that you can use when developing client scripts in the UI Builder.

This API is exposed to client scripts, also known as page scripts. Client scripts are executed in response to something happening on a page, such as:

- User interaction events/actions, such as a button click.
- Lifecycle events, such as a data broker execution started.

These scripts do not have to return anything to the framework and can be written as an asynchronous function.

This API is also exposed to scripted property values. These scripts are executed whenever the framework-runtime needs to calculate a value, such as:

- Passing to a component property.
- Determining component visibility.
- Emitting an event with a payload.

These scripts cannot be written as an asynchronous function. They also cannot invoke side-effect methods on the api object, such as, `api.emit()`, `api.setState()`, and `api.data.<data_resource_id>.*()`.
The api object contains both configuration dependent and configuration independent properties that you can access within the context of the associated page or component. You cannot directly modify the properties within this object. Modification can only be made through the available methods.

**api - api.context.props.<page_property_name>**

Page properties can be configured within UI Builder. The configuration values depend on the context in which the page is used.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;page_property_name&gt;</td>
<td>Any</td>
<td>Available values are dependant on the client script implementation. To access these properties, use the following: api.context.props.&lt;page_property_name&gt;. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>// A record page with property table could be accessed with function isActivityStreamVisible({api}) { return api.context.props.table === 'incident'; }</td>
</tr>
</tbody>
</table>

**Note:** These property values are read-only. Mutating nested object values from scripts is not supported.

**api - api.context.session.<session_property>**

Context session properties associated with the current user.

**Available session properties**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| isLoggedIn | Boolean | Flag that indicates whether the current user is logged in to the system. Possible values:
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>properties.awaEnabled</td>
<td>String</td>
<td>The system property glide.awa.enabled that indicates whether the auto assignment for work items for Advanced Work Assignment (AWA) is enabled for the current user. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: AWA is enabled for the user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: AWA is not enabled for the user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For additional information, see Components installed with Advanced Work Assignment.</td>
</tr>
<tr>
<td>properties.forgetMe.value</td>
<td>String</td>
<td>The property glide.ui.forgetme that indicates whether to remove the Remember Me check box from the login page to prevent login information from being cached. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Remove the Remember Me check box.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Display the Remember Me check box.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For additional information, see Remove remember me.</td>
</tr>
<tr>
<td>properties.sessionTimeLeft.value</td>
<td>String, Number coerced to string</td>
<td>The system property glide.ui.session_timeleft that determines the amount of time left before the current session times</td>
</tr>
</tbody>
</table>
### Available session properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>properties.sessionTimeout.value</td>
<td>String, Number coerced to string</td>
<td>The system property glide.ui.session_timeout that determines the initial session timeout value. Unit: Minutes. Values greater than 1440 minutes are treated as one day. For additional information, see <a href="#">Session activity timeout</a>.</td>
</tr>
<tr>
<td>user.avatar</td>
<td>String</td>
<td>URL of the current user's avatar.</td>
</tr>
<tr>
<td>user.dateFormat</td>
<td>String</td>
<td>Default date format.</td>
</tr>
<tr>
<td>user.domain</td>
<td>String</td>
<td>Domain path for the current user.</td>
</tr>
<tr>
<td>user.firstName</td>
<td>String</td>
<td>First name of the current user.</td>
</tr>
<tr>
<td>user.fullName</td>
<td>String</td>
<td>First and last name of the current user.</td>
</tr>
<tr>
<td>user.initials</td>
<td>String</td>
<td>Initials of the current user.</td>
</tr>
<tr>
<td>user.language</td>
<td>String</td>
<td>Primary language spoken by the current user.</td>
</tr>
<tr>
<td>user.preferences</td>
<td>Array of objects</td>
<td>Name-value pairs that describe the user preferences. These</td>
</tr>
</tbody>
</table>
Available session properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>user preferences are stored as records in the User Preference [sys_user_preference] table, and are updated each time the user changes their settings. For additional information, see User preferences.</td>
</tr>
<tr>
<td>user.roles</td>
<td>Array</td>
<td>Comma-separated list of roles assigned to the current user.</td>
</tr>
<tr>
<td>user.sys_id</td>
<td>String</td>
<td>Sys_id of the user in the User [sys_user] table.</td>
</tr>
<tr>
<td>user.timeFormat</td>
<td>String</td>
<td>Default time format to use for the user.</td>
</tr>
<tr>
<td>user.timeZone</td>
<td>String</td>
<td>Time zone of the current user.</td>
</tr>
<tr>
<td>user.timeZoneOffset</td>
<td>String</td>
<td>Time zone offset of the current user.</td>
</tr>
</tbody>
</table>

api - api.data.<data_resource_id>.addErrorMessage(Object payload)

Displays the specified error message at the top of the current form.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the error message to display.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;payload&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>payload.message</td>
<td>String</td>
<td>Error message to display.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```
api.data.gform.addErrorMessage({message: 'Error message'});
```

**`api - api.data.<data_resource_id>.addInfoMessage(Object payload)`**

Displays the specified informational message at the top of the current form.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the informational message to display.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;payload&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>payload.message</td>
<td>String</td>
<td>Informational message to display.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```
api.data.gform.addInfoMessage({message: 'Test message'});
```
api - api.data.<data_resource_id>.addOption(Object payload)

Adds an option to the specified choice type field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the field value to update.</td>
</tr>
<tr>
<td>payload.choiceIndex</td>
<td>String</td>
<td>Optional. Index into the choice list at which to insert the option. Default: End of the choice list.</td>
</tr>
<tr>
<td>payload.choiceLabel</td>
<td>String</td>
<td>Label of the option to add to the specified field.</td>
</tr>
<tr>
<td>payload.choiceValue</td>
<td>String</td>
<td>Value of the option to add to the specified field.</td>
</tr>
<tr>
<td>payload.fieldName</td>
<td>String</td>
<td>Name of the choice type form field to add the specified option to.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

/api.data.gform.addOption({fieldName: 'priority', choiceLabel: 'Extremely High', choiceValue: '10'});

api - api.data.<data_resource_id>.addWarningMessage(Object payload)

Displays the specified warning message at the top of the current form.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the warning message to display.</td>
</tr>
<tr>
<td>payload.message</td>
<td>String</td>
<td>Warning message to display.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.addWarningMessage({message: 'Test message'});
```

**api - api.data.<data_resource_id>.clearMessage()**

Removes all informational and error messages from the top of the current form.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
</tbody>
</table>
### api.data.<data_resource_id>.clearOptions(Object payload)

Clears all options from the specified choice type field.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the choice type field whose options are to be cleared.</td>
</tr>
</tbody>
</table>

```json
"payload": {
  "fieldName": "String"
}
```

| payload.fieldName | String  | Name of the choice type field whose options are to be cleared. |

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.clearOptions({fieldName: 'priority'});
```

### api.data.<data_resource_id>.executeUiAction(Object payload)

Executes the specified UI action.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the UI action to execute.</td>
</tr>
<tr>
<td>payload.actionSysId</td>
<td>String</td>
<td>Sys_id of the UI action to execute. Located in the UI Action [sys_ui_action] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.executeUiAction({actionSysId: '60615ff90f730010ac7de6f8c4767e9a'});
```

**api** - **api.data.<data_resource_id>.execute(Object inputValues)**

Triggers an execute operation on the specified data resource.

This method is only available if the data resource is one of the following types:
- Composite
- GraphQL
- REST
- Scriptlet
- Transform

ℹ️ Note: This method is only exposed if the `mutates_server_data` field is set to `true` on the corresponding data resource (sys_ux_data_broker_* table) record. It is accessible under `api.data.<data_resource_Id>.refresh()`.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resourceId</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>inputValues</td>
<td>Object</td>
<td>Object to pass to the specified data resource. This object must conform to the data resource’s input parameters.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This code example shows a page script that is invoked when the **Submit** button on the page is clicked. The page is configured with a Server Data Resource that creates a new record.

```javascript
function handler({api}) {
  if (api.state.movieYear === 2020) {
    // The data resource used in this case specifies two input parameters: name and year
    api.data.create_movie_record.execute({
      name: api.state.movieName,
      year: api.state.movieYear
    });
  }
}
```

**api - api.data.<data_resource_id>.hideFieldMessage(Object payload)**

Hides the oldest message next to the specified field or clears all messages associated with the field.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resourceId</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.)</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the field message to hide.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;payload&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;clearAll&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;fieldName&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
| payload.clearAll | Boolean | Optional. Flag that indicates whether to clear all messages associated with the specified form field. Valid values:
|               |         | • true: Clear all messages associated with the specified field.                                |
|               |         | • false: Do not clear all messages associated with the specified field.                        |
|               |         | Default: false                                                                                |
| payload.fieldName | String  | Name of the form field for which to hide the oldest message or clear all associated messages. |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

api.data.gform.hideFieldMessage({fieldName: 'short_description'});

api - api.data.<data_resource_id>.hideRelatedList(Object payload)

Hides the specified related list on the current form.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the related list to hide.</td>
</tr>
<tr>
<td>&quot;payload&quot;: {</td>
<td></td>
<td>&quot;listTableName&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>payload.listTableName</td>
<td>String</td>
<td>Name of the related list to hide. Located in the Related List [sys_ui_related_list] table. If the list to hide is through a relationship, provide the sys_id of the list instead of the name.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.hideRelatedList({listTableName:'incident.parent_incident'});
```

**api - api.data.<data_resource_id>.hideRelatedLists()**

Hides all related lists on the current form.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.hideRelatedLists();
```

**api - api.data.<data_resource_id>.lifecycle.lastFetchSucceeded**

Boolean flag that indicates whether the last fetch attempt for the specified data resource instance finished successfully.

If the value is true, the last fetch attempt for the data resource instance finished successfully; otherwise, false.

**Field**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this</td>
</tr>
<tr>
<td></td>
<td></td>
<td>method must be based off of Glide Form (gform.) The available data resource</td>
</tr>
<tr>
<td></td>
<td></td>
<td>instances are configuration-dependent and defined when you add the data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>resource to your page in UI Builder.</td>
</tr>
</tbody>
</table>

**api - api.data.<data_resource_id>.refresh()**

Triggers a refresh operation for the specified non-mutating data resource instance.

Call this method if the underlying data being fetched by the data resource changes. A data resource is considered non-mutating if the `mutates_server_data` field on the record is set to false.

This method is asynchronous and emits an internal event to trigger the refresh of the specified data resource instance. The UI Builder allows you to trigger client scripts in response to data resource lifecycle events, such as `DATA_FETCH_SUCCEEDED` and `DATA_FETCH_FAILED`. For additional information on these events, see Event mapping.

This method is only available if the data resource is one of the following types:

- Composite
- GraphQL
• REST
• Scriptlet
• Transform

Note: This method is only exposed if the `mutates_server_data` field is set to `false` on the corresponding data resource (sys_ux_data_broker_* table) record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This code example shows a page script that is invoked when a dropdown item is selected in a page. The page is configured with two Server Data Resources that query problem and incident tables.

```javascript
function handler({api, event}) {
    const value = event.payload.value[0];
    if (value === 'problem')
        api.data.problem_list_1.refresh();
    else if(value === 'incident')
        api.data.incident_list_1.refresh();
}
```

api - `api.data.<data_resource_id>`.reload()

Reloads the current form using the same table and sys_id.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.reload();
```

**api - api.data.<data_resource_id>.removeOption(Object payload)**

Removes an option from the specified choice type field.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the choice type field to update.</td>
</tr>
<tr>
<td>payload.choiceValue</td>
<td>String</td>
<td>Value of the option to remove from the specified choice type field.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>payload.fieldName</code></td>
<td><code>String</code></td>
<td>Name of the choice type form field from which to remove the specified value.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.removeOption({fieldName: 'priority', choiceValue: '1'});
```

#### `api - api.data.<data_resource_id>.save()`

Triggers form submission using the `Save` UI action.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>data_resource_id</code></td>
<td><code>String</code></td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (<code>gform</code>). The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.save();
```

#### `api - api.data.<data_resource_id>.setMandatory(Object payload)`

Sets whether the specified form field is mandatory.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this</td>
</tr>
<tr>
<td></td>
<td></td>
<td>method must be based off of Glide Form (gform.)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The available data resource instances are configuration-dependent and defined</td>
</tr>
<tr>
<td></td>
<td></td>
<td>when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the field whose mandatory information to update.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;payload&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;fieldName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;mandatory&quot;: Boolean</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>payload.field</td>
<td>String</td>
<td>Name of the form field whose mandatory value is to be set.</td>
</tr>
<tr>
<td>payload.mandatory</td>
<td>Boolean</td>
<td>Flag that indicates the specified form field is mandatory, meaning the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>form cannot be saved without this field containing a valid value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Field is mandatory.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Field is optional.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```
api.data.gform.setMandatory({fieldName: 'short_description', mandatory: false});
```

**api - api.data.<data_resource_id>.setReadOnly(Object payload)**

Sets the read/write capabilities of the specified form field.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the field whose readability information to update.</td>
</tr>
<tr>
<td>payload.fieldName</td>
<td>String</td>
<td>Name of the form field whose readability is to be set.</td>
</tr>
<tr>
<td>payload.readonly</td>
<td>Boolean</td>
<td>Flag that indicates the read/write capabilities of the specified form field. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Field is read only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Field is read/write.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.setReadOnly({fieldName: 'short_description', readonly: false});
```

**api -api.data.<data_resource_id>.setvalue(Object payload)**

Updates a specified GlideForm field with the specified value. Optionally, you can also update the display value with the same specified value.

Only the value on the form is updated. The value is not saved in the database.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the field whose value to update.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;payload&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;displayValue&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;fieldName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td>payload.displayValue</td>
<td>String</td>
<td>Optional. Name of the display value to update. If left blank, the display value is not modified.</td>
</tr>
<tr>
<td>payload.fieldName</td>
<td>String</td>
<td>Name of the form field to update.</td>
</tr>
<tr>
<td>payload.value</td>
<td>String</td>
<td>Value to update the field with.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.setValue({fieldName: 'short_description', value: 'short description'});
```

**api - api.data.<data_resource_id>.setVisible(Object payload)**

Sets the visibility of the specified form field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.)</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the field on which to set visibility.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;payload&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;fieldName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;visibility&quot;: Boolean</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

| payload.field | String   | Name of the form field whose visibility is to be set.                                                                 |
| payload.visibility | Boolean | Flag that indicates whether the associated field is visible on the current form. Valid values:                                      |
|                |          | • true: Field will display on the form.                                                                                               |
|                |          | • false: Field will not display on the form.                                                                                          |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.setVisible({'fieldName': 'short_description', 'visibility': false});
```

api - api.data.<data_resource_id>.showFieldMessage(Object payload)

Displays the specified message next to the specified field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.)</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the field message to display.</td>
</tr>
<tr>
<td>payload.fieldName</td>
<td>String</td>
<td>Name of the field next to which the message should appear.</td>
</tr>
<tr>
<td>payload.message</td>
<td>String</td>
<td>Message to display.</td>
</tr>
<tr>
<td>payload.type</td>
<td>String</td>
<td>Optional. Type of message to display. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• error</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• info</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• warning</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: info</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.showFieldMessage({fieldName: 'short_description', message: 'Error message', type: 'error'});
```

`api - api.data.<data_resource_id>.showRelatedList(Object payload)`

Displays the specified related list on the current form.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the related list to display.</td>
</tr>
<tr>
<td>payload.listTableName</td>
<td>String</td>
<td>Name of the related list to display. Located in the Related List [sys_ui_related_list] table. If the list to display is through a relationship, provide the sys_id of the list instead of the name.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.showRelatedList({'listTableName': 'incident.parent_incident'});
```

**api - api.data.<data_resource_id>.showRelatedLists()**

Displays all related lists associated with the current form.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.showRelatedLists();
```

#### api - api.data.<data_resource_id>.submit()
Triggers form submission using the specified UI action.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_resource_id</td>
<td>String</td>
<td>Unique identifier of the associated data resource. The data resource for this method must be based off of Glide Form (gform.) The available data resource instances are configuration-dependent and defined when you add the data resource to your page in UI Builder.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Object that describes the UI action to use to submit the current form.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{&quot;payload&quot;: { &quot;submitActionName&quot;: &quot;String&quot; }}</td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.submit({submitActionName:'sysverb_ws_save'});
```

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
api.data.gform.submit({submitActionName:'sysverb_ws_save'});
```

#### api - api.emit(String eventName, Object payload)
Emits an event with the specified name and payload.
The event name being emitted must be part of the associated page definition’s dispatched events list, which is stored in the UX Macroponent Definition [sys_ux_macroponent] table. Any `api.emit` call that dispatches events not declared in this table are ignored.

For additional information on events, see Work with events.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventName</td>
<td>String</td>
<td>Name of the event to emit. This name should follow the UI Framework action naming guidelines:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Should be upper snake case, such as ITEM_CHANGED. All letters in upper case and all spaces replaced with underscores.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Should be past tense, such as BUTTON_CLICKED or USER_SELECTED.</td>
</tr>
<tr>
<td>payload</td>
<td>Object</td>
<td>Optional. Object that contains the data to send with the emitted event. This object is free-form and can contain whatever data is necessary by the entity receiving the data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> Payloads of primitive type work, but could result in inconsistent behavior.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following code example shows emitting an event called `NOW_UXF_PAGE#ADD_NOTIFICATIONS` with an associated `items` payload.

```javascript
function handler({api}) {
    api.emit('NOW_UXF_PAGE#ADD_NOTIFICATIONS', {
        items: [
            
```
api - setState(String stateParm, Any value)

Sets the value of the specified client state parameter.

Use client state parameters to maintain a shared state on a page. The shared state can then be passed as values to properties of components used on the page. You can also access and update client states in multiple page scripts. A common use case is to keep track of values input by users in multiple form controls on a page. When the form is submitted, a client script could then use all of the values stored in client state parameters to create a new record with a data broker. A page can have one or more client state parameters, which you can declare for a page through the UI Builder. You can then bind a client state parameter to one or more components to share or act on the client state parameter.

api.setState() calls are executed asynchronously and do not necessarily update the UI immediately. If the value to set depends on the currentValue of the client state parameter, or any of the properties provided in the api object, you should use this variant of the api.setState() method to avoid using outdated values.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stateParm</td>
<td>String</td>
<td>Name of the client state parameter to update. This name must be declared in the client state parameters of the associated page. For additional information on declaring client state parameters, see Work with client state parameters.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Any - Must be the same as the client state parameter declaration.</td>
<td>Value to set the specified client state parameter to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows a script that could be executed to update the email client state parameter when an input value is set on a now-input component.

```javascript
function handler({api, event}) {
    api.setState('email', event.payload.value);
}
```

api - `setState(String stateParam, Function callbackFn)`

Sets the value of the specified client state parameter to the value returned by the specified callback function.

The callback function is invoked with an object that has two properties: `currentValue` and `api`. The function should never mutate the `currentValue` property or the `api` object directly.

Use client state parameters to maintain a shared state on a page. The shared state can then be passed as values to properties of components used on the page. You can also access and update client states in multiple page scripts. A common use case is to keep track of values input by users in multiple form controls on a page. When the form is submitted, a client script could then use all of the values stored in client state parameters to create a new record with a data broker. A page can have one or more client state parameters, which you can declare for a page through the UI Builder. You can then bind a client state parameter to one or more components to share or act on the client state parameter.
api.setState() calls are executed asynchronously and do not necessarily update the UI immediately. If the value to set depends on the `currentValue` of the client state parameter, or any of the properties provided in the api object, you should use this variant of the `api.setState()` method to avoid using outdated values.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stateParm</td>
<td>String</td>
<td>Name of the client state parameter to update. This name must be declared in the client state parameters of the associated page. For additional information on declaring client state parameters, see <a href="#">Work with client state parameters</a>.</td>
</tr>
<tr>
<td>callbackFn</td>
<td>Function</td>
<td>Callback function to execute to obtain the value.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to use `api.setState` to log users into a page.

```javascript
function handler({api, event}) {
    const {name, value} = event.payload;
    if (name === 'username' || name === 'password') {
        // Update the loginParameters state object with the username/password value
        api.setState('loginParameters', ({currentValue}) => {
            return {
                ...currentValue,
                [name]: value
            };
        });
    }
}
```

### api - api.state.<client_state_parameter_name>

Current value of the specified client state parameter.
### Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;client_state_parameter_name&gt;</code></td>
<td>Any. The available client state parameters are dependent on the page configuration.</td>
<td>Name of the client state parameter. Available client states are dependant on the client script implementation. To access the available client states, use the following: <code>api.state.&lt;client_state_name&gt;</code>. For example:</td>
</tr>
</tbody>
</table>

```javascript
function showRelatedLists({api}) {
  return !api.state.isCustomListSelected;
}
```

**Note:** These property values are read-only. To update a client state parameter, use `api.setState()`. Mutating nested object values from scripts is not supported.

### ArrayUtil - Global

ArrayUtil API is a script include with useful functions for working with JavaScript arrays.

These methods are available to any server-side script.

### ArrayUtil - concat(Array parent, Array child)

Merge two arrays.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>parent</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>child</td>
<td>Array</td>
<td>An array to merge</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of elements from both input arrays. Duplicates are not removed.</td>
</tr>
</tbody>
</table>

```javascript
var arrayUtil = new ArrayUtil();
var a1 = new Array("a", "b", "c");
var a2 = new Array("c", "d", "e");

gs.print("concat a1, a2: " + arrayUtil.concat(a1, a2));
```

Output: concat a1, a2: a,b,c,c,d,e

**ArrayUtil - contains(Array array, Object element)**

Searches the array for the specified element. Returns true if the element exists in the array, otherwise returns false.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>array</td>
<td>Array</td>
<td>Array to search.</td>
</tr>
<tr>
<td>element</td>
<td>Object</td>
<td>Element to search for.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag indicating whether the element was found in the array. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Element found in array.</td>
</tr>
<tr>
<td></td>
<td>• false: Element not found in array.</td>
</tr>
</tbody>
</table>
var arrayUtil = new ArrayUtil();
var a1 = new Array("a", "b", "c");

gs.print("Contains b: " + arrayUtil.contains(a1, "b");
gs.print("Contains x: " + arrayUtil.contains(a1, "x");

Output:
Contains b: true
Contains x: false

ArrayUtil - convertArray(Object a)
Converts a Java object to an array.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Object</td>
<td>Object to convert.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array created from the object.</td>
</tr>
</tbody>
</table>

This example converts a Java object to an array.

var arrayUtil = new ArrayUtil();
// Returns a JavaObject with the logged in user's groups
var groupObj = gs.getUser().getMyGroups();
gs.print('groupObj: ' + Object.prototype.toString.call(groupObj));

var groupArr = arrayUtil.convertArray(groupObj);
gs.print('groupArr: ' + Object.prototype.toString.call(groupArr));

Output

groupObj: [object JavaObject]
groupArr: [object Array]

ArrayUtil - diff(Array a, Array b)
Finds the differences between two or more arrays.
Any number of arrays can be provided as parameters.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Array</td>
<td>An array</td>
</tr>
<tr>
<td>b</td>
<td>Array</td>
<td>An array</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Returns an array of items from array a that were not found in either array b or c, or other input arrays. Duplicates are removed from the result.</td>
</tr>
</tbody>
</table>

```javascript
var arrayUtil = new ArrayUtil();
var a1 = new Array("a", "b", "c");
var a2 = new Array("c", "d", "e");
gs.print(arrayUtil.diff(a1, a2));
```

Output: a,b

### ArrayUtil - ensureArray(Object object)

Returns an array from the specified object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>Object</td>
<td>Object from which to create an array.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array created from the object.</td>
</tr>
</tbody>
</table>

```javascript
var arrayUtil = new ArrayUtil();
var o1 = {a: "1", b: "2", c: "3");
gs.print('o1 is array: ' + Array.isArray(o1));
gs.print('o1 stringified: ' + JSON.stringify(o1));
```
```javascript
var a1 = arrayUtil.ensureArray(o1);
gs.print('a1 is array: ' + Array.isArray(a1));
gs.print('a1 stringified: ' + JSON.stringify(a1));
```

**Output:**

```
o1 is array: false
o1 stringified: {"a":"1","b":"2","c":"3"}
a1 is array: true
a1 stringified: [{"a":"1","b":"2","c":"3"}]
```

**ArrayUtil - `indexOf(Array array, Object element)`**

Searches the array for the element. Returns the element index if found, -1 otherwise.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>array</td>
<td>Array</td>
<td>The array to search</td>
</tr>
<tr>
<td>element</td>
<td>Object</td>
<td>The element to search for</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The index where the element was found, -1 otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var arrayUtil = new ArrayUtil();
var a1 = new Array("a", "b", "c");

gs.print("indexOf b: ", + arrayUtil.indexOf(a1, "b");
gs.print("indexOf x: ", + arrayUtil.indexOf(a1, "x"));
```

**Output:**

```
indexOf b: 1
indexOf x: -1
```

**ArrayUtil - `indexOf(Array array, Object element, Number startIndex)`**

Searches the array for the specified element starting at the specified index. Returns the element index or -1 if not found.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>array</td>
<td>Array</td>
<td>Array to search.</td>
</tr>
<tr>
<td>element</td>
<td>Object</td>
<td>Element to search for.</td>
</tr>
<tr>
<td>startIndex</td>
<td>Number</td>
<td>Index to start searching from.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Position of the element in the array, or -1 if the element is not in the array.</td>
</tr>
</tbody>
</table>

```javascript
var arrayUtil = new ArrayUtil();
var a1 = new Array("a", "b", "c", "d", "e");
gs.print("indexOf b starting at 0: " + arrayUtil.indexOf(a1, "b", 0));
gs.print("indexOf b starting at 2: " + arrayUtil.indexOf(a1, "b", 2));
```

Output:

indexOf b starting at 0: 1
indexOf b starting at 2: -1

## ArrayUtil - intersect(Array a, Array b)

Finds the elements present in all arrays.

Any number of arrays can be provided as parameters.

## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Array</td>
<td>An array</td>
</tr>
<tr>
<td>b</td>
<td>Array</td>
<td>An array</td>
</tr>
</tbody>
</table>
### ArrayUtil - intersect(Array a, Array b)

An array of elements from array a that were found in all of the other input arrays. Duplicates are removed.

```javascript
var arrayUtil = new ArrayUtil();
var a1 = new Array("a", "b", "c");
var a2 = new Array("c", "d", "e");
gs.print(arrayUtil.intersect(a1, a2));
```

Output: c

### ArrayUtil - union(Array a, Array b)

Merge two or more arrays.

Any number of arrays can be provided as parameters.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Array</td>
<td>An array</td>
</tr>
<tr>
<td>b</td>
<td>Array</td>
<td>An array</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of items from all the input arrays. Duplicates are removed.</td>
</tr>
</tbody>
</table>

```javascript
var arrayUtil = new ArrayUtil();
var a1 = new Array("a", "b", "c");
var a2 = new Array("c", "d", "e");
gs.print(arrayUtil.union(a1, a2));
```

Output: a,b,c,d,e

### ArrayUtil - unique(Array a)

Removes duplicate items from an array.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Array</td>
<td>The array to check for duplicate elements.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of unique items from the input array.</td>
</tr>
</tbody>
</table>

```javascript
var arrayUtil = new ArrayUtil();
var a1 = new Array("a", "b", "c", "c", "b");
gs.print(arrayUtil.unique(a1));
```

Output: a,c,b

**AssessmentCreation - Global**

The `AssessmentCreation` class provides methods that administrators can use to generate assessments and surveys.

There are multiple ways to call the `createAssessments` method. You must set the `typeID` parameter in all cases to identify a metric type or survey definition.

Some of the factors that influence how the method behaves include:

- Whether the target is an assessment or survey
- Whether the assessment schedule type is on-demand or scheduled
- Survey schedule periods
- Parameters provided

For example, when the method is called with only the `typeID` parameter set, the method searches for stakeholders or survey users to send invitations to. In contrast, when the method is called with the `typeID` and `userID` parameters set, it attempts to send invitations to the specified users.

**AssessmentCreation - createAssessments (String typeID, String sourceRecordID, String userID)**

Creates one or more assessments or surveys for the specified metric type or survey definition.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeID</td>
<td>String</td>
<td>The sys_id of the metric type or survey definition for which to generate assessments or surveys.</td>
</tr>
<tr>
<td>sourceRecordID</td>
<td>String</td>
<td>One or more comma-separated survey definition sys_ids to include in the assessments generated. There must be an assessable record associated with the specified metric type for each source record. If this parameter is left blank, the assessments generated includes all assessable records for the specified type. This parameter is for use with assessments only.</td>
</tr>
<tr>
<td>userID</td>
<td>String</td>
<td>One or more comma-separated sys_ids of users to which to send assessment or survey instances. If this parameter is left blank, the assessment stakeholders or survey users receive instances. This parameter is required for on-demand assessments.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>If the method is successful, it returns a comma-separated string such as 7a2ad2253784200044e0bfc8bcbe5de1, 10, b7e8b5d4c0a80169008b49e468920048 (Instance ID, number of instances, Group ID) with the following values:</td>
</tr>
<tr>
<td></td>
<td>• instanceID: Sys_id of the assessment or survey instance created, if there is only one. Sys_id of one of the instances, if there are multiple. If there are multiple instances, use the group ID to retrieve data.</td>
</tr>
<tr>
<td></td>
<td>• #instances: Number of assessment or survey instances created.</td>
</tr>
<tr>
<td></td>
<td>• groupID: Sys_id of the assessment group that contains the assessment or survey instances created.</td>
</tr>
<tr>
<td></td>
<td>If the method is unable to generate assessments or surveys, it returns one of the error codes.</td>
</tr>
</tbody>
</table>

### Error Codes

- isscheduled - Indicates that the userID or sourceRecordID parameter should not have been used. When the typeID value represents a metric type with the Schedule type field set to Scheduled, the typeID parameter is the only parameter that may be used.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>draft_state</td>
<td>Indicates that the <code>typeID</code> value is a metric type or survey definition that is not published. Publish the metric type or survey definition.</td>
</tr>
<tr>
<td>invalidtype</td>
<td>Indicates that there is no existing metric type that matches the <code>typeID</code> value.</td>
</tr>
<tr>
<td>inactive</td>
<td>Indicates that the <code>typeID</code> value is a metric type or survey definition that is not active. Navigate to the metric type or survey definition and select the <strong>Active</strong> check box.</td>
</tr>
<tr>
<td>not_available</td>
<td>Indicates that the <code>typeID</code> value is a metric type or survey definition that has no metrics or survey questions. Add at least one metric or survey question.</td>
</tr>
<tr>
<td>notyet</td>
<td>Indicates that at least one of the users in the <code>userID</code> value is assigned a survey instance for the specified survey definition, that has yet to expire. The error code returns the expiration date.</td>
</tr>
<tr>
<td>wip</td>
<td>Indicates that at least one of the users in the <code>userID</code> value is already assigned a survey instance for the survey.</td>
</tr>
<tr>
<td>not_authorized</td>
<td>Indicates that at least one of the users in the <code>userID</code> value is not authorized to take the survey. Verify that one of the following is true:</td>
</tr>
<tr>
<td></td>
<td>◦ There are no survey users for the specified survey definition.</td>
</tr>
<tr>
<td></td>
<td>◦ All the users in the <code>userID</code> value are saved as survey users for the specified survey definition.</td>
</tr>
<tr>
<td>completed</td>
<td>Indicates that the <code>typeID</code> value is a survey definition with the Schedule period field set to Only Once and that at least one of the users in the <code>userID</code> value has already completed a survey instance for that survey definition.</td>
</tr>
<tr>
<td>nousers</td>
<td>Indicates that the <code>userID</code> parameter was not used and that one of the following is true:</td>
</tr>
<tr>
<td></td>
<td>◦ There are no stakeholders associated with the specified metric type. Set the <code>userID</code> parameter or create at least one stakeholder.</td>
</tr>
<tr>
<td></td>
<td>◦ There are no survey users associated with the specified survey definition. Set the <code>userID</code> parameter or create at least one survey user.</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• nosources</td>
<td>Indicates that the <code>typeID</code> value is a metric type with the Schedule type field set to On demand and that neither the <code>userID</code> nor <code>sourceRecordID</code> parameter was used. Set at least one of these parameters.</td>
</tr>
</tbody>
</table>

### AuthCredential - Scoped

The `AuthCredential()` API provides methods that enable you to build credentials for a REST request.

Generate outbound signing requests using these APIs in the following order:

1. `HttpRequestData`: Build the API request.
2. `AuthCredential`: Create a credential object or update an existing one. Use the credential to sign the request through the `RequestAuthAPI` class.
3. `RequestAuthAPI`: Sign the request and return an `HttpRequestAuthedData` object.
4. `HttpRequestAuthedData`: Get information about the signed request.
5. `GlideHTTPRequest`: Send the signed request.

Before using these APIs, you must configure an authentication algorithm to sign the request and associate it with the credential used to authenticate the request.

You can create an empty AuthCredential object, instantiate an existing AuthCredential object, or use the `StandardCredentialsProvider` class to instantiate an AuthCredential object using a Credential record from the Credentials [discovery_credentials] table. If you create an empty AuthCredential object, use the `setAttribute()` method to add properties to the object.

See the following example to instantiate an AuthCredential using `StandardCredentialsProvider`:

```javascript
// Return an AuthCredential object using a Credential sys_id
var credential = new
    sn_cc.StandardCredentialsProvider().getAuthCredentialByID("5b61c16f73533300f662cff8af6a74b");
```

Use the `AuthCredential` API in scoped scripts with the `sn_auth` namespace identifier.
## AuthCredential - AuthCredential(Object authCredential)

Instantiates a new AuthCredential object, or modifies an existing one.

You can create an empty AuthCredential object, instantiate an existing AuthCredential object, or use the `StandardCredentialsProvider` class to instantiate an AuthCredential object using a Credential record from the Credentials [discovery_credentials] table. If you create an empty AuthCredential object, use the `setAttribute()` method to add properties to the object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authCredential</td>
<td>Object</td>
<td>Optional. Include this parameter to update an existing AuthCredential object.</td>
</tr>
</tbody>
</table>

## AuthCredential - getAttribute(String key)

Returns the value of an AuthCredential attribute.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Key of the attribute to return the value for. If you created an empty AuthCredential object, you must use the <code>setAttribute()</code> method to add properties to the object. If you used a credential record to instantiate the AuthCredential object, pass a field name from the Credential [discovery_credentials] table to access the value.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
// Define the HttpRequestData object
var endpoint = "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
```
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addHeader('x-amz-acl', 'public-read');

// Get AuthCredential object and set an attribute
var credential = new sn_auth.AuthCredential();
credential.setAttribute("user_name", "admin");

// Sign the request and return an AuthCredential attribute value
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var signingCredential = signingAPI.getAuthCredential();
name = signingCredential.getAttribute("name");

AuthCredential - setAttribute(String key, String value)
Sets an attribute for an AuthCredential object.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the attribute to set. If you created an empty AuthCredential object, you must use this method to add properties to the object. If you used a credential record to instantiate the AuthCredential object, pass a field name from the Credential [discovery_credentials] table to set the value.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value of the attribute.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

// Define the HttpRequestData object
var endpoint = "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.service('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addHeader('x-amz-acl', 'public-read');
```javascript
// Get AuthCredential object and set an attribute
var credential = new sn_auth.AuthCredential();
credential.setAttribute("user_name", "admin");
```

**AutomationException - Global**

Provides generic exception wrapping for automation. Include this in automation code to implement error handling.

**AutomationException - getMessage()**

Retrieves the error message.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The error message</td>
</tr>
</tbody>
</table>

```javascript
var ae = new AutomationException("This is an error message");
gs.print(ae.getMessage());
```

Output: This is an error message

**BusinessServiceManager - Global**

Unify service management by converting manually-created services to application services, populating application services, and adding/removing CIs and connections.

Use BusinessServiceManager methods in global server scripts. You must have the Application Service administrator [app_service_admin] role.

**BusinessServiceManager - addCI(String service_id, String source_id, String target_id)**

Adds a CI to a manually-created application service.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>service_id</td>
<td>String</td>
<td>Sys ID of the application service to add a CI to. Must be of type cmdb_ci_service_discovered.</td>
</tr>
<tr>
<td>source_id</td>
<td>String</td>
<td>Sys ID of the CI whose outgoing connection joins the target CI.</td>
</tr>
<tr>
<td>target_id</td>
<td>String</td>
<td>Sys ID of the CI to add to the service. Cannot be an excluded item. Excluded items are CIs added to the sa.mapping.user.manual.citype.blacklist system property that are excluded from specific operations.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
//Application service is 'PeopleSoft Portals'
//Source CI is windows server 'PS Apache01'
//Target CI is 'SAP WEB01'

var bsManager = new SNC.BusinessServiceManager();
var appService = bsManager.addCI('2fce42d80a0a0bb4004af34d7e3984c8', '3a27d4370a0a0bb4006316812bf45439', '3a6cadc1c0a8ce01001f1e5d0d7d68fa');
```

**BusinessServiceManager - addManualConnection(String source_id, Object manual_endpoint, String service_id)**

Adds a manually created connection to an application service.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_id</td>
<td>String</td>
<td>Sys ID of the CI to add to the connection.</td>
</tr>
<tr>
<td>manual_endpoint</td>
<td>Object</td>
<td>GlideRecord object in the cmdb_ci_endpoint_manual table to add to the application service.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>service_id</td>
<td>String</td>
<td>Sys ID of the application service to add the connection to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord("cmdb_ci_endpoint_manual");
now_GR.initialize();
now_GR.name='myEndpoint';
now_GR.insert();

var bsManager = new SNC.BusinessServiceManager();
var appService = bsManager.addManualConnection("3a307c930a0a0bb400353965d0b8861f", now_GR, "2fce42d80a0a0bb4004af34d7e3984c8");
```

**BusinessServiceManager - BusinessServiceManager()**

Instantiates a BusinessServiceManager object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var bsManager = new SNC.BusinessServiceManager();
```

**BusinessServiceManager - migrateManualToApplicationService(String service_id)**

Converts a manual service to an application service.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>service_id</td>
<td>String</td>
<td>Sys ID of the manual service to convert.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the migration to an application service succeeds.</td>
</tr>
</tbody>
</table>

```javascript
var bsManager = new SNC.BusinessServiceManager();
var appService = bsManager.migrateManualToApplicationService("451047c6c0a8016400de0ae6df9b9d76");
```

**BusinessServiceManager - populateApplicationService(String service_id, Number levels, String blacklisted_relation_types)**

Populates an application service with relations and CIs from the designated entry point.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>service_id</td>
<td>String</td>
<td>Sys ID of the application service to populate.</td>
</tr>
<tr>
<td>levels</td>
<td>Number</td>
<td>Number of levels of CIs to add from the CMDB.</td>
</tr>
<tr>
<td>blacklisted_relation_types</td>
<td>String</td>
<td>Relation types to exclude when populating the application service.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the operation is successful.</td>
</tr>
</tbody>
</table>

```javascript
var bsManager = new SNC.BusinessServiceManager();
var appService = bsManager.populateApplicationService("5bf65ebeedb91300964f6fa662989533", 10, "Depends on::Used by");
```

**BusinessServiceManager - removeCI(String service_id, String target_id)**

Removes a manually created CI from an application service.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>service_id</td>
<td>String</td>
<td>Sys ID of the application service to remove a CI from.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> Passing a GlideRecord object is also supported.</td>
</tr>
<tr>
<td>target_id</td>
<td>String</td>
<td>Sys ID of the CI to remove from the application service.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> Passing a GlideRecord object is also supported.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var bsManager = new SNC.BusinessServiceManager();
var appService = bsManager.removeCI("2fce42d80a0a0bb4004af34d7e3984c8", "3a290cc60a0a0bb400000bdb386af1cf");
```

### BusinessServiceManager - removeManualConnection(String source_id, String endpoint_id, String service_id)

Removes a manually created connection and the connected CI from an application service.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_id</td>
<td>String</td>
<td>Sys ID of the CI connected to the endpoint.</td>
</tr>
<tr>
<td>endpoint_id</td>
<td>String</td>
<td>Sys ID of the manually created connection to remove from the application service</td>
</tr>
<tr>
<td>service_id</td>
<td>String</td>
<td>Sys ID of the application service to remove the connection from.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var bsManager = new SNC.BusinessServiceManager();
var appService = bsManager.removeManualConnection("3a307c930a0a0bb400353965d0b8861f",
  "60ce3176ed91300964f6fa6629895d1", "2fce42d80a0a0bb4004af34d7e3984c8");

**Cabrillo JS constants**
These are the constants used by Cabrillo JS.

**Cabrillo JS button styles**
Constants to use when setting the style of a button.

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cabrillo.viewLayout.MORE_MENU_BUTTON_STYLE</td>
<td>String</td>
<td>Indicates that the button appears in the navigation bar's overflow button menu.</td>
</tr>
<tr>
<td>cabrillo.viewLayout.REPLACE_BACK_BUTTON_STYLE</td>
<td>String</td>
<td>Replaces the native back button with a Cabrillo button.</td>
</tr>
</tbody>
</table>

**Note:** Use this style with caution. If the webpage does not restore the native back button after using the Cabrillo replacement, the user may not be able to navigate back.

**Cabrillo JS close button styles**
Constants to use when setting the close button style of modal interfaces.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cabrillo.modal.CLOSE_BUTTON_STYLE_CANCEL</td>
<td>String</td>
<td>The modal has a localized cancel button to close the modal.</td>
</tr>
<tr>
<td>cabrillo.modal.CLOSE_BUTTON_STYLE_CLOSE</td>
<td>String</td>
<td>The modal has a localized close button to close the modal.</td>
</tr>
</tbody>
</table>
Properties (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cabrillo.modal.CLOSE_BUTTON_STYLE_DONE</td>
<td>String</td>
<td>The modal has a localized done button to close the modal.</td>
</tr>
</tbody>
</table>

**Cabrillo JS modal presentation styles**

Constants to use when setting the presentation style of modal interfaces.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cabrillo.modal.MODAL_PRESENTATION_STYLE_FULL_SCREEN</td>
<td>String</td>
<td>The modal is presented full screen.</td>
</tr>
<tr>
<td>cabrillo.modal.MODAL_PRESENTATION_STYLE_FORM_SHEET</td>
<td>String</td>
<td>The modal is presented as a form sheet, but may adapt to full screen on small devices.</td>
</tr>
</tbody>
</table>

**Cabrillo JS data types**

These are common data types used by Cabrillo JS APIs.

**Cabrillo.Attachment**

Defines an attachment.

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>content_type</td>
<td>String</td>
<td>The attachment content type.</td>
</tr>
<tr>
<td>ext</td>
<td>String</td>
<td>The attachment file extension.</td>
</tr>
<tr>
<td>file_name</td>
<td>String</td>
<td>The full file name of the attachment including the extension.</td>
</tr>
<tr>
<td>sys_created_by</td>
<td>String</td>
<td>The user name of the user that created the attachment.</td>
</tr>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The attachment record sys_id.</td>
</tr>
<tr>
<td>sys_updated_on</td>
<td>String</td>
<td>The date the attachment record was updated.</td>
</tr>
</tbody>
</table>
### Property (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table_name</td>
<td>String</td>
<td>Name of the table containing the record to which the attachment is attached.</td>
</tr>
<tr>
<td>table_sys_id</td>
<td>String</td>
<td>The sys_id of the record that to which the attachment is attached.</td>
</tr>
<tr>
<td>thumbnail</td>
<td>String</td>
<td>The URL for the image thumbnail of the attachment. Only applies to image attachments.</td>
</tr>
</tbody>
</table>

### Cabrillo.Button

Defines a button. Different interface contexts may support a subset of a button's properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>String</td>
<td>Title of the button. The string should be localized.</td>
</tr>
<tr>
<td>enabled</td>
<td>Boolean</td>
<td>Indicates whether the button is enabled.</td>
</tr>
<tr>
<td>badgeCount</td>
<td>Number</td>
<td>Number value to display on the button's badge.</td>
</tr>
<tr>
<td>backgroundColor</td>
<td>String</td>
<td>Background color of the button or badge. A string containing hex, RGB, HSL an x11 named color.</td>
</tr>
<tr>
<td>textColor</td>
<td>String</td>
<td>The text color of the badge or button. A string containing hex, RGB, HSL an x11 named color.</td>
</tr>
<tr>
<td>buttonStyle</td>
<td>String</td>
<td>Used to customize the style of the button.</td>
</tr>
<tr>
<td>imageName</td>
<td>String</td>
<td>Specifies an image. Possible values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• add: Displays an add button.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• back: Displays a back arrow image to navigate back.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• cart: Displays a shopping cart.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• close: Displays an X image to navigate back.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• edit: Displays an edit button.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• filter: Displays a filter button.</td>
</tr>
</tbody>
</table>
## Property (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|          |      | - menu: Displays a menu button.  
|          |      | - search: Displays a search button.  |

### Cabrillo.ListSelection

Describes a selected list item.

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>The value of the selected item. Default is the sys_id of the record, or the value of a custom reference key.</td>
</tr>
<tr>
<td>displayValue</td>
<td>String</td>
<td>The display value of the selected item.</td>
</tr>
</tbody>
</table>

### Cabrillo.Location

Contains geographical information, latitude, longitude, and direction.

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>altitude</td>
<td>Number</td>
<td>The altitude in meters.</td>
</tr>
<tr>
<td>coordinate</td>
<td>Cabrillo.LocationCoordinate</td>
<td>The geographical coordinates.</td>
</tr>
<tr>
<td>direction</td>
<td>Number</td>
<td>Measured in degrees beginning at due north.</td>
</tr>
<tr>
<td>speed</td>
<td>Number</td>
<td>Measured in meters per second.</td>
</tr>
<tr>
<td>timestamp</td>
<td>String</td>
<td>The date that the location was last updated.</td>
</tr>
</tbody>
</table>
Cabrillo.LocationCoordinate
Contains a geographical coordinate.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>latitude</td>
<td>Number</td>
<td>The latitude in degrees.</td>
</tr>
<tr>
<td>longitude</td>
<td>Number</td>
<td>The longitude in degrees.</td>
</tr>
</tbody>
</table>

Cabrillo.ModalResponse
Contains the result passed from a modal window that was dismissed.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Object</td>
<td>The result from the dismissed modal window.</td>
</tr>
</tbody>
</table>

Cabrillo.NavigationRequest
Specifies a request to navigate to a list or record.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>The table to navigate to. Required for both list and record navigation.</td>
</tr>
<tr>
<td>sysId</td>
<td>String</td>
<td>The sys_id of the record to navigate to. Required for record navigation.</td>
</tr>
<tr>
<td>query</td>
<td>String</td>
<td>An encoded query for the requested list. Can be used to seed values for record navigation requests.</td>
</tr>
<tr>
<td>view</td>
<td>String</td>
<td>The list or record view.</td>
</tr>
</tbody>
</table>
**Cabrillo.Rect**
Contains the location and dimensions of a rectangle.

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>Number</td>
<td>The x origin of the rectangle.</td>
</tr>
<tr>
<td>y</td>
<td>Number</td>
<td>The y origin of the rectangle.</td>
</tr>
<tr>
<td>width</td>
<td>Number</td>
<td>The width in points.</td>
</tr>
<tr>
<td>height</td>
<td>Number</td>
<td>The height in points.</td>
</tr>
</tbody>
</table>

**cabrillo - Client**
Cabrillo JS is a client side JavaScript API for accessing capabilities inside ServiceNow native mobile applications.

Use Cabrillo JS to build a custom mobile experience in ServiceNow native mobile applications. Cabrillo JS provides APIs to use native device capabilities, native mobile UI, and other ServiceNow mobile functionality.

Cabrillo JS is available in AngularJS as an injectable parameter. For example, you can inject Cabrillo JS in a Service Portal widget client script.

```javascript
function(cabrillo) {
    var c = this;

    // Used to determine if Cabrillo is executing in ServiceNow's native mobile apps.
    c.isNative = cabrillo.isNative();
}
```

Cabrillo JS APIs are only supported when executing in ServiceNow native mobile applications. Use the `cabrillo.isNative()` API to determine if the script is executing in a native context.

⚠️ **Note:** Cabrillo JS APIs are unavailable in GlideForm client scripts.

**cabrillo - isNative()**
Boolean value indicating if Cabrillo JS is executing a ServiceNow native mobile application.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if Cabrillo JS is running in a native context; otherwise, false.</td>
</tr>
</tbody>
</table>

```javascript
var isNative = cabrillo.isNative();
```

cabrillo.attachments - Client

The name space for Cabrillo JS attachment functions. This enables adding and viewing attachments.

cabrillo.attachments - addFile(String tableName, String sysId, Object params, String options)

 Presents a document picker and uploads the selected file.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Table name of the record to which to attach the attachment.</td>
</tr>
<tr>
<td>sysId</td>
<td>String</td>
<td>The sys_id of the record to which to attach the attachment.</td>
</tr>
<tr>
<td>params</td>
<td>Object</td>
<td>Reserved for future use. Set to null.</td>
</tr>
<tr>
<td>options</td>
<td>String</td>
<td>Reserved for future use. Set to null.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>promise</td>
<td>If successful a Cabrillo.Attachment object. If the operation fails, an error.</td>
</tr>
</tbody>
</table>
var table = 'incident';
var sysID = 'a9e30c7dc61122760116894de7bcc7bd';

cabrillo.attachments.addFile(table,
    sysID,
    null,
    null
).then(function(attachment) {
    if (attachment) {
        console.log('Added a new file.', attachment);
    } else {
        console.log('User cancelled adding an attachment.');
    }
}, function(error) {
    console.log('Failed to attach new file.', error);
});

cabrillo.attachments -viewFile(Cabrillo.Attachment attachment, Cabrillo.Rect sourceRect, String sourceBase64Image)

Presents a document picker and uploads the selected file.

⚠️ **Note:** Scaling using a sourceRect parameter with a sourceBase64 image is only supported on iOS. Android ignores these parameters and opens the image without a scaling animation.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attachment</td>
<td>Cabrillo.Attachment</td>
<td>Describes the attachment to view.</td>
</tr>
<tr>
<td>sourceRect</td>
<td>Cabrillo.Rect</td>
<td>Optional. Describes the source rectangle of the image to scale up.</td>
</tr>
<tr>
<td>sourceBase64Image</td>
<td>String</td>
<td>Optional. A base64 representation of the source image to scale up.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>promise</td>
<td>If successful, is undefined; otherwise an error.</td>
</tr>
</tbody>
</table>
To scale an image that was tapped into a native image viewer, the `viewFile()` method accepts optional arguments for the image's rectangle on the page as well as a base64 encoded thumbnail of the image. The thumbnail is scaled into the full size image with an animation.

```javascript
// Grab image metadata from an image that was tapped
var imageMetadata = imageMetadataFromEvent(event);

// Optional rect of image on page
var imageRect = imageMetadata.rect;

// Optional base64 encoded image to scale up into native viewer
var base64EncodedImage = imageMetadata.base64;

// A Cabrillo.Attachment dictionary to view
var attachment = {
    sys_id: '8e99daa3ff133100ba13fffffffffff2d',
    content_type: 'image/jpeg',
    path: '8e99daa3ff133100ba13fffffffffff2d.iix'
}

cabrillo.attachments.viewFile(attachment,
    imageRect,
    base64EncodedImage
).then(function() {
    // It worked. Nothing to do here.
}, function(error) {
    console.log('Failed to view file.', error);
});
```
console.log('Failed to view file.', error);
});

**cabrillo.flowControl - Client**

Cabrillo JS flow control functions.

**cabrillo.flowControl - flowEnded(optional String flowName)**

Ends the current flow.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flowName</td>
<td>String</td>
<td>Optional. Name for the flow to end.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
cabrillo.flowControl.flowEnded('TestFlow');
```

**cabrillo.message - Client**

Cabrillo JS functions to provide access to the native UI banner.

**cabrillo.message - showMessage(String style, String title)**

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>style</td>
<td>String</td>
<td>Style of the banner. Possible values are described in the message styles section below.</td>
</tr>
<tr>
<td>title</td>
<td>String</td>
<td>Title to show in the banner.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
**Message Styles**

Constants to use when setting the message style.

<table>
<thead>
<tr>
<th>Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cabrillo.message.SUCCESS_MESSAGE_STYLE</td>
<td>Message will have Success styling.</td>
</tr>
<tr>
<td>cabrillo.message.ERROR_MESSAGE_STYLE</td>
<td>Message will have Error styling</td>
</tr>
</tbody>
</table>

```javascript
 cabrillo.message.showMessage(cabrillo.message.SUCCESS_MESSAGE_STYLE, 'This is a sample message');
```

**cabrillo.modal - Client**

Cabrillo JS functions for presenting web content inside of native modals.

**cabrillo.modal - presentModal( String title, String url, String closeButtonStyle, String modalPresentationStyle)**

Presents content in a native modal interface.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>String</td>
<td>Title of the modal interface.</td>
</tr>
<tr>
<td>url</td>
<td>String</td>
<td>The URL to open the modal. This must be an internal instance URL (fully qualified or relative; a relative URL is preferred).</td>
</tr>
<tr>
<td>closeButtonStyle</td>
<td>String</td>
<td>Must be one of cabrillo.modal.CLOSE_BUTTON_STYLE_CANCEL,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>cabrillo.modal.CLOSE_BUTTON_STYLE_CLOSE, or</td>
</tr>
<tr>
<td></td>
<td></td>
<td>cabrillo.modal.CLOSE_BUTTON_STYLE_DONE.</td>
</tr>
<tr>
<td>modalPresentationStyle</td>
<td>String</td>
<td>Must be cabrillo.modal.MODAL_PRESENTATION_STYLE_FULL_SCREEN, or</td>
</tr>
<tr>
<td></td>
<td></td>
<td>cabrillo.modal.MODAL_PRESENTATION_STYLE_FORM_SHEET.</td>
</tr>
</tbody>
</table>
Present a native modal that loads a custom URL. This presents a custom Service Portal page in a form sheet style modal. The promise is fulfilled when the modal is dismissed. See the `dismissModal()` function for custom dismissal capabilities.

```javascript
cabrillo.modal.presentModal('Portal Page',
   '/$sp.do?id=my_modal_page',
   cabrillo.modal.CLOSE_BUTTON_STYLE_CLOSE,
   cabrillo.modal.MODAL_PRESENTATION_STYLE_FORM_SHEET
 ).then(function(response) {
   // The results from the modal are in a results key on the response object.
   var results = response && response.results ? response.results : null;

   if (results) {
     console.log('Modal dismissed with results.', results);
   } else {
     console.log('Modal dismissed without results.');
   }
}, function(error) {
   console.log(error);
});
```

cabrillo.navigation - Client

Cabrillo JS functions for forward and backward navigation.

cabrillo.navigation - goBack()

Enables backward navigation.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if Cabrillo JS will navigate backward; otherwise false.</td>
</tr>
</tbody>
</table>

```javascript
cabrillo.navigation.goBack();
```

### `cabrillo.navigation - goto(String url, Cabrillo.NavigationRequest request)`

Enables forward navigation.

Use the request parameter not the url parameter for list or record navigation.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>An URL to navigate to. This should be used for custom URL navigation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Optional.</td>
</tr>
<tr>
<td>request</td>
<td>Cabrillo.NavigationRequest</td>
<td>Describes the list or record to navigate to.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Optional.</td>
</tr>
</tbody>
</table>

```javascript
cabrillo.navigation.goto('/$sp.do?id=my_custom_page');
```

Navigate to a list. The request parameter is preferred over the url parameter for list navigation.

```javascript
// A Cabrillo.NavigationRequest dictionary that specifies a list of active incidents.
var request = {
  table: 'incident',
  query: 'active=true',
};
cabrillo.navigation.goto(null, request);
```
Navigate to a record. The request parameter is preferred over the url parameter for record navigation.

var request = {
    table: 'incident',
    sysId: 'a9e30c7dc61122760116894de7bcc7bd'
};
cabrillo.navigation.goto(null, request);

Navigate to a new record.

// The new record will be seeded with the encoded query.
var request = {
    table: 'incident',
    sysId: '-1',
    query: 'short_description=This is a new incident.'
};
cabrillo.navigation.goto(null, request);

cabrillo.viewLayout - Client

Cabrillo JS functions to provide access to native UI elements like buttons and spinners.

cabrillo.viewLayout - hideSpinner()

Hides a native spinner in the current interface.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
**cabrillo.viewLayout - setBottomButtons( Array buttons, Function handler)**

Set buttons at the bottom of the current interface.

Images and badges are not supported in bottom buttons.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>buttons</td>
<td>Array of Cabrillo.Button objects</td>
<td>Describes the buttons to set. A maximum of one button is currently supported.</td>
</tr>
<tr>
<td>handler</td>
<td>Function</td>
<td>The function to call when a button is tapped. The function has no return value and takes the selected button index as its only parameter. The function must have a single parameter that is a Number.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>promise</td>
<td>If successful, is undefined; otherwise, an error.</td>
</tr>
</tbody>
</table>

**Set bottom buttons**

```javascript
var buttons = [
    {
        title: 'Add to Cart',
        enabled: true,
        backgroundColor: '#3091F9',
        textColor: '#FFFFFF'
    }
];

cabrillo.viewLayout.setBottomButtons(buttons, function(buttonIndex) {
    console.log('Received an event from the button.');
}).then(function() {
    console.log('Buttons were set.');
}, function() {
    console.log('Failed to register buttons.');
});
```

**Clear bottom buttons**
cabrillo.viewLayout.setBottomButtons();

cabrillo.viewLayout.setTitle(String title)
Set the current interface title.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>String</td>
<td>The title of the interface.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

cabrillo.viewLayout.setTitle('My Title');

cabrillo.viewLayout.setNavigationBarButtons(Array buttons, Function handler)
Set buttons in the navigation bar of the current interface.

Images and badges for buttons that appear in the overflow button menu are omitted. For this reason, it's best to provide a title and an image name when setting an image button in the navigation bar.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>buttons</td>
<td>Array of Cabrillo.Button objects</td>
<td>Describes the buttons to set. Buttons may overflow into an additional menu as needed.</td>
</tr>
<tr>
<td>handler</td>
<td>Function</td>
<td>The function to call when a button is tapped. The function has no return value and takes the selected button index as its only parameter. The function must have a single parameter that is a Number.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>promise</td>
<td>If successful is undefined; otherwise an error.</td>
</tr>
</tbody>
</table>

Set navigation bar buttons.

```javascript
var buttons = [
    {
        title: 'Save',
        enabled: true
    }
];

cabrillo.viewLayout.setNavigationBarButtons(buttons, function(buttonIndex) {
    console.log('Received an event from the button.');
}).then(function() {
    console.log('Buttons added.');
}, function() {
    console.log('Failed to register buttons.');
});
```

Handle multiple buttons.

```javascript
var buttons = [
    {
        title: 'Save',
        enabled: true
    },
    {
        title: 'Delete',
        enabled: true
    }
];

cabrillo.viewLayout.setNavigationBarButtons(buttons, function(buttonIndex) {
    switch (buttonIndex) {
        case 0:
            console.log('Received an event from the Save button.');
            break;
        case 1:
            console.log('Received an event from the Delete button.');
            break;
    }
});
```
Buttons placed in the navigation bar can be represented by an image.

```javascript
var buttons = [
    {
        title: 'Compose',
        imageName: 'compose',
        enabled: true
    }
];

cabrillo.viewLayout.setNavigationBarButtons(buttons, function(buttonIndex) {
    console.log('Received an event from the button.');
}).then(function() {
    console.log('Buttons were set.');
}, function() {
    console.log('Failed to register buttons.');
});
```

A native back button is shown in the native app by default, but you can replace the back button with a Cabrillo button by setting a button’s `buttonStyle` property.

```javascript
var buttons = [{
    title: 'Cancel',
    imageName: 'close',
    buttonStyle: cabrillo.viewLayout.REPLACE_BACK_BUTTON_STYLE,
    enabled: true
}, {
    title: 'Done',
    enabled: true
}];

cabrillo.viewLayout.setNavigationBarButtons(buttons, function(buttonIndex) {
    switch (buttonIndex) {
        case 0:
            c.log('Cancel button was clicked.');
            break;
        case 1:
            c.log('Done button was clicked.');
    }
});
```
Buttons are placed in the overflow button menu as needed. To force a button into the overflow button menu, set the button's buttonStyle property.

```javascript
var buttons = [
    { 
        title: 'Save', 
        buttonStyle: cabrillo.viewLayout.MORE_MENU_BUTTON_STYLE, 
        enabled: true 
    }
];

cabrillo.viewLayout.setNavigationBarButtons(buttons, function(buttonIndex) {
    console.log('Received an event from the button.'); 
}).then(function() { 
    console.log('Buttons were set.'); 
}, function() { 
    console.log('Failed to register buttons.'); 
});
```

Buttons may have a badge when placed in the navigation bar. This example sets a button with a shopping cart icon and a badge count of 3. The badge has a blue background with white text.

```javascript
var buttons = [
    { 
        title: 'Cart',
        imageName: 'cart',
        badgeCount: 3,
        backgroundColor: '#3091F9',
        textColor: '#FFFFFF',
        enabled: true
    }
];

cabrillo.viewLayout.setNavigationBarButtons(buttons, function(buttonIndex) {
    console.log('Received an event from the button.'); 
}).then(function() { 
    console.log('Buttons were set.'); 
}, function() { 
    console.log('Failed to register buttons.'); 
});
```
console.log('Buttons were set.');

}, function() {
    console.log('Failed to register buttons.');
});

Clear navigation bar buttons.

cabrillo.viewLayout.setNavigationBarButtons();

cabrillo.viewLayout - showSpinner()

Shows a native spinner in the current interface.

```
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Returns

```
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
```

cabrillo.viewLayout.showSpinner();

CartJS - Scoped

CartJS API enables you to access the shopping cart for a user.

To use this class in a scoped application, use the `sn_sc` namespace identifier. The Service Catalog Scoped API plugin (ID: com.glideapp.servicecatalogScoped.api) that is enabled by default is required to access the CartJS API.

CartJS - addToCart(Map request)

Adds the request for a catalog item to the current cart.

```
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request</td>
<td>Map</td>
<td>A JSON object that contains the details of the catalog item to be added to the cart.</td>
</tr>
</tbody>
</table>
```
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item_id</td>
<td>sys_id of the item to be added to the cart</td>
<td></td>
</tr>
<tr>
<td>item_quantity</td>
<td>Number of items to be added. Default value is 1.</td>
<td></td>
</tr>
<tr>
<td>var_name</td>
<td>Name of the question.</td>
<td></td>
</tr>
<tr>
<td>var_value</td>
<td>Value of the answer (Not the display value).</td>
<td></td>
</tr>
</tbody>
</table>

The structure of the request object is:

```
{
  'sysparm_id': item_id,
  'sysparm_quantity': item_quantity,
  'variables':{
    'var_name': 'var_value',
    ...
  }
}
```

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSON</td>
<td>Structure of the current cart.</td>
</tr>
</tbody>
</table>

```
{
  'subtotal': value,
  'items':[
    {
      itemName:'',
      quantity:'',
      price:'',
      recurring_price:''
    }
  ]
}
```

This example shows how you can add multiple items to the cart. Note that `var cart = new sn_sc.CartJS();` is inside the loop. You must call `sn_sc.CartJS()` multiple times to be able to add multiple items to the cart.

```javascript
for (i = 0; i < 2; i++) {
  var cart = new sn_sc.CartJS();
```
var item =
{
'sysparm_id': '0d08837237153000158bbfc8bcbe5d02',
'sysparm_quantity': '1',
'veariables':{
'carrier': 'at_and_t_mobility',
'data_plan': '500MB',
'duration': 'eighteen_months',
'color': 'slate',
'storage': 'sixtyfour'
}};
var cartDetails = cart.addToCart(item);
gs.info(JSON.stringify(cartDetails));
}

Output

//The cart after the first item is added
{
"cart_id": "c0f5828a1b476010593876a61a4bcb71",
"subtotal": "$599.99",
"items": [
{
"catalog_item_id": "0d08837237153000158bbfc8bcbe5d02",
"quantity": "1",
"localized_price": "$599.99",
"price": "$599.99",
"recurring_frequency": "Monthly",
"localized_recurring_price": "$30.00",
"recurring_price": "$29.00",
"item_name": "Apple iPhone 5",
"cart_item_id": "40f5828a1b476010593876a61a4bcb72"
}
]
}

//The cart after the second item is added
{
"cart_id": "c0f5828a1b476010593876a61a4bcb71",
"subtotal": "$1,199.98",
"items": [
{
"catalog_item_id": "0d08837237153000158bbfc8bcbe5d02",
"quantity": "1",
"localized_quantity": 1,
"localized_price": "$599.99",
"price": "$599.99",
"recurring_frequency": "Monthly",
"localized_recurring_price": "$30.00",
"recurring_price": "$29.00",
"item_name": "Apple iPhone 5",
"cart_item_id": "40f5828a1b476010593876a61a4bcb72"
},
{
  "catalog_item_id": "0d08837237153000158bbfc8bcbe5d02",
  "quantity": "1",
  "localized_price": "$599.99",
  "price": "$599.99",
  "recurring_frequency": "Monthly",
  "localized_recurring_price": "$30.00",
  "recurring_price": "$29.00",
  "item_name": "Apple iPhone 5",
  "cart_item_id": "98f5828a1b476010593876a61a4bcb75"
}
]

CartJS - CartJS(String cartName)

Creates an instance of the CartJS class with the name of a defined cart for the user who is currently logged in.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cartName</td>
<td>String</td>
<td>Optional. Name of a defined cart for the user who is currently logged in.</td>
</tr>
</tbody>
</table>

var cart = new sn_sc.CartJS(cart1);

CartJS - checkoutCart()

Performs the cart checkout. If the two-step checkout is enabled, returns the order summary. If the two-step checkout is disabled, the cart is submitted and details of the generated request are returned.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSON</td>
<td>If the two-step checkout is enabled, the summary of the items in the cart is returned.</td>
</tr>
</tbody>
</table>

```
{
  "subtotal_price": "$2,748.49",
  "subtotal_recurring_frequency": "",
  "delivery_address": "",
  "special_instructions": "",
  "total_title": "Total",
  "requested_for_user": "System Administrator",
  "requested_for": "6816f79cc0a8016401c5a33be04be441",
  "daily": [{"frequency_subtotal": "", "items": [{}, {}, ...], ...}, ...],
  "monthly": [{"frequency_subtotal": "", "items": [{}, {}, ...], ...}, ...],
  "annually": [{"frequency_subtotal": "", "items": [{}, {}, ...], ...}, ...],
  "none": [{"frequency_subtotal": "", "items": [{}, {}, ...], ...}, ...]
}
```

If the two-step checkout is disabled:

```
{
  "request_id": "sys_id of the generated request",
  "request_number": "Number of the generated request"
}
```

Example

```javascript
var cart = new sn_sc.CartJS();
var checkoutInfo = cart.checkoutCart();
gs.info(checkoutInfo);
```

Output

If two step checkout enabled:

```json
{"subtotal_price": "$2,748.49", "subtotal_recurring_frequency": ",", "delivery_address": ",", "special_instructions": ",", "total_title": "Total", "requested_for_user": "System Administrator", "requested_for": "6816f79cc0a8016401c5a33be04be441", "weekly": {"subtotal_price": "$399.50", "subtotal_recurring_frequency": "Weekly", "subtotal_recurring_price": "$0.00", "total_title": "Total", "items": [{"catalog_item_id": "e90a07237153000158bbfc8bcbe5d7f", "variable
```
CartJS - empty()

Deletes the current cart.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var cart = new sn_sc.CartJS();
cart.empty();

CartJS - getCartID()
Returns the ID of the current cart.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id for the current cart.</td>
</tr>
</tbody>
</table>

Returns the sys_id of the current cart.

```javascript
var cart = new sn_sc.CartJS();
var cartId = cart.getCartID();
gs.info(cartId);
```

Output

```
039c516237b1300054b6a3549dbe5dfc
```

CartJS - getCartItems()
Returns a GlideRecord object containing records for items in the current cart.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>GlideRecord object including records and fields from the Item [sc_cart_item] table that describe an item in the current cart.</td>
</tr>
</tbody>
</table>
Returns quantity and Catalog Item field values for items in the current cart.

```javascript
var cart = new sn_sc.CartJS();

var cartItems = cart.getCartItems();
while (cartItems.next()) {
    gs.info(cartItems.getElement('quantity').getDisplayValue()
            + " x "
            + cartItems.getElement('cat_item').getDisplayValue());
}
```

Output

```
1 x Apple iPhone 6s Plus
1 x Apple iPhone 6s
1 x Apple MacBook Pro 15"
```

**CartJS - getDeliveryAddress()**

Returns the delivery address for the current cart.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Delivery address for the current cart.</td>
</tr>
</tbody>
</table>

Sets and then returns the delivery address for the current cart.

```javascript
var cart = new sn_sc.CartJS();
cart.setDeliveryAddress("Brasilia, Brasil");
var deliveryAddress = cart.getDeliveryAddress();
gs.info(deliveryAddress);
```

Output

```
Brasilia, Brasil
```
CartJS - getRequestedFor()

Returns the sys_id of the user for whom the cart is requested.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the user record for whom the cart is requested; located in the User [sys_user] table.</td>
</tr>
</tbody>
</table>

Returns the user sys_id associated with the cart.

```javascript
var cart = new sn_sc.CartJS();
cart.setRequestedFor("039c516237b1300054b6a3549dbe5dfc")
var userId = cart.getRequestedFor();
gs.info(userId);
```

Output

```
039c516237b1300054b6a3549dbe5dfc
```

CartJS - getRequestedForDisplayName()

Returns the name of the user for whom the current cart is requested.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the user for whom the current cart is requested; located in the User [sys_user] table.</td>
</tr>
</tbody>
</table>

Returns the name of the user associated with the cart.
var cart = new sn_sc.CartJS();
var userName = cart.getRequestedForDisplayName();
gs.info(userName);

Output

Abel Tuter

**CartJS - getSpecialInstructions()**

Returns the special instructions for the current cart.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Special instructions for the current cart.</td>
</tr>
</tbody>
</table>

This example shows how to set and retrieve the special instructions for the current cart.

```javascript
var cart = new sn_sc.CartJS();
cart.setSpecialInstructions("Delivery before 8 AM.");
var specInstruction = cart.getSpecialInstructions();
gs.info(specInstruction);
```

Output

Delivery before 8 AM.

**CartJS - orderNow(Map request)**

Orders a single item. If two-step checkout is enabled, the method adds the specified item to the cart and returns the sys_id of the cart. If two-step checkout is disabled, the method completes the purchase of the specified item and returns the sys_id of the generated request.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request</td>
<td>Map</td>
<td>JSON object that contains details of the catalog item to order.</td>
</tr>
<tr>
<td>request.delivery_address</td>
<td>String</td>
<td>Address to which to deliver the items. Default: Address of user</td>
</tr>
<tr>
<td>request.sysparm_id</td>
<td>String</td>
<td>Required. The sys_id of the item to purchase.</td>
</tr>
<tr>
<td>request.special_instructions</td>
<td>String</td>
<td>Instructions to follow when processing the order.</td>
</tr>
<tr>
<td>request.sysparm_quantity</td>
<td>String</td>
<td>Quantity of the specified item to purchase. Default: 1</td>
</tr>
<tr>
<td>request.sysparm_requested_for</td>
<td>String</td>
<td>The sys_id of the user for whom the item is requested. Default: Session user</td>
</tr>
<tr>
<td>request.variables</td>
<td>Array</td>
<td>Questions and customer answers associated with the item.</td>
</tr>
<tr>
<td>request.variables.var_name</td>
<td>String</td>
<td>Name of the question.</td>
</tr>
<tr>
<td>request.variables.var_value</td>
<td>String</td>
<td>Customer's response to the associated question.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSON key/value pairs</td>
<td>Output if two-step checkout is enabled:</td>
</tr>
</tbody>
</table>
|                             | {{
|                             |   'cart_id' : '<sys_id of the cart to which the items were added>'
|                             | }}                                                                           |
|                             | Output if two-step checkout is disabled:                                    |
|                             | {{
|                             |   'request_id' : '<sys_id of the generated request>'
|                             | }}                                                                           |
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>'request_number'</td>
<td>'&lt;Number of the generated request&gt;'</td>
</tr>
</tbody>
</table>

**Example**

```javascript
var cart = new sn_sc.CartJS();
var request = {
  'sysparm_id': '0d08837237153000158bbfc8bcbe5d02',
  'sysparm_quantity': '1',
  'variables':{
    'carrier': 'at_and_t_mobility',
    'data_plan': '500MB',
    'duration': 'eighteen_months',
    'color': 'slate',
    'storage': 'sixtyfour'
  }
}
var cartDetails = cart.orderNow(request);
gs.info(cartDetails);
```

**Output**

```
// If two-step checkout is enabled:
{"cart_id":"55384df3c322320076173b0ac3d3aec5"}

// If two-step checkout is disabled:
{"request_id":"4c690137c322320076173b0ac3d3ae03", "request_number": "REQ0010003"}
```

**CartJS - setDeliveryAddress(String address)**

Sets the delivery address for the current cart.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>String</td>
<td>Delivery address for the current cart.</td>
</tr>
</tbody>
</table>
CartJS - setSpecialInstructions(String specialInstructions)
Sets the special instructions for the current cart.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>specialInstructions</td>
<td>String</td>
<td>Special instructions for the current cart.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CartJS - submitOrder(Map request)
Updates special instructions, requested for, and delivery address from the request parameter and performs the cart checkout. Use this API to modify the mentioned parameters of the cart and perform the cart checkout simultaneously. Missing parameters in the request object will have their default value.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request</td>
<td>Map</td>
<td>A JSON object that contains details of the cart to be submitted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The structure of the request object is:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>`{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'special_instructions' : 'instructions',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'sysparm_requested_for' : requested_for,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'delivery_address' : 'address'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• instructions: Special instructions for the request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• requested_for : sys_id of the requested_for user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• address: Delivery address for the request.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSON</td>
<td>Structure of the cart.</td>
</tr>
<tr>
<td></td>
<td>`{</td>
</tr>
<tr>
<td></td>
<td>'request_id' : 'sys_id of the generated Request',</td>
</tr>
<tr>
<td></td>
<td>'request_number' : 'Number of the generated Request'</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

### Example

```javascript
var cart = new sn_sc.CartJS();
var request =
{
   'special_instructions' : 'Delivery only in working hours',
   'requested_for' : '62826bf03710200044e0bfc8bcbe5df1',
   'delivery_address' : "Brasilia, Brasil",
};
var requestDetails = cart.submitOrder(request);
gs.info(requestDetails);
```

### Output

```javascript
{"request_id":"6eed229047801200e0ef563dbb9a71c2", "request_number": "REQ0000001"}
```
CartJS - `updateItem(Map request, String cart_item_id)`

Updates an item in the cart.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request</td>
<td>Map</td>
<td>A JSON object that contains details of the catalog item to be updated.</td>
</tr>
<tr>
<td>cart_item_id</td>
<td>String</td>
<td>sys_id of the cart item to be modified.</td>
</tr>
</tbody>
</table>

The structure of the request object is:

```json
{
    'sysparm_quantity': item_quantity,
    'sysparm_requested_for': requested_for,
    'variables': {
        'var_name': 'var_value',
        ...
    }
}
```

- **item_quantity**: Number of items to be added. Default value is 1.
- **var_name**: Name of the question.
- **var_value**: Value of the answer (Not the display value).

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSON</td>
<td>Details of the cart.</td>
</tr>
</tbody>
</table>

```json
{
    'subtotal': value,
    'items': [
    {
        'item_name': '',
        'quantity': '',
        'price': '',
        'recurring_price': ''
    }
    ...
}
```
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

Example

```javascript
var cart = new sn_sc.CartJS();
var request = {
    'sysparm_quantity': '1',
    'variables':{
        'carrier': 'at_and_t_mobility',
        'data_plan': '500MB',
        'duration': 'eighteen_months',
        'color': 'slate',
        'storage': 'sixtyfour'
    }
};
var cart_item_id = "4d69b672c322320076173b0ac3d3ae79";
var cartDetails = cart.updateItem(request, cart_item_id);
gs.info(cartDetails);
```

Output

```javascript
{
    "cart_id":"35ec9e8947a13200e0ef563dbb9a7109",
    "items":[
        { "cart_item_id":"35ec9e8947a13200e0ef563dbb9a710a",
            "catalog_item_id":"0d0883723715300158bbfc8bcbe5d02", "item_name":"Apple iPhone 5",
            "localized_price":"$799.99", "localized_recurring_price":"$30.00", "price":"$799.99",
            "quantity":1, "recurring_frequency":"Monthly", "recurring_price": "$29.00" } ]
    "subtotal":"$799.99" }
```

**CartJS - canViewRF()**

Specifies if the current user has the required role to edit the **Request for** field.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates whether the user has the required role to edit the requested for field. Valid values:  
|         | • true: user has the required role                                           |
|         | • false: user does not have the required role                                |

This example checks if the user has the role needed to edit the requested for field.

```javascript
var cart=new sn_sc.CartJS();
console.log(cart.canViewRF());
```

Output:

```javascript
true
```

**CartJS - getCartDetails()**

Returns the cart details.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object pointing to the current cart details.</td>
</tr>
</tbody>
</table>

Example:

```javascript
var cart=new sn_sc.CartJS();
console.log (cart.getCartDetails());
```

Output:
CartJS - setRequestedFor(String user)
Sets the sys_id in the sys_user record of the user for whom the cart is requested.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>String</td>
<td>sys_id to be set in the sys_user record of the user for whom the cart is requested.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
var cart = new sn_sc.CartJS();
cart.setRequestFor("039c516237b1300054b6a3549dfe5dfe")
```
CatalogJS - Scoped

The CatalogJS API enables you to use methods to check and retrieve catalog-specific properties.

To use this class in a scoped application, use the sn_sc namespace identifier. The Service Catalog Scoped API plugin (com.glideapp.servicecatalogScoped.api) that is enabled by default is required to access the CatalogJS API.

CatalogJS - Catalog(GlideRecord now_GR)

Creates an instance of the catalog class for the specified glide record object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>Object</td>
<td>Glide Record pointing to the sc_catalog table.</td>
</tr>
</tbody>
</table>

This example shows how to create a new instance of the catalog class.

```javascript
var now_GR = new GlideRecord('sc_catalog');
now_GR.addQuery('sys_id','e0d08b13c3330100c8b837659bba8fb4');
now_GR.query();
var catalog = new sn_sc.Catalog(now_GR);
```

CatalogJS - Catalog(String sys_id)

Creates an instance of the Catalog class with the specified sys_id.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>Sys_id of the catalog.</td>
</tr>
</tbody>
</table>

This example shows how to create an instance of the Catalog class with the specified sys_id.

```javascript
new sn_sc.Catalog(catalog_sys_id);
var catalog = new sn_sc.Catalog("31bea3d53790200044e0bfc8bcbe5dec");
```

CatalogJS - canView(Boolean mobile, String userId)

Determines whether a user can view the current category on a mobile device or desktop.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mobile</td>
<td>Boolean</td>
<td>Flag that indicates whether to check if the user can view the current catalog on a mobile view or desktop view. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Mobile view</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Desktop view</td>
</tr>
<tr>
<td>userId</td>
<td>String</td>
<td>Optional. Sys_id of the user to check if they can view the catalog. Default: Current user</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the catalog is viewable by the user. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Catalog is viewable by the user.</td>
</tr>
<tr>
<td></td>
<td>• false: Catalog is not viewable by the user.</td>
</tr>
</tbody>
</table>

Example:

```javascript
var catalog = new sn_sc.Catalog("e0d08b13c3330100c8b837659bba8fb4");
gs.info(catalog.canView(true));
```

Output:

true

**CatalogJS - getAvailableCatalog()**

Returns the available active catalog.

If only one active catalog exists, then the method returns that catalog. Otherwise, it returns the earliest catalog created from the list of the catalogs that the user can view. If no catalog is available, the method returns null.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object pointing to the earliest catalog that the user can view, or null if no catalog is available.</td>
</tr>
</tbody>
</table>

This example returns the catalog that is available to the current user.

```javascript
var catalog = sn_sc.Catalog.getAvailableCatalog()
```

### CatalogJS - getBackgroundColor()

Returns the catalog background color.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Background color of the catalog.</td>
</tr>
</tbody>
</table>

This example returns the background color for the associated catalog.

```javascript
var catalog = new sn_sc.Catalog("0f910a2ac3112200b12d9f2974d3ae3c");
gs.info("Catalog background: "+ catalog.getBackgroundColor());
```

**Output:**

```
Catalog background: white
```

### CatalogJS - getCatalogCount()

Returns the number of catalogs active in the catalog table.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>Number of catalogs available in the catalog table.</td>
</tr>
</tbody>
</table>

**Example:**

```javascript
console.log(sn_sc.Catalog.getCatalogCount());
```

**Output:**

3

---

**CatalogJS - getCategories()**

Returns the categories for the current catalog.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArrayList</td>
<td>Returns the categories for the current catalog.</td>
</tr>
</tbody>
</table>

This example returns the categories available in the current catalog object.

```javascript
var catalog=new sn_sc.Catalog("e0d08b13c3330100c8b837659bba8fb4");
catalog.log(catalog.getCategories());
```

**Output:**

0

:  
  {header_image: "", sys_id: "e15706fc0a0a0aa7007fc2e1ab70c2f", description: "Your IT gateway. Report issues and submit requests.", title: "Can We Help You?");
CatalogJS - getCategoryIds()

Specifies the sys_ids of the categories in the current catalog.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArrayList</td>
<td>Returns the sys_ids of the categories in the current catalog.</td>
</tr>
</tbody>
</table>

This example returns the sys_ids of the categories available in the current catalog.

```javascript
var catalog=new sn_sc.Catalog("e0d08b13c3330100c8b837659bba8fb4");
console.log(catalog.getCategoryIds());
```
CatalogJS - getDescription()

Returns the description of the current catalog.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Catalog description.</td>
</tr>
</tbody>
</table>

This example returns the name of the current catalog.

```javascript
var catalog=new sn_sc.Catalog("e0d08b13c3330100c8b837659bba8fb4");
console.log(catalog.getDescription());
```

**Output:**

Service Catalog - IT Now

CatalogJS - getDesktopImageSRC()

Returns the catalog desktop image file name.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Catalog desktop image file name.</td>
</tr>
</tbody>
</table>

This example returns the current catalog desktop image file name.

```javascript
var catalog = new sn_sc.Catalog("e0d08b13c3330100c8b837659bba8fb4");
console.log(catalog.getDesktopImageSRC());
```

Output:

```
adbcc271475211002ee987e8dee49001.iix
```

**CatalogJS - getGr()**

Returns the current catalog's GlideRecord.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>GlideRecord of the current catalog.</td>
</tr>
</tbody>
</table>

This example returns the GlideRecord for the specified catalog.

```javascript
var catalog = new sn_sc.Catalog("e0d08b13c3330100c8b837659bba8fb4");
data.history = catalog.getGr();
```

**CatalogJS - getHeaderIconSRC()**

Returns the current catalog's header icon.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Catalog header icon.</td>
</tr>
</tbody>
</table>

This example shows how to obtain the name of the catalog header icon.

```javascript
var catalog = new sn_sc.Catalog("e0d08b13c3330100c8b837659bba8fb4");
gs.info(catalog.getHeaderIconSRC());
```

Output:

```
service_catalog_header.png
```

CatalogJS - `getId()`

Returns the sys_id of the current catalog.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This example returns the sys_id of the current catalog.

```javascript
var catalog = new sn_sc.Catalog("e0d08b13c3330100c8b837659bba8fb4");
console.log(catalog.getId());
```

Output:

```
e0d08b13c3330100c8b837659bba8fb4
```

CatalogJS - `getTitle()`

Returns the title of the current catalog.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Title of the current catalog.</td>
</tr>
</tbody>
</table>

This example returns the title of the current catalog.

```javascript
var catalog=new sn_sc.Catalog("e0d08b13c3300100c8b83759bba8fb4");
gs.info(catalog.getTitle());
```

**Output:**

```
Service Catalog
```

### CatalogJS - hasCategories()

Specifies if the current catalog has categories.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates whether the current catalog has categories. Valid values:  
- true: Catalog has categories.  
- false: Catalog does not have categories. |

This example shows how to determine if the current catalog contains categories.

```javascript
var catalog=new sn_sc.Catalog("e0d08b13c3300100c8b83759bba8fb4");
gs.info(catalog.hasCategories());
```
**CatalogJS - hasItems()**

Specifies if the current catalog has catalog items.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the current catalog has catalog items. Valid values: • true: Catalog has catalog items. • false: Catalog does not have catalog items.</td>
</tr>
</tbody>
</table>

**Example:**

```javascript
var catalog=new sn_sc.Catalog("e0d08b13c3330100c8b837659bba8fb4");
gs.info(catalog.hasItems());
```

**Output:**

```
true
```

**CatalogJS - isWishlistEnabled()**

Specifies if the wish list is enabled for the current catalog.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the current catalog supports wish lists.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Catalog supports wish lists.</td>
</tr>
<tr>
<td></td>
<td>• false: Catalog does not support wish lists.</td>
</tr>
</tbody>
</table>

This example shows how to check if a wish lists are enabled for the current catalog.

```javascript
var catalog = new sn_sc.Catalog("e0d08b13c3330100c8b837659bba8fb4");
gs.info(catalog.isWishlistEnabled());
```

Output:

```
true
```

CatalogSearch - Scoped

CatalogSearch API enables you to search catalog item.

To use this class in a scoped application, use the `sn_sc` namespace identifier. The Service Catalog Scoped API plugin (ID: com.glideapp.servicecatalogScoped.api) that is enabled by default is required to access the CatalogSearch API.

Scoped CatalogSearch - CatalogSearch()

Creates an instance of the CatalogSearch class.

```javascript
var catSearchGR = new sn_sc.CatalogSearch().search('', '', 'Apple', false, true);
catSearchGR.query();
while(catSearchGR.next()) {
    gs.info(catSearchGR.getValue('name'));
}
```
Scoped CatalogSearch - search (String catalogID, String categoryID, String term, Boolean mobile, Boolean noDepthSearch)

Searches a catalog item based on a search term. The search can be narrowed down to a catalog category level.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalogID</td>
<td>String</td>
<td>Identifier of the catalog to search.</td>
</tr>
<tr>
<td>categoryID</td>
<td>String</td>
<td>Identifier of the catalog category to search.</td>
</tr>
<tr>
<td>term</td>
<td>String</td>
<td>Search term.</td>
</tr>
<tr>
<td>mobile</td>
<td>Boolean</td>
<td>Flag that indicates whether catalog items exposed for mobile are searched.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Search for mobile catalog items.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not search for mobile catalog items.</td>
</tr>
<tr>
<td>noDepthSearch</td>
<td>Boolean</td>
<td>Flag that indicates whether to search subcategories.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Do not search subcategories.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Search subcategories.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>Returns the GlideRecord on sc_cat_item matching the search result.</td>
</tr>
</tbody>
</table>

This example shows how to search all catalogs and all categories for term: ServiceNow.

```javascript
var now_GR = new sn_sc.CatalogSearch().search('', '', 'ServiceNow', false, false);
now_GR.query();
while(now_GR.next()) {
    gs.log(now_GR.name);
}
```

**Output**
CatalogItemVariable - Scoped

CatalogItemVariable API enables you to create and modify service catalog item variables using scripts.

CatalogItemVariable - create(Boolean standardUpdate)

Inserts the specified catalog item variable.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Flag that indicates whether to enable the running of engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Enable engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not enable engines and workflow.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Return the sys_id of the inserted variable record.</td>
</tr>
</tbody>
</table>

CatalogItemVariable - deleteRecord(Boolean standardUpdate)

Deletes the defined catalog item variable.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Flag that indicates whether to enable the running of engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Enable engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not enable engines and workflow.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CatalogItemVariable - read(Object columns, Boolean standardUpdate)

Returns a mapping of catalog item variable attribute values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columns</td>
<td>Object</td>
<td>Set of columns for which to return values.</td>
</tr>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Flag that indicates whether to enable the running of engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Enable engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not enable engines and workflow.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object mapping column names to values.</td>
</tr>
</tbody>
</table>

CatalogItemVariable - setAttributes(Object attributes)

Defines attribute values for this catalog item variable.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Object</td>
<td>An object mapping column names to values.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
CatalogItemVariable - update(Object columnValues, Boolean standardUpdate)

Updates current catalog item variable with set values.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columnValues</td>
<td>Object</td>
<td>An object mapping column names to values.</td>
</tr>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Flag that indicates whether to enable the running of engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Enable engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not enable engines and workflow.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CatalogClientScript - Scoped

CatalogClientScript API enables you to create, modify, or delete catalog client script records.

To use this class in a scoped application, use the `sn_sc` namespace identifier. The Service Catalog Scoped API plugin (ID: `com.glideapp.servicecatalogScoped.api`) that is enabled by default is required to access the CatalogClientScript API.

Scoped CatalogClientScript - addScript(String script)

Adds a script to the catalog client script.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>script</td>
<td>String</td>
<td>Script to be added to the catalog client script.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
var catalogClientScript = new sn_sc.CatalogClientScript();
catalogClientScript.addScript("function onLoad(){Enter the script}");
```

### Scoped CatalogClientScript - appliesToCatalogItem(Boolean flag)
Specifies if the catalog client script runs on a catalog item.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flag</td>
<td>Boolean</td>
<td>If true, the catalog client script runs on the catalog item. If false, the catalog client script does not run on the catalog item.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
var catalogClientScript = new sn_sc.CatalogClientScript();
catalogClientScript.appliesToCatalogItem(true);
```

### Scoped CatalogClientScript - appliesToCatalogTask(Boolean flag)
Specifies if the catalog client script runs on a catalog task.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flag</td>
<td>Boolean</td>
<td>If true, the catalog client script runs on the catalog task. If false, the catalog client script does not run on the catalog task.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
var catalogClientScript = new sn_sc.CatalogClientScript();
catalogClientScript.appliesToCatalogTask(true);
```

**Scoped CatalogClientScript - appliesToRequestedItem(Boolean flag)**

Specifies if the catalog client script runs on a requested item.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flag</td>
<td>Boolean</td>
<td>If true, the catalog client script runs on the requested item. If false, the catalog client script does not run on the requested item.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
var catalogClientScript = new sn_sc.CatalogClientScript();
catalogClientScript.appliesToRequestedItem(true);
```

**Scoped CatalogClientScript - appliesToTargetRecord(Boolean flag)**

Specifies if the catalog client script runs on a requested item.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flag</td>
<td>Boolean</td>
<td>If true, the catalog client script runs on the target record. If false, the catalog client script does not run on the target record.</td>
</tr>
</tbody>
</table>
## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

## Example

```java
var catalogClientScript = new sn_sc.CatalogClientScript();
catalogClientScript.appliesToTargetRecord(true);
```

### Scoped CatalogClientScript - CatalogClientScript()

Creates an instance of the CatalogClientScript class.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```java
var catalogClientScript = new sn_sc.CatalogClientScript();
```

### Scoped CatalogClientScript - create(Boolean standardUpdate)

Inserts the defined catalog client script in the catalog_script_client table.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td></td>
</tr>
</tbody>
</table>

Flag that indicates whether to enable the running of engines and workflow. Valid values:

- true: Enable engines and workflow.
- false: Do not enable engines and workflow.

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the catalog client script.</td>
</tr>
</tbody>
</table>

### Example
var catalogClientScript = new sn_sc.CatalogClientScript();
catalogClientScript.setAttributes({"name": "My Catalog Item", "applies_to": "item",  
"ui_type": "desktop", "type": "onLoad"});
catalogClientScript.appliesToCatalogItem(true);
catalogClientScript.appliesToRequestedItem(true);
catalogClientScript.appliesToCatalogTask(true);
catalogClientScript.appliesToTargetRecord(true);
var catalogClientScriptId = catalogClientScript.create();
gs.info(catalogClientScriptId);

Output

039c516237b1300054b6a3549dbe5dfc

Scoped CatalogClientScript - deleteRecord(String sys_id, Boolean standardUpdate)

Deletes the defined catalog client script.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>Sys_id of the catalog client script.</td>
</tr>
</tbody>
</table>
| standardUpdate | Boolean  | Flag that indicates whether to enable the running of engines and workflow. Valid values:  
- true: Enable engines and workflow.  
- false: Do not enable engines and workflow. |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to delete a catalog client script.

var sys_id = "039c516237b1300054b6a3549dbe5dfc";
var catalogClientScript = new sn_sc.CatalogClientScript();
catalogClientScript.deleteRecord("039c516237b1300054b6a3549dbe5dfc");
Scoped CatalogClientScript - setAttributes(Map attributes)
Defines attribute values for the catalog client script.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Map</td>
<td>A JSON object that has mapping for the field and value pairs.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
var catalogClientScript = new sn_sc.CatalogClientScript();
catalogClientScript.setAttributes({"name": "My Catalog Item", "applies_to": "catalog_item", "ui_type": "desktop", "type": "onLoad"});
```

Scoped CatalogClientScript - setCatalogItem(String sys_id)
Associates a catalog item with the catalog client script.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>Sys_id of the catalog item.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to associate a catalog item with the current catalog client script.

```javascript
var catalogClientScript = new sn_sc.CatalogClientScript();
catalogClientScript.setCatalogItem("039c516237b1300054b6a3549dbe5dfc");
```
Scoped CatalogClientScript - setOnChangeVariable(String sys_id)
Runs the catalog client script when a variable value is updated.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>sys_id</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

Example
```
var catalogClientScript = new sn_sc.CatalogClientScript();
catalogClientScript.setOnChangeVariable("039c516237b1300054b6a3549dbe5dfc");
```

Scoped CatalogClientScript - setVariableSet(String sys_id)
Associates a variable set with the catalog client script.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>sys_id</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

Example
```
var catalogClientScript = new sn_sc.CatalogClientScript();
catalogClientScript.setVariableSet("039c516237b1300054b6a3549dbe5dfc");
```

CatalogItemVariableSet - Scoped

CatalogItemVariableSet API enables you to create and modify service catalog item variable sets using scripts.
CatalogItemVariableSet - create(Boolean standardUpdate)

Inserts the defined catalog item variable set.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Flag indicating whether to enable the running of engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Enable the running of engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not enable the running of engines and workflow.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the inserted variable record.</td>
</tr>
</tbody>
</table>

// Given an existing catalog item
var catItemSysId = "e0d08b13c3330100c8b837659bba8fb4";
addVariableSets(catItemSysId);

function addVariableSets(catItemSysId) {
    // List of all variable sets to attach
    var myVarSets = [];

    // Create variable set
    var myVarSetAttrs = {"name": "Requester details", "order": "100"};
    var myVarSet = new sn_sc.CatalogItemVariableSet();
    myVarSet.setAttributes(myVarSetAttrs);
    var myVarSetId = myVarSet.create(true);
    myVarSets.push(myVarSetId);
}

CatalogItemVariableSet - deleteRecord(Boolean standardUpdate)

Deletes the defined catalog item variable.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Flag that indicates whether to enable the running of engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Enable engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not enable engines and workflow.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**CatalogItemVariableSet - read(Object columns, Boolean standardUpdate)**

Returns a mapping of catalog item variable set attribute values.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columns</td>
<td>Object</td>
<td>Specify the set of columns that you would like the values for.</td>
</tr>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Flag that indicates whether to enable the running of engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Enable engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not enable engines and workflow.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object mapping column names to values.</td>
</tr>
</tbody>
</table>

**CatalogItemVariableSet - setAttributes(Object attributes)**

Defines attribute values for this catalog item variable set.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Object</td>
<td>An object mapping column names to values.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CatalogItemVariableSet - update(Object columnValues, Boolean standardUpdate)

Updates current catalog item variable set with set values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columnValues</td>
<td>Object</td>
<td>An object mapping column names to values.</td>
</tr>
</tbody>
</table>
| standardUpdate | Boolean    | Flag that indicates whether to enable the running of engines and workflow. Valid values:  
• true: Enable engines and workflow.  
• false: Do not enable engines and workflow. |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CatalogItemVariableSetM2M - Scoped

CatalogItemVariableSetM2M API enables you to create and modify service catalog item variable set many-to-many (M2Ms) using scripts.

CatalogItemVariableSetM2M - create(Boolean standardUpdate)

Inserts the defined catalog item variable set M2M.
### CatalogItemVariableSetM2M - deleteRecord(Boolean standardUpdate)

Deletes the defined catalog item variable set M2M.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| standardUpdate| Boolean | Flag that indicates whether to enable the running of engines and workflow. Valid values:  
• true: Enable engines and workflow.  
• false: Do not enable engines and workflow. |

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the inserted variable record.</td>
</tr>
</tbody>
</table>

### CatalogItemVariableSetM2M - read(Object columns, Boolean standardUpdate)

Returns a mapping of catalog item variable set M2M attribute values.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columns</td>
<td>Object</td>
<td>Set of columns that you would like the values for.</td>
</tr>
</tbody>
</table>
| standardUpdate| Boolean    | Flag that indicates whether to enable the running of engines and workflow. Valid values:  
  • true: Enable engines and workflow.  
  • false: Do not enable engines and workflow. |

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object mapping column names to values.</td>
</tr>
</tbody>
</table>

### CatalogItemVariableSetM2M - setAttributes(Object attributes)

Defines attribute values for this catalog item variable set M2M.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Object</td>
<td>An object mapping column names to values.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### CatalogItemVariableSetM2M - update(Object columnValues, Boolean standardUpdate)

Updates current catalog item variable set M2M with set values.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columnValues</td>
<td>Object</td>
<td>An object mapping column names to values.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| standardUpdate  | Boolean  | Flag that indicates whether to enable the running of engines and workflow.  
|                 |          | Valid values:  
|                 |          | • true: Enable engines and workflow.  
|                 |          | • false: Do not enable engines and workflow.                                                                                           |

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### CatCategory - Scoped

`CatCategory` API enables you to create and modify service catalog categories using scripts.

### CatCategory - availableForUserCriteria(String action, Array criteriaIDs)

Adds the **Available For** user criteria to the current catalog category.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| action   | String | Action to perform.  
|          |        | • add: Adds the user criteria to the **Available For** list.  
|          |        | • delete: Deletes the user criteria from the **Available For** list.       |
| criteriaIDs | Array | Array of the user criteria sys_ids.                                        |

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var item = new sn_sc.CatCategory("31bea3d53790200044e0bfc8bcbe5dec");
item.availableForUserCriteria("add", ["0c441abbc6112275000025157c651c89"]);  
```
CatCategory - canView(Boolean isMobile, String userSysId)
Determines whether a specified user can view a specified category on a mobile device or desktop.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isMobile</td>
<td>Boolean</td>
<td>Flag that indicates whether to verify the user for access on a mobile device or desktop.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Validate for mobile.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Validate for desktop.</td>
</tr>
<tr>
<td>userSysId</td>
<td>String</td>
<td>Sys_id of the user to validate.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the user can view the associated category.</td>
</tr>
<tr>
<td></td>
<td>• true: User can view the category.</td>
</tr>
<tr>
<td></td>
<td>• false: User cannot view the category.</td>
</tr>
</tbody>
</table>

```javascript
var scopedCategoryObj = new sn_sc.CatCategory('7b1262b9530033007444ddeeff7b12ae');
scopedCategoryObj.canView(false, '62826bf03710200044e0bfc8bcbe5df1');
```

CatCategory - create(Boolean standardUpdate)
Insert the defined category.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Flag that indicates whether to enable the running of engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Enable engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not enable engines and workflow.</td>
</tr>
</tbody>
</table>
## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the inserted variable record.</td>
</tr>
</tbody>
</table>

```javascript
var categoryCreate = new sn_sc.CatCategory();
categoryCreate.setAttributes({"title": "test a scoped category", "sc_catalog": "e0d08b13c3330100c8b837659bba8fb4"});
var categorySysId = categoryCreate.create();
var isValidSysId = categorySysId.match(/^[0-9a-fA-F]{32}$/) === null ? false : true;
global.Assert.assertEquals(true, isValidSysId,"CategorySysId: [" + categorySysId + "] is not valid", true, isValidSysId);
```

### CatCategory - deleteRecord(Boolean standardUpdate)

Deletes the category record on which the CatCategory class was initially instantiated.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Flag that indicates whether to enable the running of engines and workflow. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Enable engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not enable engines and workflow.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var categoryDelete = new sn_sc.CatCategory(categorySysId);
categoryDelete.deleteRecord();
var category = new sn_sc.CatCategory(categorySysId);
values = category.read({"title": "", "sc_catalog":""}, false);
global.Assert.assertEquals("", values.title,"Category should title");
```
CatCategory - `notAvailableForUserCriteria(String action, Array[] criteriaIDs)`

Adds the **Not Available For** user criteria to a catalog category.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>String</td>
<td>Action to perform.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>add</strong>: Adds the user criteria to the <strong>Not Available For</strong> list.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>delete</strong>: Deletes the user criteria from the <strong>Not Available For</strong> list.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var item = new sn_sc.CatCategory("31be3d53790200044e0bfc8bcbe5dec");
item.notAvailableForUserCriteria("add", ["0c441abbc6112275000025157c61c89"]);
```

CatCategory - `read(Object columns, Boolean standardUpdate)`

Returns a mapping of the category.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columns</td>
<td>Object</td>
<td>Set of columns that you would like the values for.</td>
</tr>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Flag that indicates whether to enable the running of engines and workflow. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>true</strong>: Enable engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>false</strong>: Do not enable engines and workflow.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Mapping of column names to values.</td>
</tr>
</tbody>
</table>
```javascript
var category = new sn_sc.CatCategory("a96277509f300200b407b89a442e704e");
var values = category.read({"title": ""}, true);
gs.log(values.title);
```

**CatCategory - setAttributes(Object attributes)**

Defines the attribute values for this category.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Object</td>
<td>Attributes for the new field and value pairs.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var categoryCreate = new sn_sc.CatCategory();
categoryCreate.setAttributes({"title": "test a scoped category", "sc_catalog": "e0d08b13c3330100c8b837659bba8fb4"});
var categorySysId = categoryCreate.create();
var isValidSysId = categorySysId.match(/^[0-9a-fA-F]{32}$/) == null ? false : true;
global.Assert.assertEquals(true, isValidSysId,"CategorySysId: [" + categorySysId + "] is not valid", true, isValidSysId);
```

**CatCategory - setTableName(String tableName)**

Defines the table name for this category.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table that extends sc_category.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
```javascript
var categoryRead = new sn_sc.CatCategory(categorySysId);
categoryRead.setTableName("test_category");
var values = categoryRead.read({"title": "," , "sc_catalog": "," , "test": ","}, false);
gs.info(categorySysId);
gs.info(values.title);
gs.info(values.test);
global.Assert.assertEquals("testValue", values.test,"Category extends sc_category and stores its extended value");
```

**CatCategory - update(Object columnValues, Boolean standardUpdate)**

Updates the current category with the specified name-value pairs.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columnValues</td>
<td>Object</td>
<td>Mapping of column names to values.</td>
</tr>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Flag that indicates whether to enable the running of engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Enable engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not enable engines and workflow.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var categoryUpdate = new sn_sc.CatCategory(categorySysId);
categoryUpdate.update({"title": "test changed scoped category"}, true);
values = categoryUpdate.read({"title": "," , "sc_catalog": ","}, false);
global.Assert.assertEquals("test changed scoped category", values.title,"Category should title");
```

**CatCategory - getID()**

Returns the sys_id of the current category.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the current category.</td>
</tr>
</tbody>
</table>

Example:

```javascript
var cart = new sn_sc.CatCategory("2809952237b1300054b6a3549dbe5dd4");
var categoryID = cart.getID();
gs.info(categoryID);
```

Output:

```
2809952237b1300054b6a3549dbe5dd4
```

CatItem - Scoped

CatItem API enables you to create and modify service catalog items using scripts.

CatItem - availableForUserCriteria(String action, Array criteriaIDs)

Adds the Available For user criteria to the current catalog item.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>String</td>
<td>Action to perform.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• add: Adds the user criteria to the Available For list.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• delete: Deletes the user criteria from the Available For list.</td>
</tr>
<tr>
<td>criteriaIDs</td>
<td>Array</td>
<td>Array of the user criteria sys_ids.</td>
</tr>
</tbody>
</table>
This example shows how to add the specified **Available For** user criteria.

```javascript
var item = new sn_sc.CatItem("31bea3d53790200044e0bfc8bcbe5dec");
item.availableForUserCriteria("add", ["0c441abbc6112275000025157c651c89"]);
```

**CatItem - canViewOnSearch(boolean isMobile)**

Determines if the user has access to view the catalog item on global search.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isMobile</td>
<td>Boolean</td>
<td>Flag that indicates whether to perform the search for the mobile or desktop view.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Perform the search for the mobile view.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Perform the search for the desktop view.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the user has access to view the catalog item on global search.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: User has access to view the catalog item on global search.</td>
</tr>
<tr>
<td></td>
<td>• false: User does not have access to view the catalog item on global search.</td>
</tr>
</tbody>
</table>

This code example shows how to check if the user has access to view the catalog item on global search in desktop view.

```javascript
var catItem = new sn_sc.CatItem("04b7e94b4f7b4200086eeed18110c7fd");
var canView = catItem.canViewOnSearch('false');
gs.info("Can view on global search: " + canView);
```
Output:

| Can view on global search: true |

**CatItem - canViewInDomain()**

Verifies whether the current catalog item is viewable in the selected domain (domain selected in the domain picker).

Catalog items in the global domain are available across all domains.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that validates whether the current catalog item is viewable in the selected domain. Valid values:  
true: Catalog item is viewable in the domain  
false: Catalog item is not viewable in the domain |

This example shows how to verify whether a catalog item is viewable in the currently selected domain.

```javascript
var catItem = new sn_sc.CatItem("060f3afa3731300054b6a3549dbe5d3e");
gs.info(catItem.canViewInDomain());
```

**CatItem - create(Boolean standardUpdate)**

Inserts the defined catalog item.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Flag that indicates whether to enable the running of engines and workflow. Valid values:</td>
</tr>
</tbody>
</table>
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| standardUpdate   | Boolean | Flag that indicates whether to enable the running of engines and workflow. Valid values:  
  • true: Enable the running of engines and workflow.  
  • false: Do not enable the running of engines and workflow.  

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String | Sys_id of the newly created catalog item.        

This code example shows how to create a new catalog item.

```javascript
var catalogItem = new sn_sc.CatItem();
catalogItem.setAttributes({"name":"My FirstCatItem");
catalogItem.setCatalogs(catalogSysId); // Service catalog
catalogItem.setCategories(categorySysId); // Use scoped category

var catItemSysId = catalogItem.create(true); // Returns sys_id of created item
```

**CatItem - deleteRecord(Boolean standardUpdate)**

Deletes the defined catalog item.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| standardUpdate| Boolean | Flag that indicates whether to enable the running of engines and workflow. Valid values:  
  • true: Enable the running of engines and workflow.  
  • false: Do not enable the running of engines and workflow.  

This code example shows how to delete a catalog item.

```javascript
var catItem = new sn_sc.CatItem("a96277509f300200b407b89a442e704e");
catItem.deleteRecord(false);
```

**CatItem - getFirstAccessibleCategoryForSearch(String catalogId)**

Returns the first category that the user can view in a catalog.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalogId</td>
<td>String</td>
<td>Sys_id of the catalog.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the first category that the user can view in a catalog.</td>
</tr>
</tbody>
</table>

**Example:**

```javascript
var CatItem=new sn_sc.CatItem("04b7e94b4f7b42000866ed18110c7fd");
console.log(CatItem.getFirstAccessibleCategoryForSearch("e0d08b13c3330100c8b837659bba8fb4"));
```

**Output:**

d258b953c611227a0146101fb1be7c31

**CatItem - getInvalidDelegatedUsers(Array requestForUsers)**

Returns an array of users for whom the associated item cannot be delegated (requested on behalf of).

The method verifies each of the users passed in the array.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestForUsers</td>
<td>Object</td>
<td>Array of user sys_ids to check whether the associated user can acquire the current item and that the item can be requested on behalf of them. User sys_ids are located in the Users [sys_user] table.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of user names (Name column from Users [sys_user] table) for whom the item cannot be requested for by a delegate.</td>
</tr>
</tbody>
</table>

This example shows how to obtain a list of user names for whom the item cannot be requested for by a delegate.

```javascript
function getInvalidDelegatedUsers(itemId, userIds) {
    var catItem = new sn_sc.CatItem(itemId);
    var invalidUsers = catItem.getInvalidDelegatedUsers(userIds);
    return invalidUsers;
}
```

**Output:**

```
[
    "Joe Smith",
    "Jenny Brown",
    "Fred Bennet",
    "Alice Jones"
]
```

---

**CatItem - getRecordClass()**

Returns the class name for the current catalog item record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Class name for the current catalog item record.</td>
</tr>
</tbody>
</table>

Example:

```javascript
var CatItem = new sn_sc.CatItem("04b7e94b4f7b4200086eeed18110c7fd");
console.log(CatItem.getRecordClass());
```

Output:

```javascript
sc_cat_item
```

**CatItem - isDelegationAllowed(String delegatedUser)**

Verifies whether the specified delegated user has acquisition rights to the current service catalog item.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>delegatedUser</td>
<td>String</td>
<td>Optional. Sys_id of the user to request the service catalog item for (delegate). The method verifies whether the user has acquisition rights to the item. Default: Checks whether the calling user has acquisition rights to the item.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates whether the user has acquisition rights to the current service catalog item. Valid values:  
  • true: User has acquisition rights to the item.  
  • false: User does not have acquisition rights to the item. |

This code example shows how to determine if delegation is allowed for the catalog item.

```javascript
function canRequestFor(itemId, user) {
    var catItem = new sn_sc.CatItem(itemId);
    var delegatedUser = user;  
    var _result = catItem.isDelegationAllowed(delegatedUser);
    return result;
}
```
var result = catItem.isDelegationAllowed(user);
return result;
}

Output: true

CatItem - isVisibleServicePortal()
Determines if the current catalog item is available in service portal.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the catalog item is available in the Service Portal. Valid values: • true: Available on Service Portal. • false: Not available on Service Portal.</td>
</tr>
</tbody>
</table>

Example:

var catItem = new sn_sc.CatItem("04b7e94b4f7b4200086e6ed18110c7fd");
var catItemAvail = catItem.isVisibleServicePortal();
gs.info("Is item available on Service Portal: " + catItemAvail);

Output:

Is item available on Service Portal: true

CatItem - notAvailableForUserCriteria(String action, Array criteriaIDs)
Adds the Not Available For user criteria to a catalog item.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>String</td>
<td>Action to perform.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>add</td>
<td></td>
<td>Adds the user criteria to the <strong>Not Available For</strong> list.</td>
</tr>
<tr>
<td>delete</td>
<td></td>
<td>Deletes the user criteria from the <strong>Not Available For</strong> list.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to add the specified **Not Available For** user criteria.

```javascript
var item = new sn_sc.CatItem("31bea3d53790200044e0bfc8bcbe5dec");
item.notAvailableForUserCriteria("add", ["0c441abbc6112275000025157c651c89"]);```

**CatItem - read(Object columns, Boolean standardUpdate)**

Returns a mapping of catalog item attribute values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| columns          | Object  | Name-value pairs of columns for which to return values \.
| standardUpdate   | Boolean | Flag that indicates whether to enable the running of engines and workflow.  |
|                  |         | Valid values:                                                               |
|                  |         | • true: Enable the running of engines and workflow.                          |
|                  |         | • false: Do not enable the running of engines and workflow.                  |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Mapping column names to values.</td>
</tr>
</tbody>
</table>

This code example shows how to read a catalog item.
```javascript
var catItem = new sn_sc.CatItem("a96277509f300200b407b89a442e704e");
var values = catItem.read({"name" : "," , "price" : ","}, true);
gs.info("Catalog item name: " + values.name);
gs.info("Catalog item price: " + values.price)
```

**Output:**

Catalog item name: Standard Laptop  
Catalog item price: 1100

---

**CatItem - setAttributes(Object attributes)**  
Defines attribute values for this catalog item.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Object</td>
<td>Name-value pairs of the columns for which to set.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This code example shows how to set the attribute for a new price field.

```javascript
var catItem = new sn_sc.CatItem();
catItem.setAttribute({"price2" : "12.50"});
```

---

**CatItem - setCatalogs(String catalogs)**  
Defines the catalogs that this catalog item is associated with.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalogs</td>
<td>String</td>
<td>Comma-separated list of sys_ids of the catalogs to associate with the item the item.</td>
</tr>
</tbody>
</table>
This example shows how to associate two catalogs to an item.

```javascript
var catItem = new sn_sc.CatItem("060f3afa3731300054b6a3549dbe5d3e");
catItem.setCatalogs("af443cfa5f130100a9ad2572f2b47747", 
"d467125fd7500200d74460affd6103a1");
```

CatItem - setCategories(String categories)

Defines the categories that this catalog item is associated with.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>categories</td>
<td>String</td>
<td>Comma-separated list of sys_ids of the categories to associate with the item the item.</td>
</tr>
</tbody>
</table>

This example shows how to associate two categories to an item.

```javascript
var catItem = new sn_sc.CatItem("060f3afa3731300054b6a3549dbe5d3e");
catItem.setCatagories("af443cfa5f130100a9ad2572f2b47747", 
"d467125fd7500200d74460affd6103a1");
```

CatItem - setImage(String dbImageSysId, String type)

Sets the image of a catalog item to the specified database image record.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbImageSysId</td>
<td>String</td>
<td>Sys_id of the database image.</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>Type of image.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• picture</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• icon</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to associate an image to a catalog item.

```javascript
var catItem = new sn_sc.CatItem("060f3afa3731300054b6a3549dbe5d3e");
catItem.setImage("d467125fd7500200d74460affd6103a1", "picture");
```

CatItem - setTableName(String tableName)

Defines the table name for this catalog item.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table that extends Catalog Item [sc_cat_item].</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to set the name of an extended table.

```javascript
var catItem = new sn_sc.CatItem();
catItem.setTableName("New_catalog_table");
```

CatItem - update(Object columnValues, Boolean standardUpdate)

Updates the current catalog item fields with a specified set of values.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columnValues</td>
<td>Object</td>
<td>Name-value pairs of the fields to update and their associated values.</td>
</tr>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Flag that indicates whether to enable the running of engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Enable the running of engines and workflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not enable the running of engines and workflow.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to update the price of an existing catalog item.

```javascript
var catItem = new sn_sc.CatItem("04b7e94b4f7b4200086eeed18110c7fd");
var values = catItem.read({"name" : ", "price" : ", false);
gs.info("Catalog item name: " + values.name);
gs.info("Catalog item price: " + values.price)
catItem.update({"price" : "1100"}, false);
var values = catItem.read({"price" : ", false);
gs.info("Catalog item price: " + values.price)
```

Output:

- Catalog item name: Standard Laptop
- Catalog item price: 1400
- Catalog item price: 1100

### Cell - Scoped, Global

Creates a Cell object as a cell in a table. You can use this API to format the cell and include additional blocks, such as paragraphs and images.

This API is part of the ServiceNow PDF Generation Utilities plugin (com.snc.apppdfgenerator) and is provided within the sn_pdfgeneratorutils namespace. The plugin is activated by default.
This API is a component used with the Document API to generate a PDF.

**Cell - Cell(Number rowspan, Number colspan)**

Instantiates a new `Cell` object. Creates a cell which takes a custom amount of cell spaces in the table.

ℹ️ **Note:** If you provide a negative number for either parameter, the value sets to 1.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rowspan</td>
<td>Number</td>
<td>Number of rows this cell is to occupy.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: 0</td>
</tr>
<tr>
<td>colspan</td>
<td>Number</td>
<td>Number of columns this cell is to occupy.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: 0</td>
</tr>
</tbody>
</table>

The following example shows how to create a `Cell` object spanning a single row and single column.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 1);
```

**Cell – addImage(Image image)**

Adds an image to a table cell.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>image</td>
<td><code>Image</code></td>
<td>Image to add to a table cell.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to instantiate an existing image attachment and add it to a cell in a table. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 1);
```
Cell – addParagraph(Paragraph paragraph)

Adds text to a table cell.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paragraph</td>
<td>Paragraph</td>
<td>Text to add to a table cell.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to instantiate paragraph objects and add the content cells in a table. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var para = new sn_pdfgeneratorutils.Paragraph("Paragraph text.");
cell.addParagraph(para);
```

Cell – addStyle(Style style)

Applies a predefined style to table cells.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>style</td>
<td>Style</td>
<td>Style to apply to this element.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to apply a style to a table cell. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var style = new sn_pdfgeneratorutils.Style("My Style");
cell.addStyle(style);
```
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var style = new sn_pdfgeneratorutils.Style();
style.setBold();

```
cell.addStyle(style);
```

**Cell – addTable(Table table)**

Adds a table to a cell.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>Table</td>
<td>Table to add to a cell.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to instantiate a table object and add it to a cell. For a document usage example, see Document API.

```
var cell = new sn_pdfgeneratorutils.Cell(1, 1);

var columnWidths = [100, 50, 50];

var table = new Table(columnWidths, false);

var cell = new sn_pdfgeneratorutils.Cell(1, 3);

var style = new sn_pdfgeneratorutils.Style();

```

```
cell.addStyle(style);
```

```
cell.addTable(table);
```

**Cell – getColumn()**

Gets the number of the column in which the cell is located.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of the column position for a cell.</td>
</tr>
</tbody>
</table>

The following example shows how to get the number of a column position for a cell.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 1);
var columnWidths = [100, 50, 50];
var table = new sn_pdfgeneratorutils.Table(true, columnWidths, false);
table.addCell(cell);
var colNum = cell.getColumn();
gs.info("The column position for cell is ", colNum);
```

Output:

The column position for cell is 1

**Cell – getRow()**

Gets the number of rows in which the cell is located.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of the row position for a cell.</td>
</tr>
</tbody>
</table>

The following example shows how to get the number of a row position for a cell.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 1);
var rowWidths = [100, 50, 50];
```
var table = new sn_pdfgeneratorutils.Table(true,rowWidths,false);

table.addCell(cell);

var rowNum = cell.getRow();

gs.info("The row position for cell is " + rowNum);

Output:

The row position for cell is 4927

**Cell – setBackGroundColor(Color color)**

Specifies a background color for the cell.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>Color</td>
<td>Background color.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set cell background color. For a document usage example, see Document API.

```
var cell = new sn_pdfgeneratorutils.Cell(1, 3);

var color = new sn_pdfgeneratorutils.Color([1, 0.9, 0.9]); // provided as array of RGB float values

cell.setBackGroundColor(color);
```

**Cell – setBorder(Number width)**

Sets a border for all four edges of a cell.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Cell border width in points.</td>
</tr>
</tbody>
</table>
The following example shows how to set a cell border of one point to all four edges of a cell. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var width = 1.0;
cell.setBorder(width);
```

**Cell – setBorderBottom(Number width)**

Sets a border for the lower limit of a cell.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Cell border width in points.</td>
</tr>
</tbody>
</table>

The following example shows how to set a cell border. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var width = 1.0;
cell.setBorder(width);
```

**Cell – setBorderLeft(Number width)**

Sets a border for the left limit of a cell.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Cell border width in points.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a left-side cell border. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var width = 1.0;
cell.setLeftBorder(width);
```

### Cell – `setBorderRight(Number width)`

Sets a border for the right limit of a cell.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Cell border width in points.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a right-side cell border. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var width = 1.0;
cell.setRightBorder(width);
```
Cell – setBorderTop(Number width)
Sets a border for the upper limit of a cell.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Cell border width in points.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a top cell border. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var width = 1.0;
cell.setBorderTop(width);
```

Cell – setColoredBorder(Color color, Number width)
Sets a colored border for all four edges of a cell.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>Color</td>
<td>Cell border color.</td>
</tr>
<tr>
<td>width</td>
<td>Number</td>
<td>Cell border width in points.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a colored cell border. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var width = 1.0;
var color = 'blue';
cell.setColoredBorder(color, width);
```
var cell = new sn_pdfgeneratorutils.Cell(1, 3);

var borderColor = new sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
var width = 1.0;

cell.setColoredBorderBottom(borderColor, width);

**Cell – setColoredBorderBottom(Color color, Number width)**

Sets a colored border for the lower limit of a cell.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>Color</td>
<td>Cell border color.</td>
</tr>
<tr>
<td>width</td>
<td>Number</td>
<td>Cell border width in points.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a colored lower cell border. For a document usage example, see Document API.

var cell = new sn_pdfgeneratorutils.Cell(1, 3);

var borderColor = new sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
var width = 1.0;

cell.setColoredBorderBottom(borderColor, width);

**Cell – setColoredBorderLeft(Color color, Number width)**

Sets a colored border for the left limit of a cell.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>Color</td>
<td>Cell border color.</td>
</tr>
<tr>
<td>width</td>
<td>Number</td>
<td>Cell border width in points.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a colored left cell border. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var borderColor = new sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
var width = 1.0;

cell.setColoredBorderLeft(borderColor, width);
```

**Cell – setColoredBorderLeft(Color color, Number width)**

Sets a colored border for the left limit of a cell.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>Color</td>
<td>Cell border color.</td>
</tr>
<tr>
<td>width</td>
<td>Number</td>
<td>Cell border width in points.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a colored right cell border. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var borderColor = new sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
var width = 1.0;

cell.setColoredBorderRight(borderColor, width);
```

**Cell – setColoredBorderRight(Color color, Number width)**

Sets a colored border for the right limit of a cell.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var borderColor = new sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
var width = 1.0;

cell.setColoredBorderTop(borderColor, width);
```

**Cell – setColoredBorderTop(Color color, Number width)**

Sets a colored border for the upper limit of a cell.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>Color</td>
<td>Cell border color.</td>
</tr>
<tr>
<td>width</td>
<td>Number</td>
<td>Cell border width in points.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a colored top cell border. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var borderColor = new sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
var width = 1.0;
cell.setColoredBorderTop(borderColor, width);
```

**Cell – setHeight(Number height)**

Sets the height of a cell.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Number</td>
<td>Cell height in points.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how set a cell height of 10 points. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var height = 10;
```
Cell – setHorizontalAlignment(String alignment)

Sets the horizontal alignment for this cell.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alignment</td>
<td>String</td>
<td>Horizontal alignment setting. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• center: Align contents to the center.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• left: Align contents to the left.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• right: Align contents to the right.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set horizontal alignment on a cell. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var alignment = "center";
cell.setHorizontalAlignment(alignment);
```

Cell – setMaxHeight(Number value)

Sets the maximum height of a cell.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Number</td>
<td>Maximum cell height in points.</td>
</tr>
</tbody>
</table>

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var alignment = "center";
cell.setHorizontalAlignment(alignment);
```
The following example shows how to set a maximum cell height of 409 points. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var height = 409;
cell.setMaxHeight(height);
```

**Cell – setMaxWidth(float value)**
Sets the maximum width of a cell.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Number</td>
<td>Maximum cell width in points.</td>
</tr>
</tbody>
</table>

The following example shows how to set a maximum cell width of 1530 points. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var width = 1530;
cell.setMaxWidth(width);
```

**Cell – setMinHeight(Number value)**
Sets the minimum height of a cell.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Number</td>
<td>Minimum cell height in points.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how set a minimum cell height of 12.75 points. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var height = 12.75;
cell.setMinHeight(height);
```

Cell – setMinWidth(Number value)

Sets the minimum width of a cell.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Number</td>
<td>Minimum cell width in points.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how set a minimum cell width of 50.58 points. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var width = 50.58;
cell.setMinWidth(width);
```
Cell – setOpacity(Number opacity)

Sets the opacity of cell content, borders, and background.

⚠️ Note: This setting affects all child elements of the cell.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>opacity</td>
<td>Number</td>
<td>Float decimal value from 0 through 1, in which 0 is transparent and 1 is fully opaque. Default: 0</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the opacity of a cell.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var opacity = 0.5;
cell.setOpacity(opacity);
```

Cell – setPadding(Number padding)

Sets the padding of all four sides of a cell to the same width.

See also:

- setPaddingBottom()
- setPaddingLeft()
- setPaddingRight()
- setPaddingTop()

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>padding</td>
<td>Number</td>
<td>Padding width in points as a decimal value.</td>
</tr>
</tbody>
</table>
The following example shows how to set the bottom cell to one point. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var padding = 1.0;
cell.setPaddingBottom(padding);
```

**Cell – setPaddingBottom(Number padding)**

Sets the value of the bottom padding width of a cell.

See also:
- `setPadding()`
- `setPaddingLeft()`
- `setPaddingRight()`
- `setPaddingTop()`

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>padding</td>
<td>Number</td>
<td>Padding width in points as a decimal value.</td>
</tr>
</tbody>
</table>

The following example shows how to set bottom cell padding to one point. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var padding = 1.0;
```
cell.setPaddingBottom(padding);

**Cell – setPaddingLeft(Number padding)**

Sets the value of the left padding width of a cell.

See also:
- setPadding()
- setPaddingBottom()
- setPaddingRight()
- setPaddingTop()

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>padding</td>
<td>Number</td>
<td>Padding width in points as a decimal value.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set left cell padding to one point. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var padding = 1.0;
cell.setPaddingLeft(padding);
```

**Cell – setPaddingRight(Number padding)**

Sets the value of the right padding width of a cell.

See also:
- setPadding()
- setPaddingBottom()
- setPaddingLeft()
- setPaddingTop()
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>padding</td>
<td>Number</td>
<td>Padding width in points as a decimal value.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set right cell padding to one point. For a document usage example, see [Document API](#).

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var padding = 1.0;
cell.setPaddingRight(padding);
```

**Cell – setPaddingTop(Number padding)**

Sets the value of the top padding width of a cell.

See also:
- `setPadding()`
- `setPaddingBottom()`
- `setPaddingLeft()`
- `setPaddingRight()`

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>padding</td>
<td>Number</td>
<td>Padding width in points as a decimal value.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
The following example shows how to set top cell padding to one point. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var padding = 1.0;

cell.setPaddingTop(padding);
```

**Cell – setTextAlignment(String alignment)**

Sets the text alignment of this cell.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| alignment | String | Text alignment position. Valid values:
|           |        | • text-center: Aligns text to the center.
|           |        | • text-justified: Modifies the space between characters to completely fill text between the left and right sides. The final line is left-aligned.
|           |        | • text-justified-all: Justifies text alignment including the final line.
|           |        | • text-left: Align text to the left.
|           |        | • text-right: Align text to the right. |

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the cell text to left alignment.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var alignment = "text-left";

cell.setTextAlignment(alignment);
```
Cell – setVerticalAlignment(String alignment)
Sets the vertical alignment for this cell.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alignment</td>
<td>String</td>
<td>Vertical alignment setting. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• bottom: Aligns contents to the bottom.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• mid: Aligns contents to the center.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• top: Aligns contents to the top.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set vertical alignment on a cell. For a document usage example, see Document API.

```javascript
var cell = new sn_pdfgeneratorutils.Cell(1, 3);
var alignment = "mid";
cell.setVerticalAlignment(alignment);
```

CertificateEncryption - Scoped

APIs available for encrypting certificates in scoped applications.

Use these methods to generate a hash for the certificate, sign data using a private key, and generate a message authentication code.


Instantiates a CertificateEncryption object in a scoped application.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CertificateEncryption - generateMac(String key, String algorithm, String data)

Generates the Message Authentication Code (MAC), which is used to authenticate a message.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Key to use to sign the message. Must be Base64 encoded.</td>
</tr>
<tr>
<td>algorithm</td>
<td>String</td>
<td>Algorithm to use to generate the MAC: HmacSHA256, HmacSHA1, HmacMD5, and so on.</td>
</tr>
<tr>
<td>data</td>
<td>String</td>
<td>Data to process.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>MAC in base64 format.</td>
</tr>
</tbody>
</table>

Example

```javascript
var mac = new CertificateEncryption;
var key = "sample_key";
key = gs.base64Encode(key);
mac.generateMac(key, "HmacSHA256", "sample_data");
```

CertificateEncryption - getThumbPrint(String certificateID, String algorithm)

Generates a hash (SHA-1, SHA-256, and so on) for the certificate from Trust Store Cert.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>algorithm</td>
<td>String</td>
<td>SHA-1, SHA-256, and so on</td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>Thumbprint in base64 format.</td>
<td></td>
</tr>
</tbody>
</table>

**CertificateEncryption - getThumbPrintFromKeystore(String certificateID, String alias, String algorithm)**

Generates a hash (SHA-1, SHA-256, and so on) for the certificate from the keystore entry.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>alias</td>
</tr>
<tr>
<td>algorithm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

**CertificateEncryption - sign(String certificateID, String alias, String aliaspassword, String algorithm, String datatosign)**

Signs the data using the private key and the specified algorithm.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>alias</td>
</tr>
<tr>
<td>aliaspassword</td>
</tr>
<tr>
<td>algorithm</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>• NONEwithRSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MD2withRSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MD5withRSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA1withRSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA224withRSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA256withRSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA384withRSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA512withRSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NONEwithDSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA1withDSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA224withDSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA256withDSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA512withECDSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA1withECDSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA224withECDSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA256withECDSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA384withECDSA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SHA512withECDSA</td>
</tr>
</tbody>
</table>

| datatosign | String | Data to sign. |

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Signed data in base64 format.</td>
</tr>
</tbody>
</table>

```javascript
var ce = new CertificateEncryption;
ce.sign("recordID", "alias", "password", "SHA1withRSA", "sign this data");
```

### ChangeCollisionHelper - Global

Helper functions found in the Change Management Collision Detector Plugin.
Use these methods in server side scripts, or when using AJAX calls on the client. You must have the Change Management Collision Detector Plugin installed to use these methods.

**ChangeCollisionHelper - addCiToChangeAffectedCis(String ci, String changeld)**

Adds the CI to the change's affected CI list.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>ci</td>
<td>String</td>
<td>The sys_id of the configuration item</td>
</tr>
<tr>
<td>changeld</td>
<td>String</td>
<td>The change record's sys_id</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>void</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**ChangeCollisionHelper - getAffectedCisByChangeld(String changeld)**

Returns the Affected CI sys_ids for the given change.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>changeld</td>
<td>String</td>
<td>A change record's sys_id</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>Array</td>
<td>An array of sys_ids of affected CIs.</td>
<td></td>
</tr>
</tbody>
</table>

**ChangeCollisionHelper - getBlackoutsByDate(GlideDateTime startDate, GlideDateTime endDate)**

Returns any blackout that overlap the period defined by startDate and endDate.

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>GlideDateTime</td>
<td>The beginning date</td>
</tr>
<tr>
<td>endDate</td>
<td>GlideDateTime</td>
<td>The ending date</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of blackouts (blackoutId:stringSpan).</td>
</tr>
</tbody>
</table>

**ChangeCollisionHelper - getChangesWithAffectedCi(String ci, GlideDateTime startDate, GlideDateTime endDate)**

Returns changes scheduled in the timespan (defined by startDate and endDate) that have the given CI in their Affected CIs list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ci</td>
<td>String</td>
<td>The configuration item's sys_id</td>
</tr>
<tr>
<td>startDate</td>
<td>GlideDateTime</td>
<td>The beginning date</td>
</tr>
<tr>
<td>endDate</td>
<td>GlideDateTime</td>
<td>The ending date of the time span</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of change record's sys_ids</td>
</tr>
</tbody>
</table>

**ChangeCollisionHelper - getChangesWithCi(String ci, GlideDateTime startDate, GlideDateTime endDate)**

Returns the changes that are in the timespan (startDate, endDate) and that are linked to the given CI.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ci</td>
<td>String</td>
<td>The configuration item’s sys_id</td>
</tr>
<tr>
<td>startDate</td>
<td>GlideDateTime</td>
<td>The beginning date</td>
</tr>
<tr>
<td>endDate</td>
<td>GlideDateTime</td>
<td>The ending date of the time span</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of sys_ids for change records</td>
</tr>
</tbody>
</table>

**ChangeCollisionHelper - getCiMaintenanceSchedule(String ci)**

Returns the Maintenance Schedule for a CI.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ci</td>
<td>String</td>
<td>The configuration item’s sys_id</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**ChangeCollisionHelper - getDependants(String ci)**

Returns all the CIs that depend on the given CI.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ci</td>
<td>String</td>
<td>A configuration item’s sys_id</td>
</tr>
</tbody>
</table>
### ChangeCollisionHelper - getDependencies(String ci)

Returns all the CIs that the given CI depends on.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ci</td>
<td>String</td>
<td>The configuration item's sys_id</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of CIs</td>
</tr>
</tbody>
</table>

### ChangeCollisionHelper - isCiInAffectedCis(String ci, String changecId)

Check if an CI is already in the change's affected CIs list.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ci</td>
<td>String</td>
<td>The sys_id of the configuration item</td>
</tr>
<tr>
<td>changecId</td>
<td>String</td>
<td>The change record's sys_id</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the CI already is in the change's affected CI list.</td>
</tr>
</tbody>
</table>

### ChangeCollisionHelper - isDateInCiMaintenanceWindows(GlideDateTime startDate, GlideDateTime endDate, String maintenanceWindow)

Checks if the time span defined by startDate and endDate falls in the CI's maintenance window.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>GlideDateTime</td>
<td>The beginning date</td>
</tr>
<tr>
<td>endDate</td>
<td>GlideDateTime</td>
<td>The ending date</td>
</tr>
<tr>
<td>maintenanceWindow</td>
<td>String</td>
<td>The configuration item's sys_id</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the time span is in the CI's maintenance window</td>
</tr>
</tbody>
</table>

**ChangeConflict - Global**

Helper functions found in the Change Management Collision Detector Plugin. Use these methods in server side scripts, or when using AJAX calls on the client. You must have the Change Management Collision Detector Plugin installed to use these methods.

**ChangeConflict(String cild, String changeld, Number type)**

Creates an instance of ChangeConflict.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cild</td>
<td>String</td>
<td>A configuration item's sys_id</td>
</tr>
<tr>
<td>changeld</td>
<td>String</td>
<td>A change request's sys_id</td>
</tr>
<tr>
<td>type</td>
<td>Number</td>
<td>A value from the conflict's type choice list</td>
</tr>
</tbody>
</table>

**ChangeConflict - toString()**

Returns a string representation of the conflict.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## ChangeConflictHandler - Global

Helper functions found in the Change Management Collision Detector Plugin. Use these methods in server side scripts, or when using AJAX calls on the client. You must have the Change Management Collision Detector Plugin installed to use these methods.

### ChangeConflictHandler - addChangeConflict(String chgConflict)

Adds the Change Conflict to a Change Conflict Container.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chgConflict</td>
<td>String</td>
<td>The sys_id of the change conflict</td>
</tr>
</tbody>
</table>

### ChangeConflictHandler - changeConflictHandler()

Creates an instance of ChangeConflictHandler.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### ChangeConflictHandler - deleteConflictsByChangeId(String changeld)

Deletes conflicts that are associated with the same change request (by sys_id).

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>changeld</td>
<td>String</td>
<td>The sys_id of the change request</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**ChangeConflictHandler - getConflicts()**

Returns an array of Change Conflicts from a Change Conflict Container.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of change conflicts</td>
</tr>
</tbody>
</table>

**ChangeConflictHandler - saveConflicts()**

Writes out the Change Conflicts in a Change Conflict Container array to individual Change Conflict records.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
CheckDefinitionTestBuilder API - Scoped

The `CheckDefinitionTestBuilder` script include enables building test check requests on check definitions.

This API requires the Agent Client Collector Framework (sn_agent) store application and is provided within the `sn_agent` namespace. For more information, refer to Agent Client Collector.

For the REST API solution, refer to Agent Client Collector API.

This API includes methods that enable the following:

- Set the check definition to test
- Set the configuration item to run the test against

You can also specify one of the following identifiers to use during the test:

- Credential sys_id
- Credential alias id
- Credential name

**CheckDefinitionTestBuilder - CheckDefinitionTestBuilder()**

Creates an `CheckDefinitionTestBuilder` instance.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

The following example shows how to initialize `CheckDefinitionTestBuilder`.

```csharp
var builder = new sn_agent.CheckDefinitionTestBuilder();
```

**CheckDefinitionTestBuilder - build()**

Builds the test check request with the arguments set.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JSON object with two attributes – the test result record ID and an error. If the build is successful, the test result ID returns and error is null. If an error occurs during the build, the test result ID is null and the error displays.</td>
</tr>
</tbody>
</table>

The following example shows how build a test check request:

```javascript
var checkId = "158279505372b30034b8ddee7f7b1270";
var ciId = "cc0dcc855374301062d1ddee7f7b12b0";
var builder = new sn_agent.CheckDefinitionTestBuilder();

var testResultJson = builder.withCheckId(checkId).withCiId(ciId).build();

if (!gs.nil(testResultJson.error))
gs.error(testResultJson.error);
```

**CheckDefinitionTestBuilder - withCheckId(String checkId)**

Sets the sys_id of the check definition to test.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>checkId</td>
<td>String</td>
<td>Sys_id of the check definition listed in the Check Definitions [sn_agent_check_def] table.</td>
</tr>
</tbody>
</table>

The following example shows how create a test check:

```javascript
var checkId = "158279505372b30034b8ddee7f7b1270";
var ciId = "cc0dcc855374301062d1ddee7f7b12b0";
var builder = new sn_agent.CheckDefinitionTestBuilder();
var testResultJson = builder.withCheckId(checkId).withCiId(ciId).build();
if (!gs.nil(testResultJson.error))
gs.error(testResultJson.error);`
CheckDefinitionTestBuilder - withCiId(String ciId)
Sets the sys_id of the configuration item to run the test against.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ciId</td>
<td>String</td>
<td>Sys_id of a CMDB Configuration Item.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckDefinitionTestBuilder</td>
<td>The builder instance the method was invoked for.</td>
</tr>
</tbody>
</table>

The following example shows how to create a test check request with:

```javascript
var checkId = "158279505372b30034b8ddeeff7b1270";
var ciId = "cc0dcc855374301062d1ddeeff7b12b0";
var builder = new sn_agent.CheckDefinitionTestBuilder();
var testResultJson = builder.withCheckId(checkId).withCiId(ciId).build();
if (!gs.nil(testResultJson.error))
  gs.error(testResultJson.error);
```

CheckDefinitionTestBuilder - withCredentialsAliasId(String credentialsAliasId)
Sets the credentials alias sys_id to use during the test.

⚠️ Note: Running this removes any previous assignment done by calling withCredentialsName, withCredentialsAliasId or withCredentialsAliasName.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsAliasId</td>
<td>String</td>
<td>Sys_id of a credentials record.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckDefinitionTestBuilder</td>
<td>The builder instance the method was invoked for.</td>
</tr>
</tbody>
</table>

The following example shows how create a test check and set the credentials alias ID.
var checkId = "158279505372b30034b8ddee7b1270";
var credAlias = "<sys_id>";
var builder = new sn_agent.CheckDefinitionTestBuilder();
var testResultJson =
    builder.withCheckId(checkId).withCredentialsAliasId(credAlias).build();
if (!gs.nil(testResultJson.error))
    gs.error(testResultJson.error);

CheckDefinitionTestBuilder - withCredentialsAliasName(String credentialsAliasName)

Sets the credentials alias name to use during the test. If the given value matches the name of multiple credentials aliases records, then one of them is chosen randomly.

Note: Running this removes any previous assignment done by calling withCredentialsName, withCredentialsAliasId or withCredentialsAliasName.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsAliasName</td>
<td>String</td>
<td>Name of a credentials alias.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckDefinitionTestBuilder</td>
<td>builder instance the method was invoked for.</td>
</tr>
</tbody>
</table>

The following example shows how create a test check and set the credentials alias name.

var checkId = "158279505372b30034b8ddee7b1270";
var credentialsAliasName = "credentialsAliasName";
var builder = new sn_agent.CheckDefinitionTestBuilder();
var testResultJson =
    builder.withCheckId(checkId).withCredentialsAliasName(credentialsAliasName).build();
if (!gs.nil(testResultJson.error))
    gs.error(testResultJson.error);

CheckDefinitionTestBuilder - withCredentialsId(String credentialsId)

Sets the credentials sys_id to use during the test.
**Note:** Running this removes any previous assignment done by calling `withCredentialsName`, `withCredentialsAliasId` or `withCredentialsAliasName`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsId</td>
<td>String</td>
<td>Sys_id of a credentials record.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckDefinitionTestBuilder</td>
<td>The builder instance the method was invoked for.</td>
</tr>
</tbody>
</table>

The following example shows how create a test check and set the credentials ID.

```javascript
var checkId = "158279505372b30034b8d2eef7b1270";
var credID = "<sys_id>";
var builder = new sn_agent.CheckDefinitionTestBuilder();
var testResultJson = builder.withCheckId(checkId).withCredentialsId(credID).build();
if (!gs.nil(testResultJson.error))
    gs.error(testResultJson.error);
```

**CheckDefinitionTestBuilder - withCredentialsName(String credentialsName)**

Sets the credentials name to use during the test. If the given value is the name of several credentials records, then one of them is chosen randomly.

**Note:** Running this removes any previous assignment done by calling `withCredentialsName`, `withCredentialsAliasId` or `withCredentialsAliasName`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsName</td>
<td>String</td>
<td>Name of the credentials record.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckDefinitionTestBuilder</td>
<td>The builder instance the method was invoked for.</td>
</tr>
</tbody>
</table>
The following example shows how to create a test check and set the credentials name.

```javascript
var checkId = "158279505372b30034b8dfeeff7b1270";
var credname = "credentialsName";
var builder = new sn_agent.CheckDefinitionTestBuilder();
var testResultJson = builder.withCheckId(checkId).withCredentialsName(credName).build();
if (!gs.nil(testResultJson.error))
gs.error(testResultJson.error);
```

**CheckInstanceTestBuilder API - Scoped**

The `CheckInstanceTestBuilder` script includes enables building test check requests on check instances.

This API requires the Agent Client Collector Framework (sn_agent) store application and is provided within the `sn_agent` namespace. For more information, refer to `Agent Client Collector`.

For the REST API solution, refer to `Agent Client Collector API`.

This API includes methods that enable the following:

- Set the check instance to test
- Set the configuration item to run the test against

You can also specify one of the following identifiers to use during the test:

- Credential sys_id
- Credential alias id
- Credential name

**CheckInstanceTestBuilder - CheckInstanceTestBuilder()**

Creates an `CheckInstanceTestBuilder` instance.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to initialize `CheckInstanceTestBuilder`.

```javascript
var builder = new sn_agent.CheckInstanceTestBuilder();
```
CheckInstanceTestBuilder - build()
Builds the test check request with the arguments set.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JSON object with two attributes – the test result record ID and an error. If the build is successful, the test result ID returns and error is null. If an error occurs during the build, the test result ID is null and the error displays.</td>
</tr>
</tbody>
</table>

The following example shows how build a test check request:

```java
var checkId = "158279505372b30034b8ddeeff7b1270";
var ciId = "cc0dcc855374301062d1ddeeff7b12b0";
var builder = new sn_agent.CheckInstanceTestBuilder();

var testResultJson = builder.withCheckId(checkId).withCiId(ciId).build();

if (!gs.nil(testResultJson.error))
    gs.error(testResultJson.error);
```

CheckInstanceTestBuilder - withCheckId(String checkId)
Sets the sys_id of the check definition to test.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>checkId</td>
<td>String</td>
<td>Sys_id of the check definition listed in the Check Definitions [sn_agent_check_def] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckInstanceTestBuilder</td>
<td>builder instance the method was invoked for.</td>
</tr>
</tbody>
</table>
The following example shows how to create a test check.

```javascript
var checkId = "158279505372b30034b8ddee77b1270";
var ciId = "cc0dcc855374301062d1ddee77b12b0";
var builder = new sn_agent.CheckInstanceTestBuilder();
var testResultJson = builder.withCheckId(checkId).withCiId(ciId).build();
if (!gs.nil(testResultJson.error))
    gs.error(testResultJson.error);
```

**CheckInstanceTestBuilder - withCiId(String ciId)**

Sets the sys_id of the configuration item to run the test against.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ciId</td>
<td>String</td>
<td>Sys_id of a CMDB Configuration Item.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckInstanceTestBuilder</td>
<td>builder instance the method was invoked for.</td>
</tr>
</tbody>
</table>

The following example shows how to create a test check request with

```javascript
var checkId = "158279505372b30034b8ddee77b1270";
var ciId = "cc0dcc855374301062d1ddee77b12b0";
var builder = new sn_agent.CheckInstanceTestBuilder();
var testResultJson = builder.withCheckId(checkId).withCiId(ciId).build();
if (!gs.nil(testResultJson.error))
    gs.error(testResultJson.error);
```

**CheckInstanceTestBuilder - withCredentialsAliasId(String credentialsAliasId)**

Sets the credentials alias sys_id to use during the test.

⚠️ **Note:** Running this removes any previous assignment done by calling withCredentialsName, withCredentialsAliasId or withCredentialsAliasName.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsAliasId</td>
<td>String</td>
<td>Sys_id of a credentials alias record.</td>
</tr>
</tbody>
</table>
The following example shows how to create a test check and set the credentials alias ID.

```javascript
var checkId = "158279505372b30034b8ddee0ff7b1270";
var credAlias = "<sys_id>";
var builder = new sn_agent.CheckInstanceTestBuilder();
var testResultJson =
    builder.withCheckId(checkId).withCredentialsAliasId(credAlias).build();
if (!gs.nil(testResultJson.error))
    gs.error(testResultJson.error);
```

---

**CheckInstanceTestBuilder - withCredentialsAliasName(String credentialsAliasName)**

Sets the credentials alias name to use during the test. If the given value matches the name of multiple credentials aliases records, then one of them is chosen randomly.

ℹ️ **Note:** Running this removes any previous assignment done by calling `withCredentialsName`, `withCredentialsAliasId` or `withCredentialsAliasName`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsAliasName</td>
<td>String</td>
<td>Name of a credentials alias.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckInstanceTestBuilder</td>
<td>builder instance the method was invoked for.</td>
</tr>
</tbody>
</table>

The following example shows how to create a test check and set the credentials alias name.

```javascript
var checkId = "158279505372b30034b8ddee0ff7b1270";
var credentialsAliasName = "credentialsAliasName";
var builder = new sn_agent.CheckInstanceTestBuilder();
```
```javascript
var testResultJson = 
  builder.withCheckId(checkId).withCredentialsAliasName(credentialsAliasName).build();
if (!gs.nil(testResultJson.error))
  gs.error(testResultJson.error);
```

**CheckInstanceTestBuilder - withCredentialsId(String credentialsId)**

Sets the credentials sys_id to use during the test.

⚠️ **Note**: Running this removes any previous assignment done by calling
`withCredentialsName`, `withCredentialsAliasId` or `withCredentialsAliasName`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsId</td>
<td>String</td>
<td>Sys_id of a credentials record.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckInstanceTestBuilder</td>
<td>builder instance the method was invoked for.</td>
</tr>
</tbody>
</table>

The following example shows how create a test check and set the credentials ID.

```javascript
var checkId = "158279505372b30034b8d9eff7b1270";
var credID = "<sys_id>";
var builder = new sn_agent.CheckInstanceTestBuilder();
var testResultJson = builder.withCheckId(checkId).withCredentialsId(credID).build();
if (!gs.nil(testResultJson.error))
  gs.error(testResultJson.error);
```

**CheckInstanceTestBuilder - withCredentialsName(String credentialsName)**

Sets the credentials name to use during the test. If the given value is the name of
several credentials records, then one of them is chosen randomly.

⚠️ **Note**: Running this removes any previous assignment done by calling
`withCredentialsName`, `withCredentialsAliasId` or `withCredentialsAliasName`. 
The following example shows how to create a test check and set the credentials name.

```javascript
var checkId = "158279505372b30034b8ddee7b1270";
vartestResultJson = builder.withCheckId(checkId).withCredentialsName(credName).build();
if (!gs.nil(testResultJson.error))
gs.error(testResultJson.error);
```

**CheckInstanceTestBuilder - withProxyAgentId(String agentId)**

Sets the agent to run the test. Allowed only in case the tested check instance and policy are proxy.

The following example shows how to create a test check and set the proxy agent ID.

```javascript
var checkId = "158279505372b30034b8ddee7b1270";
varagentId = "<agentId>";
```
var builder = new sn_agent.CheckInstanceTestBuilder();
var testResultJson = builder.withCheckId(checkId).withAgentId(agentId).build();
if (!gs.nil(testResultJson.error))
  gs.error(testResultJson.error);

CIData - Global

Utility class for working with CI data structures in JavaScript.

Instances of this class represent a CI (both its base record and any related lists), and the provided methods allow loading from or storing both the base record and related lists.

This class acts as a container for simple classes acting as maps of property name/value pairs. This class contains three kinds of such maps.

• A single map representing the base CI table entry (such as one row cmdb_ci_linux_server and its superclasses). This map can be retrieved with the `getData()` method.

• Arrays of maps representing the instances of related lists, with each array representing a single related list and each element of such an array representing a single row of that related list. These arrays can be retrieved with the `getRelatedList()` method.

• Arrays of maps representing the instances of many-to-many tables, with each array representing a single many-to-many table and each element of such an array representing a single row of that many-to-many table. These arrays can be retrieved with the `getM2MTable()` method.

Use with any server-side discovery script.

CIData - addRelatedList(Array relatedList)

Adds the specified related list to this instance.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relatedList</td>
<td>Array</td>
<td>Related list to add.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
This example creates and adds a related list to the CIData object.

```javascript
var ciDataObj = new CIData();
var rl = new CIRelatedList('cmdb_serial_number', 'cmdb_ci');
var sr = {};
  sr['serial_number_type'] = "bios";
  sr['serial_number'] = "1BC5E4z89246";
  sr['valid'] = "true";
rl.addRec(sr);
  ciDataObj.addRelatedList(rl);
```

CIData - CIData(String debugFlag)

Creates an instance of the CIData class.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| debugFlag | String | Flag that indicates whether to enable debug logging. Valid values:  
  • True: Debug logging is enabled.  
  • False: Debug logging is not enabled. |

This example creates and adds a related list to the CIData object.

```javascript
var ciDataObj = new CIData();
var rl = new CIRelatedList('cmdb_serial_number', 'cmdb_ci');
var sr = {};
  sr['serial_number_type'] = "bios";
  sr['serial_number'] = "1BC5E4z89246";
  sr['valid'] = "true";
rl.addRec(sr);
  ciDataObj.addRelatedList(rl);
```

CIData - convertRelatedList(String sensor, String tableName, String refField, String keyName)

Converts the specified related list to a related list in a given sensor.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sensor</td>
<td>String</td>
<td>The sensor</td>
</tr>
<tr>
<td>tableName</td>
<td>String</td>
<td>The table name</td>
</tr>
<tr>
<td>refField</td>
<td>String</td>
<td>The reference field</td>
</tr>
<tr>
<td>keyName</td>
<td>String</td>
<td>The key name</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CIData - fromXML(String xml)

Initializes the current CIData instance from the specified XML string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xml</td>
<td>String</td>
<td>An XML string</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CIData - getData()

Gets the data map in the base CI record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>Object</td>
<td>The data map</td>
<td></td>
</tr>
</tbody>
</table>

**CIData - getM2MTable(String table, String refField)**

Gets an array of data maps in the given many-to-many list (to this CI).

The array is not in any particular order. If this is not a many-to-many list, returns an empty array.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>The name of the table containing the related list. In the case of a many-to-many list, this is the name of the many-to-many table, not the target table.</td>
</tr>
<tr>
<td>refField</td>
<td>String</td>
<td>The name of the field in the related list that refers to this CI. In the case of a many-to-many list, this is the name of the referring field in the many-to-many table, not in the target table.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>Array</td>
</tr>
</tbody>
</table>

**CIData - getRelatedList(String table, String refField)**

Gets an array of maps of data in the given related list (to this CI).

The array is not in any particular order. In the case of a many-to-many list, this will be an array of instances of the target table, not the many-to-many table. For example, given 'cmdb_software_instance' and 'installed_on' (a many-to-many table and the field that refers to a CI), this method will return an array of maps representing cmdb_ci_spkg (the target table) instances.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>Name of the table containing the related list. In the case of a many-to-many list, this is the name of the many-to-many table, not the target table.</td>
</tr>
<tr>
<td>refField</td>
<td>String</td>
<td>Name of the field in the related list that refers to this CI. In the case of a many-to-many list, this is the name of the referring field in the many-to-many table, not in the target table.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of maps of data in the specified related list.</td>
</tr>
</tbody>
</table>

### CIData - getRelatedListInstance(String table, String refField)

Returns the instance of CIRelatedList for the given list.

## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>Name of the table containing the related list. In the case of a many-to-many list, this is the name of the many-to-many table, not the target table.</td>
</tr>
<tr>
<td>refField</td>
<td>String</td>
<td>Name of the field in the related list that refers to this CI. In the case of a many-to-many list, this is the name of the referring field in the many-to-many table, not in the target table.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Instance of the CIRelatedList for the specified list.</td>
</tr>
</tbody>
</table>

### CIData - init()

Initializes this instance in preparation for loading a new CI.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var cdta = CIData('false');
cdta.init();
```

**CIData - loadFromCI(String cmdb_ci)**

Loads the current data from a specified sys_id.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmdb_ci</td>
<td>String</td>
<td>The sys_id of the CI</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The current data from the specified CI</td>
</tr>
</tbody>
</table>

```javascript
var cdta = CIData('false');
cdta.init();
cdta.loadFromCI('ccaf9c0a8016400b98a06818d57c7');
```

**CIData - toString()**

Converts the name of this instance to a string.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the instance</td>
</tr>
</tbody>
</table>

CIData - toXML()

Returns an XML string containing a serialized version of this instance (including any related lists).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The XML string containing a serialized version of this instance and any related lists.</td>
</tr>
</tbody>
</table>

```javascript
var cdata = CIData();
cdata.toXML();
```

CIIdentification - Global

Main class for discovery CI identification.

Use this with any server-side discovery script for CI identification.

CIIdentification - CIIdentification(String ciData, String logger)

Creates an instance of the CIIdentification class.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ciData</td>
<td>String</td>
<td>The CI data to identify</td>
</tr>
<tr>
<td>logger</td>
<td>String</td>
<td>The discovery logger</td>
</tr>
</tbody>
</table>

### CIIdentification - debug(String msg)

Logs a message to the CI Identification log if debug logging is turned on.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>The message to log</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### CIIdentification - process()

Identifies the CI. This is the entry point for the entire CI Identification process.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDResult</td>
<td>The result returned by the identifier</td>
</tr>
</tbody>
</table>

### CIIdentifierResult - Global

Gets the result returned by an identifier. Use with any server-side discovery script.
CIIdentifierResult - CIIdentifierResult(Array matched, Boolean matchable, Boolean terminatedChain)

Creates an instance of the CIIdentifierResult class.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>matched</td>
<td>Array</td>
<td>An array of sys_ids of matching CIs; may have any number of entries including zero.</td>
</tr>
<tr>
<td>matchable</td>
<td>Boolean</td>
<td>True if identifier had sufficient data to match</td>
</tr>
<tr>
<td>terminatedChain</td>
<td>Boolean</td>
<td>True if the identifier chain should stop processing filters. False to continue.</td>
</tr>
</tbody>
</table>

CimCIData - Global

Provides a wrapper for CIM CI data manipulation.

Use with any server-side discovery script.

CimCIData - addSerial(String type, String serial)

Adds a serial number to the current CI, by type.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>The serial number type. If out-of-box, use the CimCIData.serialType enum.</td>
</tr>
<tr>
<td>serial</td>
<td>String</td>
<td>The serial number</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The current CI</td>
</tr>
</tbody>
</table>

CimCIData - SerialRecord(String type, String serial)

Creates a new serial record.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>The serial type to assign to the CI. If out-of-box, uses the CimCIData.serialType enum.</td>
</tr>
<tr>
<td>serial</td>
<td>String</td>
<td>The serial number to assign.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CimCIData - setMakeAndModel(String make, String model)

Sets the make and model for the current CI.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>make</td>
<td>String</td>
<td>The manufacturer</td>
</tr>
<tr>
<td>model</td>
<td>String</td>
<td>The model</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CimIDSensor - Global

Performs identification for CIM probe results.

Use in any server-side script to perform identification of CIM probe results.

CimIDSensor - configureTriggeredProbe(String probe)

Configures a triggered probe before it is launched.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>probe</td>
<td>String</td>
<td>An already configured probe that is ready to fire.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True to fire this probe, false to skip it.</td>
</tr>
</tbody>
</table>

### CimIDSensor - getInstanceHashToken()

Gets the instance hash token.

## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The hash token</td>
</tr>
</tbody>
</table>

### CimIDSensor - prepare()

Prepares the probe results.

## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
CimIDSensor - process(Object results, CimCIData cimData, CimIDSensor sensor)
Processes all CIM probe query results for a CIM ID sensor.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Object</td>
<td>The query results</td>
</tr>
<tr>
<td>cimData</td>
<td>CimCIData</td>
<td>The CI data of the device being identified.</td>
</tr>
<tr>
<td>sensor</td>
<td>CimIDSensor</td>
<td>The wrapping sensor, extends CimIDSensor</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CimIDSensor - runMultiProbeScript(String script, Object probeResult, String probeRecord)
Runs a multi-probe script against a probe result.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>script</td>
<td>String</td>
<td>The script</td>
</tr>
<tr>
<td>probeResult</td>
<td>Object</td>
<td>The probe result</td>
</tr>
<tr>
<td>probeRecord</td>
<td>String</td>
<td>The probe ID</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CimIDSensor - updateDeviceCount()
Updates the device count.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**CimInstanceToken - Global**

 Represents the unique query identity of a CIM instance.  
Use with any server-side discovery script.

**CimInstanceToken - getHashToken(String cimomip)**

 Parses the instance token.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cimomip</td>
<td>String</td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The hash token</td>
</tr>
</tbody>
</table>

**CimInstanceToken - parse(XMLObj instance)**

 Parses the instance token.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
<td>XMLObj</td>
<td>The CIM instance</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>The parsed instance token, for example,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CIM_ClassName{Key1='Value1',Key2='Value2'}</td>
<td></td>
</tr>
</tbody>
</table>

**CimProbe - Global**

Maintains CIM probe meta data.

Use in any server-side script where you need to maintain a CIM probe.

**CimProbe - getNamedQueries()**

Returns all the named queries for the current CIM probe.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

**CimProbe - getQueries()**

Returns all the queries for the current CIM probe.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Array</td>
</tr>
</tbody>
</table>
CimProbe - getQueryRecords()
Returns all the records in the CIM Probe table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The query result</td>
</tr>
</tbody>
</table>

CimProbeResult - Global
Processes CIM probe results.
Use to process CIM probe results in any server-side discovery script.

CimProbeResult - getNamedInstances(String namedQueries)
Gets the named instances for the specified named queries.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>namedQueries</td>
<td>String</td>
<td>The named queries</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The named instances</td>
</tr>
</tbody>
</table>

CimProbeResult - getQueries()
Gets all the queries for the current CIM probe.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String Array</td>
<td>All Queries</td>
</tr>
</tbody>
</table>

**CIRelatedList - Global**

Utility class for working with CI Related lists.

Each instance of this class contains the details of a single list that is related to a particular CI. The details of this list and the contents of the list are included.

Use with any server-side discovery script.

**CIRelatedList - addRec(String record)**

Adds the specified record to the related list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>String</td>
<td>The record to add</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**CIRelatedList - addRecs(String records)**

Sets the given records to the related list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>records</td>
<td>String</td>
<td>The records to set</td>
</tr>
</tbody>
</table>
## CIRelatedList - appendXMLChildFld(String parent, String name, String datum)

Appends an XML child to a field.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent</td>
<td>String</td>
<td>The parent element</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The element name</td>
</tr>
<tr>
<td>datum</td>
<td>String</td>
<td>The data to append</td>
</tr>
</tbody>
</table>

## CIRelatedList - appendXMLChildRecord(String parent, String name, String record)

Appends an XML child to a record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent</td>
<td>String</td>
<td>The parent element</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The element name</td>
</tr>
<tr>
<td>record</td>
<td>String</td>
<td>The record</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
CIRelatedList - checkSysCollection()
Checks to see if this is a SysCollection table. If it is, sets the target table name and reference field name.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIRelatedList</td>
<td>An instance of CIRelatedList for the specified list.</td>
</tr>
</tbody>
</table>

CIRelatedList - checkSysM2M()
Checks to see if this is a Sys many-to-many table. If it is, sets the target table name and reference field name.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if it is a sys many-to-many table; otherwise, false.</td>
</tr>
</tbody>
</table>

CIRelatedList - CIRelatedList(String table, String reffield, String cmdb_cname, String debugFlag)
Creates an instance of the CIRelatedList class. With no arguments, simply constructs an empty instance.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>The table containing the related list. If many-to-many, the many-to-many table.</td>
</tr>
<tr>
<td>refField</td>
<td>String</td>
<td>The reference field in the table for this related list.</td>
</tr>
<tr>
<td>cmdb_ci</td>
<td>String</td>
<td>The sys_id of the CI this list is related to.</td>
</tr>
<tr>
<td>debugFlag</td>
<td>String</td>
<td>If true, debug logging is enabled.</td>
</tr>
</tbody>
</table>

#### CIRelatedList - fromXML(String element)

Initializes this instance from the specified XML element.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>element</td>
<td>String</td>
<td>An XML element</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### CIRelatedList - populate()

Populates the `this.records` field of this instance.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
CIRelatedList - toString(Array result)
Makes a string representation of this instance, pushing each line onto the end of
the given result array.

This method is called from CIData.toString(), and assumes related list lines are to
be indented two spaces.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Array</td>
<td>The result to convert</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CIRelatedList - toXML(String document, String element)
Serializes this instance to XML in the given document and <rl> element.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>document</td>
<td>String</td>
<td>The document</td>
</tr>
<tr>
<td>element</td>
<td>String</td>
<td>The rl element</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CIRelatedList - remove()
Removes all the related list items.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### CIRelatedList - xmlToRecord(String records, String element)

Converts a related list from XML to record format.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>records</td>
<td>String</td>
<td>The record to create</td>
</tr>
<tr>
<td>element</td>
<td>String</td>
<td>The XML element</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### CIUtils - Global

CIUtils is a utility class for working with configuration items.

By default, when traversing CI relationships the system will use a max depth of 10. This can be overridden in the `glide.relationship.max_depth` property.

The maximum number of items returns is 1000. This can be overridden in the `glide.relationship.threshold` property.

The CIUtils class is available to server-side scripts.

### CIUtils - servicesAffectedByCI(String CI_sys_id)

Determines which business services are affected by a specific CI.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CI_sys_id</td>
<td>String</td>
<td>The sys_id of a configuration item (cmdb_ci) to check.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of sys_id values for cmdb_ci records downstream of (or affected by) the specified item.</td>
</tr>
</tbody>
</table>

This example displays the names of the services affected by the CI items with the name = Linux100.

```javascript
var CIUtil = new CIUtils();

//get a server record
var server = new GlideRecord("cmdb_ci_server");
server.addQuery("name", "Linux100");
server.query();
if (server.next()) {
    //get the affected services, array of ids
    var serviceIds = CIUtil.servicesAffectedByCI(server.getUniqueValue());
    for (var i=0; i < serviceIds.length; i++) {
        //get the service record
        var service = new GlideRecord("cmdb_ci_service");
        service.get(serviceIds[i]);
        gs.print(service.getDisplayValue());
    }
}
```

Output:

- Client Services
- IT Services
- Bond Trading

**CIUtils - servicesAffectedByTask(GlideRecord task)**

Determines which business services are affected by the specified task.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>task</td>
<td>GlideRecord</td>
<td>A task GlideRecord, for example incident, change_request, or problem.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of sys_id values for CIs downstream of (or affected by) the configuration item referenced by the task's cmdb_ci field.</td>
</tr>
</tbody>
</table>

This example displays the names of the services affected by the incident INC00050.

```javascript
var CIUtil = new CIUtils();

// get an incident record
var inc = new GlideRecord("incident");
inc.addQuery("number", "INC00050");
inc.query();
if (inc.next()) {
    // get the affected services, array of ids
    var serviceIds = CIUtil.servicesAffectedByTask(inc);
    for (var i=0; i < serviceIds.length; i++) {
        // get the service record
        var service = new GlideRecord("cmdb_ci_service");
        service.get(serviceIds[i]);
        gs.print(service.getDisplayValue());
    }
}
```

Output:

- IT Services
- Email
- Windows Mobile
- Electronic Messaging
- Outlook Web Access (OWA)
- Blackberry

### ClassificationSolutionStore - Global

Enables storing and retrieving solutions.
The `ClassificationSolutionStore` API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the `sn_ml` namespace.

**ClassificationSolutionStore - add(Object mlSolution)**

Adds a new solution object to the store and returns a unique name.

ℹ️ **Note:** Label values do not need to be unique. For example, if you run this method with the same label 10 times, this method adds 10 different uniquely-named objects to the store.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mlSolution</td>
<td>ClassificationSolution()</td>
<td>object to add to the store.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>System-generated solution name.</td>
</tr>
</tbody>
</table>

The following example shows how to add a solution to the store. Use `ClassificationSolution - submitTrainingJob()` to run the training job after adding it to the store.

```javascript
// Create a dataset
var myData = new sn_ml.DatasetDefinition({
  'tableName': 'incident',
  'fieldNames': ['assignment_group', 'short_description', 'description'],
  'encodedQuery': 'activeANYTHING'
});

// Create a solution
var mySolution = new sn_ml.ClassificationSolution({
  'label': "my solution definition",
  'dataset': myData,
  'predictedFieldName': 'assignment_group',
  'inputFieldNames': ['short_description']
});
```
// Add the solution to the store to later be able to retrieve it.
var my_unique_name = sn_ml.ClassificationSolutionStore.add(mySolution);

ClassificationSolutionStore - deleteObject(String name)
Removes a specified solution object from the store.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the ClassificationSolution() object to be deleted.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to delete a solution from the store.

sn_ml.ClassificationSolutionStore.deleteObject("ml_sn_global_global_solution");

ClassificationSolutionStore - get(String name)
Gets a solution object from the store.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of a solution in the store.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>ClassificationSolutionVersion object. Returns an error if the object does not exist.</td>
</tr>
</tbody>
</table>

The following example shows how to get a solution object from the store using the get() method and view its training status using the ClassificationSolution - getActiveVersion() and ClassificationSolutionVersion - getStatus() methods.
// Get status
var mlSolution = sn_ml.ClassificationSolutionStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getStatus(), null, 2)));

Output:

```json
{
    "state":"solution_complete",
    "percentComplete":"100",
    "hasJobEnded":"true"
}
```

**ClassificationSolutionStore - getAllNames(Object options)**

Gets the names of all solution definition records in the store.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Options for restricting results within the specified properties.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>options.label</td>
<td>String</td>
<td>Optional. Label of your solution object.</td>
</tr>
<tr>
<td>options.domainName</td>
<td>String</td>
<td>Optional. Name of the domain for your solution object. Refer to <a href="https://datalabs.servicenow.com/servicenow/article/labs/doc/predictive-intelligence">Domain separation and Predictive Intelligence</a>.</td>
</tr>
<tr>
<td>options.scope</td>
<td>String</td>
<td>Optional. Name of an application scope for your solution object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of strings representing solution object names in the store.</td>
</tr>
</tbody>
</table>
In the following example, the `getAllNames()` method returns a list of all names in the store.

```javascript
gs.print(JSON.stringify(JSON.parse(sn_ml.ClassificationSolutionStore.getAllNames()), null, 2));
```

Output:

```
[
  "ml_x_snc_global_global_my_solution_definition_3",
  "ml_incident_assignment",
  "ml_x_snc_global_global_my_solution_definition",
  "ml_x_snc_global_global_my_solution_definition_2",
  "ml_sn_global_global_incident_service"
]
```

In the following example, the `getAllNames()` method returns only names associated with values set in the `options` parameter.

```javascript
var options = {
  'label' : 'my solution definition',
  'domainName' : 'global',
  'scope' : 'global'
};
var solNames = sn_ml.ClassificationSolutionStore.getAllNames(options);
gs.print(JSON.stringify(JSON.parse(solNames), null, 2));
```

Output:

```
[
  "ml_x_snc_global_global_my_solution_definition"
]
```

**ClassificationSolutionStore - update(String name, Object mlSolution)**

Replaces an existing object in the store with the object passed as a parameter. The object name provided must be empty or match.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the solution to update.</td>
</tr>
<tr>
<td>mlSolution</td>
<td><code>ClassificationSr:ClassificationSolution()</code></td>
<td>object properties to update.</td>
</tr>
</tbody>
</table>
The following example shows how to update a solution object in the store.

```javascript
var solutionUpdate = new sn_ml.ClassificationSolution({
  'label': 'my solution definition',
  'dataset': myData,
  'predictedFieldName': 'assignment_group',
  'inputFieldNames': ['short_description']
});

sn_ml.ClassificationSolutionStore.update('ml_sn_global_global_incident_service',
  solutionUpdate);
```

### ClassificationSolutionVersion - Global

Scriptable object used in Predictive Intelligence stores.

The `ClassificationSolutionVersion` API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the `sn_ml` namespace.

This API is used for working with solution versions based on `ClassificationSolution` API objects in the `ClassificationSolution` store.

The system creates a solution version each time you train a solution definition. Most versions are created during scheduled solution training.

Methods in this API are accessible using the following `ClassificationSolution` methods:

- `getActiveVersion()`
- `getAllVersions()`
- `getLatestVersion()`
- `getVersion()`

### ClassificationSolutionVersion - getPredictionPerformance(Object options)

Calculates solution precision and coverage.

To use this method, you must first create settings using the `setPredictionSettings()` method.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Selects a metric to retrieve within a span of dates. For more information, see <a href="#">Configuring target metrics</a>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{</td>
</tr>
</tbody>
</table>
|            |          |   "metricName" : "String",
|            |          |   "fromDate" : "String",
|            |          |   "toDate" : "String"
|            |          | }                                                                                                                                                                                                         |
| options.metricName | Name of the metric. Valid values: • precision • coverage |                                                                                                                                                                                                      |
| options(fromDate) | Optional. Start date from which to retrieve metrics in system date time format.                                                               |
| options(toDate)   | Optional. End date from which to retrieve metrics in system date time format.                                                                   |

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Parseable JavaScript object containing the results for the input metric.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
</tbody>
</table>
|               |   "metricName" : "String",
|               |   "metricValue" : "String",
|               |   "numberOfPredictionsConsidered" : "String"
|               | }                                                                                                                                              |
| Object.metricName | Name of metric selected as input. Data type: String                                                                                         |
| Object.metricValue | Value of the selected metric. Data type: String                                                                                             |
| Object.numberOfRecordsConsidered | Number of records in the Predictor Results [ml_predictor_results] table considered for calculations.                                      |
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

The following example shows how to get a precision value from the active version of a solution.

```javascript
// Get precision value
var mlSolution = sn_ml.ClassificationSolutionStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getPredictionPerformance({
  "metricName": "precision"})), null, 2));
```

Output:

```
{
  "metricName": "precision",
  "metricValue": "70.10",
  "numberOfPredictionsConsidered": "10"
}
```

**ClassificationSolutionVersion - getPredictionSettings()**

Gets estimated precision, estimated coverage, estimated recall values from the ML Solutions [ml_solution] table and records from the Class Confidence [ml_class] table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Returns prediction settings for each class. Data includes precision, coverage, recall, and distribution values of all classes included in the trained model from the ML Solutions [ml_solution] table. For details, see Predictive model components.</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;.&lt;class name&gt;</code></td>
<td>Array of objects providing training statistics data from the Class Confidence [ml_class] table. For more information, see <a href="#">Configuring target metrics</a>.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.&lt;class name&gt;.precision</code></td>
<td>Estimated solution precision metric for this class. Data type: String (numerical value)</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.&lt;class name&gt;.coverage</code></td>
<td>Estimated solution coverage metric for this class. Data type: String (numerical value)</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.&lt;class name&gt;.recall</code></td>
<td>Estimated solution recall metric for this class. Data type: String (numerical value)</td>
</tr>
</tbody>
</table>

The following example shows how to get prediction settings of an active solution version.

```javascript
var mlSolution = sn_ml.ClassificationSolutionStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getPredictionSettings()), null, 2));
```

Output:

```json
{
  "solutionSettings": {
    "precision": "61.69",
    "coverage": "66.75",
    "recall": "57.54"
  },
  "classSettings": {
    "Approvals": {
      "precision": "100",
```
ClassificationSolutionVersion - getProperties()

Gets solution object properties and version number.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Contents of the Dataset and ClassificationSolution version details. Results vary by object property setup.</td>
</tr>
</tbody>
</table>

```json
{
    "algorithmConfig": {
        "algorithm": "String",
        "targetClassRecall": "String"
    },
    "datasetProperties": {Object},
    "domainName": "String",
    "encoder": {Object},
    "groupByFieldName": "String",
    "inputFieldNames": [Array],
    "isActive": "Boolean",
    "label": "String",
    "name": "String",
    "predictedFieldName": "String"
}
```
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns (continued)</td>
<td></td>
</tr>
<tr>
<td><code>algorithmConfig</code></td>
<td>Method for encoding the solution.</td>
</tr>
<tr>
<td><code>algorithmConfig.algorithm</code></td>
<td>Name of the encoding algorithm for training this solution. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• <code>xgboost</code>: XGBoost encoding to optimize the training.</td>
</tr>
<tr>
<td></td>
<td>• <code>logisticRegression</code>: Method using the logistic regression model for categorical targets such as nominal or ordinal.</td>
</tr>
<tr>
<td><code>algorithmConfig.targetClassRecall</code></td>
<td>Class recall parameter to steer a solution's training to bias a specific class. The recall value is a number between 0 and 100 representing a percentage.</td>
</tr>
<tr>
<td><code>datasetProperties</code></td>
<td>Properties of the <code>DatasetDefinition()</code> object associated with the solution.</td>
</tr>
<tr>
<td></td>
<td>`{</td>
</tr>
<tr>
<td></td>
<td>&quot;encodedQuery&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;fieldDetails&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;fieldName&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;tableName&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><code>datasetProperties.tableName</code></td>
<td>Name of the dataset. For example, <code>&quot;tableName&quot; : &quot;Incident&quot;</code>.</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.datasetProperties.fieldNames</td>
<td>List of field names from the specified table as strings. For example, &quot;fieldNames&quot; : [&quot;short_description&quot;, &quot;priority&quot;]. Data type: Array.</td>
</tr>
<tr>
<td>&lt;Object&gt;.datasetProperties.fieldNames.fieldDetails</td>
<td>List of JavaScript objects that specify field properties. Data type: Array.</td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;name&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;type&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td>Data type: Array.</td>
</tr>
<tr>
<td>&lt;Object&gt;.datasetProperties.fieldDetails.&lt;object&gt;.name</td>
<td>Name of the field defining the type of information to restrict this dataset to. Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.domainName</td>
<td>Domain name associated with this dataset. See Domain separation and Predictive Intelligence. Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.encoder</td>
<td>Encoder object assigned to this solution. See Encoder - Encoder(Object config). Data type: Object.</td>
</tr>
<tr>
<td>&lt;Object&gt;.groupByFieldName</td>
<td>Field name by which the system groups records to build classification solutions. Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.inputFieldNames</td>
<td>List of input field names as strings. The model uses these fields used to make predictions.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.isActive</td>
<td>Flag that indicates whether this version is active. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Version is active.</td>
</tr>
<tr>
<td></td>
<td>• false: Version is not active.</td>
</tr>
</tbody>
</table>

Data type: String

<table>
<thead>
<tr>
<th>&lt;Object&gt;.label</th>
<th>Identifies the prediction task.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;label&quot;: &quot;my first prediction&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

Data type: String

<table>
<thead>
<tr>
<th>&lt;Object&gt;.name</th>
<th>System-assigned name.</th>
</tr>
</thead>
</table>

Data type: String

<table>
<thead>
<tr>
<th>&lt;Object&gt;.predictedFieldName</th>
<th>Identifies a field to be trained for predictability.</th>
</tr>
</thead>
</table>

Data type: String

<table>
<thead>
<tr>
<th>&lt;Object&gt;.processingLanguage</th>
<th>Processing language in two-letter ISO 639-1 language code format.</th>
</tr>
</thead>
</table>

Data type: String

<table>
<thead>
<tr>
<th>&lt;Object&gt;.scope</th>
<th>Object scope. Currently the only valid value is global.</th>
</tr>
</thead>
</table>

Data type: String

| <Object>.stopwords         | Optional. Preset list of strings that the system automatically generates   |
|-----------------------------| based on the language property setting. For details, see Create a custom |
|                              | stopwords list.                                                            |
|                              |                                                                             |
|                              |                                                                             |
|                              |                                                                             |
|                              |                                                                             |

Data type: Array

<table>
<thead>
<tr>
<th>&lt;Object&gt;.trainingFrequency</th>
<th>Frequency to retrain the model.</th>
</tr>
</thead>
</table>

Possible values:
• every_30_days
• every_60_days
• every_90_days
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• every_120_days</td>
</tr>
<tr>
<td></td>
<td>• every_180_days</td>
</tr>
<tr>
<td></td>
<td>• run_once</td>
</tr>
<tr>
<td>Default: run_once</td>
<td>Data type: String.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>&lt;Object&gt;.versionNumber</th>
<th>Version number of the ClassificationSolution object.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String.</td>
<td></td>
</tr>
</tbody>
</table>

The following example gets properties of the active object version in the store.

```javascript
// Get properties
var mlSolution = sn_ml.ClassificationSolutionStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getProperties()), null, 2));
```

**Output:**

```javascript
*** Script: {
"datasetProperties": {
"encodedQuery": "activeANYTHING\^EQ",
"fieldNames": [
"short_description",
"category"
],
"tableName": "incident"
},
"domainName": "global",
"inputFieldNames": [
"short_description"
],
"isActive": "true",
"label": "Incident Categorization_Trainer",
"name": "ml_incident_categorization",
"predictedFieldName": "category",
"processingLanguage": "en",
"stopwords": [
"Default English Stopwords"
]}
```
ClassificationSolutionVersion - getStatus(Boolean includeDetails)

Gets training completion status.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>includeDetails</td>
<td>Boolean</td>
<td>Flag that indicates whether to return status details. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Return additional details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not return additional details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: False</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JavaScript object containing training status information for a ClassificationSolution object.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;state&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;percentComplete&quot;: &quot;Number as a String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;hasJobEnded&quot;: &quot;Boolean value as a String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;details&quot;: {Object}</td>
</tr>
<tr>
<td>&lt;Object&gt;.state</td>
<td>Training completion state. If the training job reaches a terminal state, the job does not leave that state. If the state is terminal, the hasJobEnded property is set to true. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• fetching_files_for_training</td>
</tr>
<tr>
<td></td>
<td>• preparing_data</td>
</tr>
<tr>
<td></td>
<td>• retry</td>
</tr>
<tr>
<td></td>
<td>• solution_cancelled (terminal)</td>
</tr>
<tr>
<td></td>
<td>• solution_complete (terminal)</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>solution_error</td>
<td>(terminal)</td>
</tr>
<tr>
<td>solution_incomplete</td>
<td></td>
</tr>
<tr>
<td>training_request_received</td>
<td></td>
</tr>
<tr>
<td>training_request_timed_out</td>
<td>(terminal)</td>
</tr>
<tr>
<td>training_solution</td>
<td></td>
</tr>
<tr>
<td>uploading_solution</td>
<td></td>
</tr>
<tr>
<td>waiting_for_training</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

<Object>.hasJobEnded: Flag that indicates whether training is complete.
- Valid values:
  - true: Training is complete.
  - false: Training is incomplete.
- Data type: Boolean value as a String

<Object>.percentComplete: Number between zero and 100 representing training percent complete. If the completion percentage is less than 100, the job might be in a terminal state. For example, if training times out.
- Data type: Number as a String

<Object>.details: Object containing a list of additional training details.
- Data type: Object

The following example shows a successful result with training complete.

```javascript
// Get status
var mlSolution = sn_ml.ClassificationSolutionStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getStatus(true), null, 2)));
```

**Output:**

```json
{
    "state":"solution_complete",
```
The following example shows an unsuccessful result with training complete.

```javascript
// Get status
var solutionName = 'ml_x_snc_global_global_classification_solution';
var mlSolution = sn_ml.ClassificationSolutionStore.get(solutionName);
var trainingStatus = mlSolution.getLatestVersion().getStatus();

gs.print(JSON.stringify(JSON.parse(trainingStatus), null, 2));
```

Output:

```
{
  "state": "solution_error",
  "percentComplete": "100",
  "hasJob Ended": "true"
}
```

### ClassificationSolutionVersion - getTrainingStatistics()

Gets all the training statistics for a given solution.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JavaScript object with statistics for each class included in training and values for each class that was excluded from training. For details, see <a href="#">Predictive model components</a>.</td>
</tr>
</tbody>
</table>

```javascript
{
  "includedClasses": Object,
  "excludedClasses": Object
}
```
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.includedClasses</td>
<td>Object containing precision, coverage, recall, and distribution values of each class in the trained model.</td>
</tr>
</tbody>
</table>

```
"includedClasses": {
"<class name>": {
  "distribution": "String",
  "statistics": [Array]
},
```

Data type: Object.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.includedClasses.&lt;class name&gt;.distribution</td>
<td>Value of the outcome probability distribution for this class.</td>
</tr>
</tbody>
</table>

Data type: String

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.includedClasses.&lt;class name&gt;.statistics</td>
<td>List of objects providing training statistics data. For more information, see Configuring target metrics.</td>
</tr>
</tbody>
</table>

```
[
  {
    "precision": "String",
    "coverage": "String",
    "recall": "String",
    "selected": "String"
  }
]
```

Data type: Array

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.includedClasses.&lt;class name&gt;.statistics.precision</td>
<td>Estimated solution precision metric for this class.</td>
</tr>
</tbody>
</table>

Data type: String (numerical value)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.includedClasses.&lt;class name&gt;.statistics.coverage</td>
<td>Estimated solution coverage metric for this class.</td>
</tr>
</tbody>
</table>

Data type: String (numerical value)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.includedClasses.&lt;class name&gt;.statistics.recall</td>
<td>Estimated solution recall metric for this class.</td>
</tr>
</tbody>
</table>

Data type: String (numerical value)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.includedClasses.&lt;class name&gt;.statistics.selected</td>
<td>Flag that indicates whether this class contains records included in solution training.</td>
</tr>
</tbody>
</table>

Valid values:
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• true: This class contains records selected for training the solution.</td>
</tr>
<tr>
<td></td>
<td>• false: This class is excluded from training.</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Boolean value)</td>
</tr>
</tbody>
</table>

`<Object>.excludedClasses`  
Distribution and row count of all the classes excluded from training.

```json
"excludedClasses": {
  "<class name>": {
    "distribution": "0.11",
    "rowCount": "9"
  }
}
```

Data type: Object.

`<Object>.excludedClasses.<class name>.distribution`  
Value of the outcome probability distribution for this class.

Data type: String

`<Object>.excludedClasses.<class name>.rowCount`  
Number of rows excluded from training.

Data type: String (numerical value)

The following example shows how to get training statistics from the active solution.

```javascript
// Get training stats
var mlSolution = sn_ml.ClassificationSolutionStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getTrainingStatistics()), null, 2));
```

Output:

```json
{
  "includedClasses": {
    "Approvals": {
      "distribution": "0.43",
      "statistics": [
      
    }
```
ClassificationSolutionVersion - getVersionNumber()

Gets the version number of a solution object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Version number.</td>
</tr>
</tbody>
</table>

The following example shows how to get a version number.

```javascript
// Get version number
var mlSolution = sn_ml.ClassificationSolutionStore.get('ml_incident_categorization');

gs.print("Version number: "+JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getVersionNumber()), null, 2));
```

Output:

Version number: 1

`ClassificationSolutionVersion - predict(Object input, Object options)`

Gets the input data for a prediction.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>Object</td>
<td>GlideRecord or array of JSON objects containing field names and values as key-value pairs.</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional values for filtering prediction results.</td>
</tr>
<tr>
<td>options.apply_threshold</td>
<td></td>
<td>Flag that indicates whether to check the threshold value for the solution and apply it to the result set. Valid values:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>• true: Return results in which confidence is greater than threshold.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Return all results.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: True</td>
</tr>
<tr>
<td>options.top_nNumber</td>
<td>Number</td>
<td>If provided, returns the top results, up to the specified number of predictions.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JSON object containing the prediction results sorted by sys_id or record_number.</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&lt;identifier&gt;: [Array]</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;</td>
<td>List of objects with details for each prediction result.</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&lt;identifier&gt;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;confidence&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;predictedSysId&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;predictedValue&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;threshold&quot;: Number</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;</td>
<td>Value of the confidence associated with the prediction. For example, 53.84.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;</td>
<td>The sys_id of the predicted sys_id. Results can be from any table on which information is being predicted.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;</td>
<td>Value representing the predicted value.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td>&lt;Object&gt;.&lt;identifier&gt;.&lt;object&gt;.threshold</td>
</tr>
<tr>
<td>Value of the configured threshold associated with the prediction.</td>
<td></td>
</tr>
<tr>
<td>Data type: Number</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to display prediction results for a `predict()` method that takes a GlideRecord by `sys_id` for input and includes optional parameters to restrict to top three results and exclude the threshold value.

```javascript
var mlSolution = sn_ml.ClassificationSolutionStore.get('ml_incident_categorization');

// single GlideRecord input
define input = new GlideRecord("incident");
input.get("<sys_id>");

// configure optional parameters
var options = {};
options.top_n = 3;
options.apply_threshold = false;

var results = mlSolution.getVersion(1).predict(input, options);
// pretty print JSON results
gs.print(JSON.stringify(JSON.parse(results), null, 2));
```

**Output:**

```json
{
  "<sys_id/gr>": [
    {
      "confidence": 62.10782320780268,
      "threshold": 20.36,
      "predictedValue": "Clone Issues",
      "predictedSysId": ""
    },
    {
      "confidence": 6.945237375770391,
      "threshold": 16.63,
      "predictedValue": "Instance Administration",
      "predictedSysId": ""
    }
  ]
}
```
The following example shows how to display prediction results for a `predict()` method that takes an array of field names as key-value pairs for input and includes optional parameters to restrict to top three results and exclude the threshold value.

```javascript
var mlSolution = sn_ml.ClassificationSolutionStore.get('ml_incident_categorization');

// key-value pairs input
var input = ["short_description":"my email is not working"], {short_description:"need help with password"}];

// configure optional parameters
var options = {
  top_n: 3,
  apply_threshold: false
};

var results = mlSolution.predict(input, options);

// pretty print JSON results
gs.print(JSON.stringify(JSON.parse(results), null, 2));
```

Output:

```json
{
  "i": [
    {
      "confidence": 37.5023032262591,
      "threshold": 10.72,
      "predictedValue": "Authentication",
      "predictedSysId": ""
    },
    {
      "confidence": 24.439964862166583,
      "threshold": 23.7,
      "predictedValue": "Administration",
      "predictedSysId": ""
    }
  ]
}
```
ClassificationSolutionVersion - setPredictionSettings(Object options)

Sets precision, coverage, or recall values at solution level or class level.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Metric configuration values. For information about these features, see Configuring target metrics.</td>
</tr>
</tbody>
</table>

```
{  
  "confidence": 11.736320486031047,
  "threshold": 100,
  "predictedValue": "Security",
  "predictedSysId": ""
}  
},  
"2": [  
{  
  "confidence": 99,
  "threshold": 17.77,
  "predictedValue": "Email",
  "predictedSysId": ""
  
},  
{  
  "confidence": 3.182137005157543,
  "threshold": 10.72,
  "predictedValue": "Authentication",
  "predictedSysId": ""
  
},  
{  
  "confidence": 2.8773826570713514,
  "threshold": -1,
  "predictedValue": "Email (I/f)",
  "predictedSysId": ""
  
}  
]  
}
```
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options.metricName</td>
<td>String</td>
<td>Name of the metric to set. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• precision</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• coverage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• recall</td>
</tr>
<tr>
<td>options.metricValue</td>
<td>String</td>
<td>Numeric value to assign to the metric.</td>
</tr>
<tr>
<td>options.className</td>
<td>String</td>
<td>Optional. Name of the class to restrict results to. Use the getTrainingStatistics() method to retrieve a complete list of classes for a solution.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a precision metric to 89.5 for a class named Clone Issues.

```javascript
var mlSolution = 
    sn_ml.ClassificationSolutionStore.get('ml_x_snc_global_global_classification');
var input = {
    "metricName" : "precision", 
    "metricValue" : "89.5", 
    "className" : "Clone Issues"
};
mlSolution.getActiveVersion().setTrainingStatistics(input);
```

**ClassificationSolution - Global**

Scriptable object used in Predictive Intelligence stores.

The ClassificationSolution API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the sn_ml namespace.

The solution setup-to-training flow is as follows:

1. Create a dataset using the DatasetDefinition API.
2. Optional. Build an encoder using the Encoder API.
3. Use the constructor to create a classification solution object.
4. Add the solution object to the classification solution store using the `ClassificationSolutionStore - add()` method.

5. Train the solution using the `submitTrainingJob()` method. This creates a version of the object that you can manage using the `ClassificationSolutionVersion` API.


⚠ **Note:** This API runs with full privileges. To restrict user access, include an access control mechanism in the script.

For usage guidelines, refer to Using ML APIs.

**ClassificationSolution - ClassificationSolution(Object config)**

Creates a classification solution.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>Object</td>
<td>JavaScript object containing configuration properties of the solution.</td>
</tr>
<tr>
<td>config.algorithmConfig</td>
<td>Object</td>
<td>JavaScript object containing algorithm configuration properties.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| config.algorithmConfig.algorithm | String        | Method for encoding your solution. Possible values:  
- **xgboost**: XGBoost encoding to optimize the training.  
- **logisticRegression**: Method using the logistic regression model for categorical targets such as nominal or ordinal. |
| config.algorithmConfig.targetClassRecall | String        | Applies a class recall parameter to steer a solution's training to bias a specific class. Format is 
"<ClassName:RecallValue>"  
where the recall value is a number between 0 and 100 representing a percentage.  
For example, to set and apply this solution parameter to 90% accuracy for all records you train in the Email class, the value is set to "Email:90". |
| config.dataset               | Object        | DatasetDefinition name.                                                                                                                      |
| config.domainName            | String        | Optional. Domain name associated with this dataset. See Domain separation and Predictive Intelligence.  
Default: Current domain, for example, "global". |
| config.encoder               | Object        | Optional. Trained encoder object to assign to this solution. See Encoder - Encoder(Object config).  

Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config.groupByFieldName</td>
<td>String</td>
<td>Optional. Field name by which the system groups records to build classification solutions. For usage information, see Using group by for classification.</td>
</tr>
<tr>
<td>config.inputFieldNames</td>
<td>Array</td>
<td>List of input field names as strings. The model uses these fields used to make predictions.</td>
</tr>
<tr>
<td>config.label</td>
<td>String</td>
<td>Identifies the prediction task.</td>
</tr>
<tr>
<td>config.minRowCount</td>
<td>String</td>
<td>Optional. Minimum number of records required in the dataset for training. Default: 10000</td>
</tr>
<tr>
<td>config.predictedFieldName</td>
<td>String</td>
<td>Identifies a field to be trained for predictability.</td>
</tr>
<tr>
<td>config.processingLanguage</td>
<td>String</td>
<td>Optional. Processing language in two-letter ISO 639-1 language code format. Default: &quot;en&quot;</td>
</tr>
<tr>
<td>config.stopwords</td>
<td>Array</td>
<td>Optional. Preset list of strings that the system automatically generates based on the language property setting. For details, see Create a custom stopwords list. Default: English Stopwords</td>
</tr>
<tr>
<td>config.trainingFrequency</td>
<td>String</td>
<td>The frequency to retrain the model. Possible values: every_30_days, every_60_days, every_90_days</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• every_120_days</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• every_180_days</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• run_once</td>
<td></td>
<td>Default: run_once</td>
</tr>
</tbody>
</table>

The following example shows how to create an object and add it to the ClassificationSolution store.

```javascript
var myData = new sn_ml.DatasetDefinition(
    {
        'tableName': 'incident',
        'fieldNames': ['category', 'short_description', 'priority'],
        'fieldDetails': [
            {
                'name': 'category',
                'type': 'nominal'
            },
            {
                'name': 'short_description',
                'type': 'text'
            }
        ],
        'encodedQuery': 'activeANYTHING'
    });

var mySolution = new sn_ml.ClassificationSolution(
    {
        'label': 'my solution definition',
        'dataset': myData,
        'predictedFieldName': 'category',
        'inputFieldNames': ['short_description']
    });

var myClassificationName = sn_ml.ClassificationSolutionStore.add(mySolution);
```

**ClassificationSolution - cancelTrainingJob()**

Cancels a job for a solution object that has been submitted for training.
The following example shows how to cancel an existing training job.

```javascript
var mySolution =
    sn_ml.ClassificationSolutionStore.get('ml_sn_global_global_classification');
mySolution.cancelTrainingJob();
```

**ClassificationSolution - getActiveVersion()**

Gets the active `ClassificationSolutionVersion` object.

The following example shows how to get an active `ClassificationSolution` version from the store and return its training status.

```javascript
var mlSolution =
    sn_ml.ClassificationSolutionStore.get('ml_x_snc_global_global_classification');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getStatus()), null, 2));
```

Output:

```json
{
    "state": "solution_complete",
}
ClassificationSolution - getAllVersions()

Gets all versions of a classification solution.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Existing versions of a solution object. See also ClassificationSolutionVersion API.</td>
</tr>
</tbody>
</table>

The following example shows how to get all ClassificationSolution version objects and call the getVersionNumber() and getStatus() solution version methods on them.

```javascript
var mlSolution = 
    sn_ml.ClassificationSolutionStore.get('ml_x_snc_global_global_classification');

var mlSolutionVersions = mlSolution.getAllVersions();

for (i = 0; i < mlSolutionVersions.length; i++) {
    gs.print("Version " + mlSolutionVersions[i].getVersionNumber() + " Status: " + 
        mlSolutionVersions[i].getStatus() + 
    
};
```

Output:

Version 3 Status: 
({"state":"solution_complete","percentComplete":"100","hasJobEnded":"true"})

Version 2 Status: 
({"state":"solution_complete","percentComplete":"100","hasJobEnded":"true"})

Version 1 Status: {"state":"solution_cancelled","percentComplete":"0","hasJobEnded":"true"}
ClassificationSolution - getLatestVersion()

Gets the latest version of a solution.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>ClassificationSolutionVersion object corresponding to the latest version of a ClassificationSolution().</td>
</tr>
</tbody>
</table>

The following example shows how to get the latest version of a solution and return its training status.

```javascript
var mlSolution =
    sn_ml.ClassificationSolutionStore.get('ml_x_snc_global_global_classification');

gs.print(JSON.stringify(JSON.parse(mlSolution.getLatestVersion().getStatus()), null, 2));
```

**Output:**

```javascript
{
  "state": "solution_complete",
  "percentComplete": "100",
  "hasJobEnded": "true"
}
```

ClassificationSolution - getName()

Gets the name of the object to use for interaction with the store.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The following example shows how to update `ClassificationSolution` dataset information and print the name of the object.

```javascript
// Update solution
var myIncidentData = new sn_ml.DatasetDefinition({
    'tableName': 'incident',
    'fieldNames': ['category', 'short_description', 'priority'],
    'encodedQuery': 'activeANYTHING'
});

var eligibleFields = JSON.parse(myIncidentData.getEligibleFields('classification'));

var myClassification = new sn_ml.ClassificationSolution({
    'label': "my classification solution",
    'dataset': myIncidentData,
    'inputFieldNames': eligibleFields['eligibleInputFieldNames'],
    'predictedFieldName': 'category'
});

// update solution
sn_ml.ClassificationSolutionStore.update('ml_x_snc_global_global_my_solution_definition_4',
    myClassification);

// print solution name
gs.print('Solution Name: ' + myClassification.getName());
```

**Output:**

```
Solution Name: ml_x_snc_global_global_my_solution_definition_4
```

### ClassificationSolution - getProperties()

Gets solution object properties.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Object | Contents of the Dataset and `ClassificationSolution` object details in the `ClassificationSolutionStore`.

```json
{
    "algorithmConfig": {
        "algorithm": "String",
        "targetClassRecall": "String"
    },
    "datasetProperties": (Object),
    "domainName": "String",
    "encoder": (Object),
    "groupByFieldName": "String",
    "inputFieldNames": [Array],
    "label": "String",
    "name": "String",
    "predictedFieldName": "String",
    "processingLanguage": "String",
    "scope": "String",
    "stopwords": [Array],
    "trainingFrequency": "String"
}
```

**<Object>.algorithmConfig**
Method for encoding the solution.
Data type: Object.

**<Object>.algorithmConfig.algorithm**
Name of the encoding algorithm for training the solution.
Possible values:
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• <strong>xgboost</strong>: XGBoost encoding to optimize training.</td>
<td></td>
</tr>
<tr>
<td>• <strong>logisticRegression</strong>: Method using the logistic regression model for categorical targets such as nominal or ordinal.</td>
<td></td>
</tr>
</tbody>
</table>

**<Object>.algorithmConfig.targetClassRecall**

Class recall parameter to steer a solution's training to bias a specific class. The recall value is a number between 0 and 100 representing a percentage.

Data type: String.

**<Object>.datasetProperties**

Lists the properties of the `DatasetDefinition()` object associated with the solution.

```
{
    "encodedQuery": "String",
    "fieldDetails": [Array],
    "fieldNames": [Array],
    "tableName": "String"
}
```

Data type: Object.

**<Object>.datasetProperties tableName**

Name of the table for the dataset. For example,

```
"tableName" : "Incident"
```

Data type: String.

**<Object>.datasetProperties fieldNames**

List of field names from the specified table as strings. For
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>example, &quot;fieldNames&quot;: {&quot;short_description&quot;, &quot;priority&quot;}.</td>
<td>Data type: Array.</td>
</tr>
<tr>
<td>&lt;Object&gt;.datasetProperties.fieldNames.fieldDetails</td>
<td>List of JavaScript objects that specify field properties.</td>
</tr>
<tr>
<td></td>
<td>[</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>Data type: Array.</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.datasetProperties.fieldNames.fieldDetails.&lt;object&gt;.name</td>
<td>Name of the field defining the type of information to restrict this dataset to.</td>
</tr>
<tr>
<td>Data type: String.</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.datasetProperties.fieldDetails.&lt;object&gt;.type</td>
<td>Machine-learning field type.</td>
</tr>
<tr>
<td>Data type: String.</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.datasetProperties.fieldDetails.encodedQuery</td>
<td>Encoded query string in standard Glide format.</td>
</tr>
<tr>
<td>Data type: String.</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.domainName</td>
<td>Domain name associated with this dataset. See Domain separation and Predictive Intelligence.</td>
</tr>
<tr>
<td>Data type: String.</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.encoder</td>
<td>Encoder object assigned to this solution. See Encoder().</td>
</tr>
<tr>
<td>Data type: Object.</td>
<td></td>
</tr>
<tr>
<td>Returns (continued)</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td><code>&lt;Object&gt;.groupByFieldName</code></td>
<td>Field name by which the system groups records to build classification solutions. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.inputFieldNames</code></td>
<td>List of input field names as strings. The model uses these fields used to make predictions. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.label</code></td>
<td>Identifies the prediction task.</td>
</tr>
<tr>
<td></td>
<td>`{</td>
</tr>
<tr>
<td></td>
<td>&quot;label&quot;: &quot;my first prediction&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.name</code></td>
<td>System-assigned name. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.predictedFieldName</code></td>
<td>Identifies a field to be trained for predictability. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.processingLanguage</code></td>
<td>Processing language in two-letter ISO 639-1 language code format. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.scope</code></td>
<td>Object scope. Currently, the only valid value is <code>global</code>. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.stopwords</code></td>
<td>Optional. Preset list of strings that the system automatically generates based on the <code>language</code> property setting. For details,</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>see Create a custom stopwords list.</td>
<td></td>
</tr>
<tr>
<td>Data type: Array.</td>
<td>The frequency to retrain the model. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• every_30_days</td>
</tr>
<tr>
<td></td>
<td>• every_60_days</td>
</tr>
<tr>
<td></td>
<td>• every_90_days</td>
</tr>
<tr>
<td></td>
<td>• every_120_days</td>
</tr>
<tr>
<td></td>
<td>• every_180_days</td>
</tr>
<tr>
<td>Default: run_once</td>
<td>Data type: String.</td>
</tr>
</tbody>
</table>

The following example gets properties of a solution object in the store.

```javascript
var mySolution = sn_ml.ClassificationSolutionStore.get('ml_sn_global_global_classification_solution');

gs.print(JSON.stringify(JSON.parse(mySolution.getProperties()), null, 2));
```

**Output:**

```json
*** Script: {
  "datasetProperties": {
    "tableName": "incident",
    "fieldNames": ["category", "short_description", "priority", "assignment_group.name"],
    "fieldDetails": [{
      "name": "category",
      "type": "nominal"
    }]
  }
}
```
ClassificationSolution - getVersion(String version)

Gets a solution by provided version number.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Existing version number of a solution.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Specified version of the ClassificationSolution() object on which you can call ClassificationSolutionVersion API methods.</td>
</tr>
</tbody>
</table>

The following example shows how to get the training status of a solution by version number.

```javascript
var mlSolution = 
sn_ml.ClassificationSolutionStore.get('ml_x_snc_global_global_classification');
```
gs.print(JSON.stringify(JSON.parse(mlSolution.getVersion('1').getStatus()), null, 2));

Output:

```javascript
{
  "state": "solution_complete",
  "percentComplete": "100",
  "hasJobEnded": "true"
}
```

**ClassificationSolution - setActiveVersion(String version)**

Activates a specified version of a solution in the store.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Name of the <a href="#">ClassificationSolution()</a> object version to activate. Activating this version deactivates any other version.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to activate a solution version in the store.

```javascript
sn_ml.ClassificationSolution.setActiveVersion("ml_incident_categorization");
```

**ClassificationSolution - submitTrainingJob()**

Submits a training job.

⚠️ **Note:** Before running this method, you must first add a solution to the store using the [ClassificationSolutionStore - add()](#) method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>ClassificationSolutionVersion object corresponding to the ClassificationSolution being trained.</td>
</tr>
</tbody>
</table>

The following example shows how to create a dataset, apply it to a solution, add the solution to a store, and submit the training job.

```javascript
// Create a dataset
var myData = new sn_ml.DatasetDefinition({
    'tableName' : 'incident',
    'fieldNames' : ['assignment_group', 'short_description', 'description'],
    'encodedQuery' : 'activeANYTHING'
});

// Create a solution
var mySolution = new sn_ml.ClassificationSolution({
    'label': "my solution definition",
    'dataset' : myData,
    'predictedFieldName' : 'assignment_group',
    'inputFieldNames': ['short_description']
});

// Add the solution to the store to later be able to retrieve it.
var my_unique_name = sn_ml.ClassificationSolutionStore.add(mySolution);

// Train the solution - this is a long running job
var myClassifierVersion = mySolution.submitTrainingJob();
```

**ClassifierProbes - Global**

Provides a classifier probe launch facility. Encapsulates an array of probe information records, initialized either from a JavaScript array or the XML serialized version.

Use in any server-side script where you need to define a classifier probe launch facility.
**ClassifierProbes - launch()**

Launches the next highest priority classifier from those in the list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if a classifier is launched; otherwise, false.</td>
</tr>
</tbody>
</table>

**ClassifierProbes - launchSupplementary()**

Launches a supplementary probe.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**ClassifierProbes - size()**

Returns the probe length.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The probe length</td>
</tr>
</tbody>
</table>

Client - Scoped, Global

Provides methods to add data to the MetricBase database, to execute transforms on the MetricBase database, and to receive the results of the transforms.

The Client class can be used in scoped and global server scripts. When using the Client class, use the sn_clotho namespace identifier.

This class is part of the MetricBase application.

Scoped Client - Client()

Create an instance of the client class to access the MetricBase database.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var client = new sn_clotho.Client();
```

Scoped Client - deleteSeries(GlideRecord now_GR, String metric)

Remove the data in the MetricBase database associated with the specified metric in the specified GlideRecord. Use this method for removing test data.

Note: This method deletes data from the MetricBase database. There is no recovery mechanism.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>The records whose time series data for the specified metric is to be deleted.</td>
</tr>
<tr>
<td>metric</td>
<td>String</td>
<td>The name of the metric.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var client = new sn_clotho.Client();
// query drones of a specific model
var drones = new GlideRecord("mb_demo_drone");
drones.addQuery("model", "Kingfisher Phantom");
drones.query();
client.deleteSeries(drones, 'mb_demo_mt_speed');
```

Scoped Client - put(Object metricData)

Saves metric data to the MetricBase database.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metricData</td>
<td>Object</td>
<td>One of the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DataBuilder object containing metric data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Array of DataBuilder objects containing metric data.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var time = new GlideDateTime();
// two different GlideRecord instances and metrics
var dataBuilder = new sn_clotho.DataBuilder(now_GR, 'cpu_percentage');
dataBuilder.add(time, 0.6);
var dataBuilder2 = new sn_clotho.DataBuilder(gr2, 'disk_free_percentage');
dataBuilder2.add(time, 0.2);
new sn_clotho.Client().put([dataBuilder, dataBuilder2]);
```

ClusteringSolution - Global

Scriptable object used in Predictive Intelligence stores.
The ClusteringSolution API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the sn_ml namespace.

The solution setup-to-training flow is as follows:

1. Create a dataset using the DatasetDefinition API.
2. Mandatory if using the K-means clustering algorithm. Build an encoder using the Encoder API.
3. Use the constructor to create a clustering solution object.
4. Add the solution object to the clustering solution store using the ClusteringSolutionStore - add() method.
5. Train the solution using the submitTrainingJob() method. This creates a version of the object that you can manage using the ClusteringSolutionVersion API.

Note: This API runs with full privileges. To restrict user access, include an access control mechanism in the script.

For usage guidelines, refer to Using ML APIs.

ClusteringSolution - ClusteringSolution(Object config)

Creates a cluster solution.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>Object</td>
<td>JavaScript object containing configuration properties of the solution.</td>
</tr>
</tbody>
</table>

```json
{
"algorithmConfig": Object,
"clusterConcept": "String",
"clusterConceptFieldNames": [Array],
"dataset": Object,
"domainName": "String",
"encoder": Object,
"groupByFieldName": "String",
"groupUnclusteredRecords": Boolean,
"inputFieldNames": [Array],
"label": "String",
```
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>&quot;maxTimeWindowForUpdate&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;minRecordsPerCluster&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;minRowCount&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;processingLanguage&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;stopwords&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;trainingFrequency&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;updateFrequency&quot;: &quot;String&quot;</td>
</tr>
</tbody>
</table>

**config.algorithmConfig**

Object

Required unless setting the encoder property. JavaScript object containing algorithm configuration properties. Property settings vary by the value set in the algorithm property.

```javascript
'algorithmConfig': { 
  "algorithm": "String", 
  // See algorithmConfig.algorithm setting description for property settings based on algorithm }
```

**config.algorithmConfig.algorithm**

String

Method for encoding your solution. Valid values:

- **dbscan** – Density-Based Spatial Clustering of Applications with Noise (DBSCAN) clustering algorithm. Properties used with this algorithm:
  - **distanceMetric**
  - **epsilon**
  - **minimumNeighbours**
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>hdbscan</em></td>
<td>– Hierarchical Density Based Spatial Clustering of Applications with Noise (HDBSCAN) clustering algorithm. Properties used with this algorithm:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ <em>minimumSamples</em></td>
<td></td>
</tr>
<tr>
<td><em>kmeans</em></td>
<td>– K-means clustering algorithm. Default. The <em>targetCoverage</em> property is used with this algorithm.</td>
<td></td>
</tr>
</tbody>
</table>

Some users prefer DBSCAN because it doesn’t require you to specify the number of clusters in the data before clustering.

Properties for *dbscan*:

```
'algorithConfig': {
    "algorithm": "dbscan",
    "distanceMetric": "String",
    "epsilon": Number,
    "minimumNeighbours": Number
}
```

Properties for *hdbscan*:

```
'algorithConfig': {
    "algorithm": "hdbscan",
    "minimumSamples": Number
}
```

Properties for *kmeans*:

```
'algorithConfig': {
    "algorithm": "kmeans",
    "targetCoverage": Number
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>config.algorithmConfig.distanceMetric</code></td>
<td>String</td>
<td>DBSCAN algorithm only. Distance metric to scan for similar data objects. Valid values: <code>levenshteinDistance</code></td>
</tr>
<tr>
<td><code>config.algorithmConfig.epsilon</code></td>
<td>Number</td>
<td>DBSCAN algorithm only. Decimal value between 0 and 1 representing the size of the neighborhood search radius.</td>
</tr>
<tr>
<td><code>config.algorithmConfig.minimumNeighbours</code></td>
<td>Number</td>
<td>DBSCAN algorithm only. Minimum number of neighbors required in a point to be a part of a cluster. For <code>levenshteinDistance</code> the value must be 1 so that no points are excluded from the dataset.</td>
</tr>
<tr>
<td><code>config.algorithmConfig.minimumSamples</code></td>
<td>Number</td>
<td>Minimum number of data samples in a neighborhood required to determine if a point is a core point. Default: None</td>
</tr>
<tr>
<td><code>config.algorithmConfig.targetCoverage</code></td>
<td>Number</td>
<td>K-means algorithm only. Percentile field to filter out records that are less similar to each other.</td>
</tr>
<tr>
<td><code>config.clusterConcept</code></td>
<td>String</td>
<td>Optional. Concept type. A concept is a set of words listed in descending order of frequency. To generate a TFIDF-based cluster concept, set the value to <code>tfidf</code>. Concept types are listed in the Clustering Definitions [ml_capability_definition_clustering] table.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>config.clusterConceptFieldNames</td>
<td>Array</td>
<td>Optional. List of cluster concept field names. These values are external columns for creating a cluster concept and not used for cluster solution training. If external columns are provided, those columns are only used for the cluster concept and not for clustering solution training. Cluster concept fields are listed in the Clustering Definitions [ml_capability_definition_clustering] table. Default: Input text columns generate the cluster concept</td>
</tr>
<tr>
<td>config.dataset</td>
<td>Object</td>
<td>DatasetDefinition object name.</td>
</tr>
<tr>
<td>config.domainName</td>
<td>String</td>
<td>Optional. Domain name associated with this dataset. See Domain separation and Predictive Intelligence. Default: Current domain, for example, &quot;global&quot;.</td>
</tr>
<tr>
<td>config.encoder</td>
<td>Object</td>
<td>Required unless setting the algorithmConfig property to &quot;levenshteinDistance&quot;. Trained encoder object to assign to this solution. See Encoder - Encoder(Object config).</td>
</tr>
<tr>
<td>config.groupByFieldName</td>
<td>String</td>
<td>Optional. Field name by which the system groups records into one or more clusters.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>In the following setup example, the system groups each type into an individual cluster, rendering 10 clusters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>groupByFieldName</strong> value is 'category'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>DatasetDefinition</strong> <strong>tableName</strong> value is 'incident'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Incident [incident] table has 10 category types</td>
</tr>
<tr>
<td>config.groupUnclusteredRecord</td>
<td>Boolean</td>
<td>Flag that indicates whether to group unclustered records in results.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Group unclustered records separately in results.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not group unclustered records in results. Unclustered values (-1) display with the rest of the results.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>config.inputFieldNames</td>
<td>Array</td>
<td>List of input field names as strings. The model uses these fields used to make predictions.</td>
</tr>
<tr>
<td>config.label</td>
<td>String</td>
<td>Identifies the prediction task.</td>
</tr>
<tr>
<td>config.maxTimeWindowForUpdate</td>
<td>Number</td>
<td>Optional. Number of minutes preceding the model update point to look for records. For example, if the value is 15, the system only looks for records created in the preceding 15</td>
</tr>
</tbody>
</table>
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config.minRecordsPerCluster</td>
<td>Number</td>
<td>Optional. Minimum number of records to allow in any cluster. The value must be greater than or equal to 2. Default: 2</td>
</tr>
<tr>
<td>config.minRowCount</td>
<td>String</td>
<td>Optional. Minimum number of records required in the dataset for training. Default: 10000</td>
</tr>
<tr>
<td>config.processingLanguage</td>
<td>String</td>
<td>Processing language in two-letter ISO 639-1 language code format.</td>
</tr>
<tr>
<td>config.stopwords</td>
<td>Array</td>
<td>Optional. Preset list of strings that the system automatically generates based on the <code>language</code> property setting. For details, see <a href="#">Create a custom stopwords list</a>. Default: English Stopwords</td>
</tr>
<tr>
<td>config.trainingFrequency</td>
<td>String</td>
<td>The frequency to retrain the model. Possible values: every_30_days, every_60_days, every_90_days, every_120_days, every_180_days, run_once. Default: run_once</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config.updateFrequency</td>
<td></td>
<td>The frequency at which the model for the solution definition must be rebuilt.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• do_not_update</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_1_day</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_1_hour</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_6_hours</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_12_hours</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_1_minute</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_15_minutes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_30_minutes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: do_not_update</td>
</tr>
</tbody>
</table>

The following example shows how to create an object and add it to the ClusteringSolution store. The example also shows how to submit the object for training.

```javascript
try{
  try{
    var myData = new sn_ml.DatasetDefinition({
      'tableName' : 'incident',
      'fieldNames' : ['category', 'short_description', 'state', 'description'],
      'encodedQuery' : 'activeANYTHING'
    });

    // get a trained encoder from the store
    var myEncoder = sn_ml.EncoderStore.get('<encoder_name >');

    var mySolution = new sn_ml.ClusteringSolution({
      'label': "clustering solution",
      'dataset' : myData,
      'encoder' : myEncoder,
      'inputFieldNames':["short_description"],
      'groupByFieldName' : 'category',
      'algorithmConfig' : {
        'algorithm' : 'kmeans',
      }
    });
  }
}
```
'targetCoverage' : '90'
});

// add solution
var solutionName = sn_ml.ClusteringSolutionStore.add(mySolution);
var solutionVersion = mySolution.submitTrainingJob();
var trainingStatus = solutionVersion.getStatus();
gs.print(JSON.stringify(JSON.parse(trainingStatus), null, 2));
} catch(ex){
    gs.print('Exception caught: ' + ex.getMessage());
}

Output:
{
    "state": "waiting_for_training",
    "percentComplete": "0",
    "hasJobEnded": "false"
}

The following example shows how to include the 'description' field as a cluster concept field.

var myIncidentData = new sn_ml.DatasetDefinition({
    'tableName' : 'incident',
    'fieldNames' : ['category', 'short_description', 'description'],
});

var encodersolutionName = sn_ml.EncoderStore.get('<encoder_name >');

var mySolution = new sn_ml.ClusteringSolution({
    'label': 'clustering_test',
    'dataset': myIncidentData,
    'inputFieldNames': ['short_description'],
    'encoder': encodersolutionName,
    'clusterConceptFieldNames': ['description']
});

var solutionNameFromStore = sn_ml.ClusteringSolutionStore.add(mySolution);
var myClassifier = mySolution.submitTrainingJob();

**ClusteringSolution - cancelTrainingJob()**

Cancels a job for a solution object that has been submitted for training.
The following example shows how to cancel an existing training job.

```javascript
var mySolution = sn_ml.ClusteringSolutionStore.get('ml_sn_global_global_clustering');
mySolution.cancelTrainingJob();
```

**ClusteringSolution - getActiveVersion()**

Gets the active ClusteringSolutionVersion object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to get an active ClusteringSolution version from the store and return its training status.

```javascript
var mlSolution = sn_ml.ClusteringSolutionStore.get('ml_x_snc_global_global_clustering');
gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getStatus()), null, 2));
```

**Output:**

```json
{
    "state": "solution_complete",
    "percentComplete": "100",
}
ClusteringSolution - getAllVersions()

Gets all versions of a clustering solution.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Existing versions of a solution object. See also ClusteringSolutionVersion API.</td>
</tr>
</tbody>
</table>

The following example shows how to get all ClusteringSolution version objects and call the getVersionNumber() and getStatus() solution version methods on them.

```javascript
var mlSolution = sn_ml.ClusteringSolutionStore.get('ml_x_snc_global_global_clustering');
var mlSolutionVersions = mlSolution.getAllVersions();
for (i = 0; i < mlSolutionVersions.length; i++) {
    gs.print("Version "+ mlSolutionVersions[i].getVersionNumber() + " Status: "+
    mlSolutionVersions[i].getStatus() +"\n");
};
```

**Output:**

Version 3 Status:
{"state":"solution_complete","percentComplete":"100","hasJobEnded":"true"}

Version 2 Status:
{"state":"solution_complete","percentComplete":"100","hasJobEnded":"true"}

Version 1 Status: 
{"state":"solution_cancelled","percentComplete":"0","hasJobEnded":"true"}

ClusteringSolution - getLatestVersion()

Gets the latest version of a solution.
The following example shows how to get the latest version of a solution and return its training status.

```javascript
var mlSolution = sn_ml.ClusteringSolutionStore.get('ml_x_snc_global_global_clustering');
gs.print(JSON.stringify(JSON.parse(mlSolution.getLatestVersion().getStatus()), null, 2));
```

Output:

```
{
    "state": "solution_complete",
    "percentComplete": "100",
    "hasJobEnded": "true"
}
```

**ClusteringSolution - getName()**

Gets the name of the object to use for interaction with the store.

The following example shows how to update ClusteringSolution dataset information and print the name of the object.
// Update solution
var myIncidentData = new sn_ml.DatasetDefinition({
    'tableName': 'incident',
    'fieldNames': ['category', 'short_description', 'priority'],
    'encodedQuery': 'activeANYTHING'
});

var eligibleFields = JSON.parse(myIncidentData.getEligibleFields('clustering'));

var myCluster = new sn_ml.ClusteringSolution({
    'label': "my clustering solution",
    'dataset': myIncidentData,
    'inputFieldNames': eligibleFields['eligibleInputFieldNames'],
    'predictedFieldName': 'category'
});

// update solution
sn_ml.ClusteringSolutionStore.update('ml_x_snc_global_global_clustering_solution', myCluster);

// print solution name
gs.print('Solution Name: ' + myCluster.getName());

Output:
Solution Name: ml_x_snc_global_global_clustering_solution

**ClusteringSolution - getProperties()**

Gets solution object properties.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Contents of the Dataset and ClusteringSolution() object details in the ClusteringSolutionStore</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;algorithmConfig&quot;:</td>
</tr>
<tr>
<td></td>
<td>{Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;datasetProperties&quot;:</td>
</tr>
<tr>
<td></td>
<td>{Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;domainName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;encoder&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;groupByFieldName&quot;:</td>
</tr>
<tr>
<td></td>
<td>&quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;inputFieldNames&quot;:</td>
</tr>
<tr>
<td></td>
<td>[Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;label&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;minRecordsPerCluster&quot;:</td>
</tr>
<tr>
<td></td>
<td>Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;processingLanguage&quot;:</td>
</tr>
<tr>
<td></td>
<td>&quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;scope&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;stopwords&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;trainingFrequency&quot;:</td>
</tr>
<tr>
<td></td>
<td>&quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;updateFrequency&quot;:</td>
</tr>
<tr>
<td></td>
<td>&quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

<Object>.algorithmConfig

JavaScript object containing algorithm configuration properties. Property results vary by the value set in the algorithm property.

'algorithmConfig' : ( |
  "algorithm": "String", |

  // See algorithmConfig.algorithm setting description for
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| <Object>.algorithmConfig.algorithm | Method for encoding your solution. Properties for **dbscan**:

```
'algorithmConfig': {
    "algorithm": "dbscan",
    "distanceMetric": "String",
    "epsilon": Number,
    "minimumNeighbours": Number
}
```

Data type: **Object**.

Properties for **kmeans**:

```
'algorithmConfig': {
    "algorithm": "kmeans",
    "targetCoverage": Number
}
```

Data type: **String**.

<table>
<thead>
<tr>
<th>&lt;Object&gt;.algorithmConfig.distanceMetric</th>
<th>DBSCAN algorithm only. Distance metric to scan for similar data objects. Data type: <strong>String</strong>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.algorithmConfig.epsilon</td>
<td>DBSCAN algorithm only. Decimal value between 0 and 1 representing the size of the neighborhood search radius. Data type: <strong>Number</strong>.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.algorithmConfig.minimumNeighbours</code></td>
<td>DBSCAN algorithm only. Minimum number of neighbors required in a point to be a part of a cluster. For <code>levenshteinDistance</code> the value must be 1 so that no points are excluded from the dataset. Data type: Number.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.algorithmConfig.targetCoverage</code></td>
<td>K-means algorithm only. Percentile field to filter out records that are less similar to each other. Data type: Number.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.datasetProperties</code></td>
<td>Lists the properties of the <code>DatasetDefinition</code> object associated with the solution. Data type: Object.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.datasetProperties.tableName</code></td>
<td>Name of the table for the dataset. For example, &quot;tableName&quot;: &quot;Incident&quot;. Data type: String.</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| `<Object>.datasetProperties.fieldNames` | List of field names from the specified table as strings. For example, 
"fieldNames":
["short_description", "priority"].
Data type: Array. |
| `<Object>.datasetProperties.fieldNames.fieldDetails` | List of JavaScript objects that specify field properties.

```javascript
[
  {
    "name": "String",
    "type": "String"
  }
]
```
Data type: Array. |
| `<Object>.datasetProperties.fieldNames.fieldDetails.<object>.name` | Name of the field defining the type of information to restrict this dataset to.
Data type: String. |
| `<Object>.datasetProperties.fieldDetails.<object>.type` | Machine-learning field type.
Data type: String. |
| `<Object>.datasetProperties.fieldDetails.encodedQuery` | Encoded query string in standard Glide format. See Encoded query strings.
Data type: String. |
| `<Object>.domainName` | Domain name associated with this dataset. See Domain |
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;.encoderProperties</code></td>
<td>Encoder object assigned to this solution. See Encoder - Encoder(Object config). Data type: Object.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.groupByFieldName</code></td>
<td>Field name by which the system groups records into one or more clusters. Data type: String</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.inputFieldNames</code></td>
<td>List of input field names as strings. The model uses these fields used to make predictions. Data type: String</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.label</code></td>
<td>Identifies the prediction task. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.minRecordsPerCluster</code></td>
<td>Minimum number of records to allow in any cluster. Data type: Number.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.name</code></td>
<td>System-assigned name. Data type: String.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.predictedFieldName</code></td>
<td>Identifies a field to be trained for predictability. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.processingLanguage</code></td>
<td>Processing language in two-letter ISO 639-1 language code format. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.scope</code></td>
<td>Object scope. Currently the only valid value is <code>global</code>. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.stopwords</code></td>
<td>Optional. Preset list of strings that the system automatically generates based on the <code>language</code> property setting. For details, see Create a custom stopwords list. Data type: Array.</td>
</tr>
</tbody>
</table>
| `<Object>.trainingFrequency` | The frequency to retrain the model. Possible values:  
  - `every_30_days`  
  - `every_60_days`  
  - `every_90_days`  
  - `every_120_days`  
  - `every_180_days`  
  - `run_once`  
  Default: `run_once`  
 Data type: String. |
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;.updateFrequency</code></td>
<td>The frequency at which the model for the solution definition must be rebuilt. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• do_not_update</td>
</tr>
<tr>
<td></td>
<td>• every_1_day</td>
</tr>
<tr>
<td></td>
<td>• every_1_hour</td>
</tr>
<tr>
<td></td>
<td>• every_6_hours</td>
</tr>
<tr>
<td></td>
<td>• every_12_hours</td>
</tr>
<tr>
<td></td>
<td>• every_1_minute</td>
</tr>
<tr>
<td></td>
<td>• every_15_minutes</td>
</tr>
<tr>
<td></td>
<td>• every_30_minutes</td>
</tr>
<tr>
<td></td>
<td>Default: do_not_update</td>
</tr>
<tr>
<td></td>
<td>Datatype: String</td>
</tr>
</tbody>
</table>

The following example gets properties of a solution object in the store.

```javascript
var myCluster = new sn_ml.ClusteringSolutionStore.get("ml_x_snc_global_global_clustering_solution");
gs.print(JSON.stringify(JSON.parse(myCluster.getProperties()), null, 2));
```

**Output:**

```json
*** Script: {
  "algorithmConfig": {
    "algorithm": "kmeans",
    "targetCoverage": "90"
  },
  "datasetProperties": {
    "tableName": "incident",
    "fieldNames": [
      "category",
      "short_description"
    ]
}
```
"state",
"description"
],
"encodedQuery": "activeANYTHING"
],
"domainName": "global",
"encoderProperties": {
"datasetsProperties": [
{
"tableName": "incident",
"fieldNames": [
"assignment_group",
"short_description",
"description"
],
"encodedQuery": "activeANYTHING"
}
],
"domainName": "global",
"label": "my encoder definition",
"name": "ml_x_snc_global_global_my_encoder_definition",
"processingLanguage": "en",
"scope": "global",
"stopwords": [
"Default English Stopwords"
],
"trainingFrequency": "run_once"
],
"groupByFieldName": "category",
"inputFieldNames": [
"short_description"
],
"label": "clustering solution",
"minRecordsPerCluster": 2,
"name": "ml_x_snc_global_global_clustering_solution",
"processingLanguage": "en",
"scope": "global",
"stopwords": [
"Default English Stopwords"
],
"trainingFrequency": "run_once",
"updateFrequency": "do_not_update"}
}
ClusteringSolution - getVersion(String version)

Gets a solution by provided version number.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Existing version number of a solution.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Specified version of the ClusteringSolution() object on which you can call ClusteringSolutionVersion API methods.</td>
</tr>
</tbody>
</table>

The following example shows how to get the training status of a solution by version number.

```javascript
var mlSolution = sn_ml.ClusteringSolutionStore.get('ml_x_snc_global_global_clustering');
gs.print(JSON.stringify(JSON.parse(mlSolution.getVersion('1').getStatus()), null, 2));
```

Output:

```json
{
  "state": "solution_complete",
  "percentComplete": "100",
  "hasJobEnded": "true"
}
```

ClusteringSolution - setActiveVersion(String version)

Activates a specified version of a solution in the store.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Name of the ClusteringSolution() object version to activate. Activating this version deactivates any other version.</td>
</tr>
</tbody>
</table>
The following example shows how to activate a solution version in the store.

```javascript
sn_ml.ClusteringSolution.setActiveVersion("ml_incident_categorization");
```

**ClusteringSolution - submitTrainingJob()**

Submits a training job.

**Note:** Before running this method, you must first add a solution to the store using the `ClusteringSolutionStore - add()` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td><code>ClusteringSolutionVersion</code> object corresponding to the <code>ClusteringSolution</code> being trained.</td>
</tr>
</tbody>
</table>

The following example shows how to create a dataset, apply it to a solution, add the solution to a store, and submit the training job.

```javascript
// Create a dataset
var myData = new sn_ml.DatasetDefinition({
  'tableName': 'incident',
  'fieldNames': ['assignment_group', 'short_description', 'description'],
  'encodedQuery': 'activeANYTHING'
});

// get a trained encoder from the store
var myEncoder = sn_ml.EncoderStore.get('ml_x_snc_global_global_encoder');

// Create a solution
```
```javascript
var mySolution = new sn_ml.ClusteringSolution(

  'label': "my solution definition",
  'dataset': myData,
  'encoder': myEncoder,
  'predictedFieldName': 'assignment_group',
  'inputFieldNames': ['short_description']

);

// Add the solution to the store to later be able to retrieve it.
var my_unique_name = sn_ml.ClusteringSolutionStore.add(mySolution);

// Train the solution - this is a long running job
var myClusterVersion = mySolution.submitTrainingJob();
```

---

**ClusteringSolutionStore - Global**

Enables storing and retrieving solutions.

The ClusteringSolutionStore API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the sn_ml namespace.

**ClusteringSolutionStore - add(Object mlSolution)**

Adds a new solution object to the store and returns a unique name.

⚠️ **Note:** Label values do not need to be unique. For example, if you run this method with the same label 10 times, this method adds 10 different uniquely-named objects to the store.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mlSolution</td>
<td>ClusteringSolution()</td>
<td>object to add to the store.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>System-generated solution name.</td>
</tr>
</tbody>
</table>
The following example shows how to add a solution to the store. Use `ClusteringSolution - submitTrainingJob()` to run the training job after adding it to the store.

```javascript
// Create a dataset
var myData = new sn_ml.DatasetDefinition({
    'tableName': 'incident',
    'fieldNames': ['assignment_group', 'short_description', 'description'],
    'encodedQuery': 'activeANYTHING'
});

// get a trained encoder from the store
var myEncoder = sn_ml.EncoderStore.get('ml_x_snc_global_global_encoder');

// Create a solution
var mySolution = new sn_ml.ClusteringSolution({
    'label': 'my solution definition',
    'dataset': myData,
    'encoder': myEncoder,
    'predictedFieldName': 'assignment_group',
    'inputFieldNames': ['short_description']
});

// Add the solution to the store to later be able to retrieve it.
var my_unique_name = sn_ml.ClusteringSolutionStore.add(mySolution);
```

**ClusteringSolutionStore - deleteObject(String name)**

Removes a specified solution object from the store.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the ClusteringSolution() object to be deleted.</td>
</tr>
</tbody>
</table>
The following example shows how to delete a solution from the store.

```java
sn_ml.ClusteringSolutionStore.deleteObject("ml_sn_global_global_solution");
```

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**ClusteringSolutionStore - get(String name)**

Gets a solution object from the store.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of a solution in the store.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>ClusteringSolution object. Returns an error if the object does not exist.</td>
</tr>
</tbody>
</table>

The following example shows how to get a solution object from the store using the `get()` method and view its training status using the `ClusteringSolution - getActiveVersion()` and `ClusteringSolutionVersion - getStatus()` methods.

```java
// Get status
var mlSolution = sn_ml.ClusteringSolutionStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getStatus(), null, 2)));  
```

**Output:**

```json
{
  "state":"solution_complete",
  "percentComplete":"100",
  "hasJobEnded":"true"
}
```

**ClusteringSolutionStore - getAllNames(Object options)**

Gets the names of all solution definition records in the store.

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Options for restricting results within the specified properties.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;label&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;domainName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;scope&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>};</td>
</tr>
<tr>
<td>options.label</td>
<td>String</td>
<td>Optional. Label of your solution object.</td>
</tr>
<tr>
<td>options.domainName</td>
<td>String</td>
<td>Optional. Name of the domain for your solution object. Refer to Domain separation and Predictive Intelligence.</td>
</tr>
<tr>
<td>options.scope</td>
<td>String</td>
<td>Optional. Name of an application scope for your solution object.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of strings representing solution object names in the store.</td>
</tr>
</tbody>
</table>

In the following example, the `getAllNames()` method returns a list of all names in the store.

```javascript
gs.print(JSON.stringify(JSON.parse(sn_ml.ClusteringSolutionStore.getAllNames()), null, 2));
```

Output:

```
[
  "ml_x_snc_global_global_clustering_solution_1",
  "ml_x_snc_global_global_clustering_solution"
]
```

In the following example, the `getAllNames()` method returns only names associated with values set in the `options` parameter.

```javascript
var options = {
  'label' : 'my solution definition',
  'domainName' : 'global',
};
```
'scope' : 'global' 
};

var solNames = sn_ml.ClusteringSolutionStore.getAllNames(options);
gs.print(JSON.stringify(JSON.parse(solNames), null, 2));

Output:

```javascript
[
  "ml_x_snc_global_global_my_solution_definition"
]
```

**ClusteringSolutionStore - update(String name, Object mlSolution)**

Replaces an existing object in the store with the object passed as a parameter. The object name provided must be empty or match.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the solution to update.</td>
</tr>
<tr>
<td>mlSolution</td>
<td>ClusteringSolution()</td>
<td>object properties to update.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to update a solution object in the store.

```javascript
var solutionUpdate = new sn_ml.ClusteringSolution({
  'label': 'my solution definition',
  'dataset': myData,
  'predictedFieldName': 'assignment_group',
  'inputFieldNames': ['short_description']
});

sn_ml.ClusteringSolutionStore.update('ml_sn_global_global_incident_service',
  solutionUpdate);
```

**ClusteringSolutionVersion - Global**

Scriptable object used in Predictive Intelligence stores.
The ClusteringSolutionVersion API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the sn_ml namespace.

This API is used for working with solution versions based on ClusteringSolution API objects in the ClusteringSolution store.

The system creates a solution version each time you train a solution definition. Most versions are created during scheduled solution training.

Methods in this API are accessible using the following ClusteringSolution methods:

- `getActiveVersion()`
- `getAllVersions()`
- `getLatestVersion()`
- `getVersion()`

ClusteringSolutionVersion - `cancelUpdateJob()`
Cancels an update job on a trainer.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

The following example shows how to cancel an active training job that has been submitted using the `submitTrainingJob()` method.

```javascript
var myCluster = new sn_ml.ClusteringSolutionStore.get("ml_x_snc_global_global_clustering_solution");
var mlSolutionVersion = myCluster.getActiveVersion();
mlSolutionVersion.cancelUpdateJob();
```

ClusteringSolutionVersion - `deleteClusterAssignments(Object options)`
Deletes cluster assignments from rows by position in table sequence or GlideDateTime.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Object defining one option for deleting cluster assignments.</td>
</tr>
<tr>
<td>Note:</td>
<td></td>
<td>Only one delete option is valid.</td>
</tr>
<tr>
<td>options.updatedUntil</td>
<td>String</td>
<td>Deletes rows for clusters with updated_since values occurring before this value. Format must be provided as GlideDateTime.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{ &quot;updatedUntil&quot; : &quot;String&quot; }</td>
</tr>
<tr>
<td>options.sequenceUntil</td>
<td>String</td>
<td>Deletes rows for clusters with insert_sequence values occurring before this value in the ML Cluster Detail [ml_cluster_detail] table. The sequence position starting point value is 1.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{ &quot;sequenceUntil&quot; : Number }</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of rows deleted from the ML Cluster Detail [ml_cluster_detail] table.</td>
</tr>
</tbody>
</table>

The following example shows how to delete rows for clusters with updated_since time preceding '2020-06-28 02:50:53'.

```javascript
var mlSolution =
    sn_ml.ClusteringSolutionStore.get("ml_x_snc_global_global_clustering_solution");

var mlSolutionVersion = mlSolution.getActiveVersion();

var options = {};
options.updatedUntil = '2020-06-28 02:50:53';

var results = mlSolutionVersion.deleteClusterAssignments(options);

gs.print("Number of deleted rows: "+results);
```

**Output:**

Number of deleted rows: 6417
The following example shows how to delete rows for clusters with updated_since that are sequentially positioned prior to 1000.

```javascript
var mlSolution = sn_ml.ClusteringSolutionStore.get("ml_x_snc_global_global_clustering_solution");

var mlSolutionVersion = mlSolution.getActiveVersion();

var options = {};
options.sequenceUntil = 1000;

var results = mlSolutionVersion.deleteClusterAssignments(options);

gs.print("Number of deleted rows: "+results);
```

Output:

Number of deleted rows: 999

**ClusteringSolutionVersion - getClusterAssignments(Object options)**

Gets assignments for a clustering solution.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Values to use to narrow down the returned results by group, level, system update times, and table row number within a clustering solution. Minimum of one argument must be specified.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>options.group_by</td>
<td>String</td>
<td>Optional. Identifies the segmentation field for which to retrieve cluster memberships, for example, assignment_group.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field provides the same grouping as options provided in the <strong>Use Group By</strong> check box in the <strong>Clustering Definition</strong> form. The information provided varies based on the table selected in the <strong>Table</strong> field. For more information, see Create and train a clustering solution.</td>
</tr>
<tr>
<td>options.limit</td>
<td>Number</td>
<td>Optional. Maximum number of cluster assignments to retrieve.</td>
</tr>
<tr>
<td>options.sequenceSince</td>
<td>Number</td>
<td>Starting position in a table sequence. Returns information for clusters with insert_sequence enabled and positioned after this value in the ML Cluster Detail [ml_cluster_detail] table. The sequence position starting point value is 1.</td>
</tr>
<tr>
<td>options.sequenceUntil</td>
<td>Number</td>
<td>Returns assignments for clusters with insert_sequence values occurring before this value in the ML Cluster Detail [ml_cluster_detail] table. The sequence position starting point value is 1.</td>
</tr>
<tr>
<td>options.topNPerCluster</td>
<td>Number</td>
<td>Number of top results to receive for each cluster.</td>
</tr>
<tr>
<td>options.updatedSince</td>
<td>String</td>
<td>Optional. Date and time. Returns information for clusters with sys_updated_on after the value provided in GlideDateTime.</td>
</tr>
<tr>
<td>options.updatedUntil</td>
<td>String</td>
<td>Optional. Date and time. Returns information for clusters with sys_updated_on before the value provided in GlideDateTime.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of objects containing cluster information in increasing order by cluster_id.</td>
</tr>
<tr>
<td></td>
<td>[</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|               | "cluster_id": "String",
|               | "group_by": "String",
|               | "rec_display_id": "String",
|               | "rec_sys_id": "String"
|
*<object>.cluster_id* Unique cluster number within a solution of clusters.

*<object>.group_by* Grouped, name of the segmentation field associated with this cluster.

*<object>.rec_display_id* Record type and number.

*<object>.rec_sys_id* Record sys_id.

The following example shows how to use the *options* object to filter clustering results.

```javascript
var mlSolution = sn_ml.ClusteringSolutionStore.get("solution_name");

var mlSolutionVersion = mlSolution.getActiveVersion();

var options = {};
options.clusterId = 56;
options.topNPerCluster = 10;
options.updatedUntil = '2020-01-17 23:16:14';
options.updatedSince = '2020-01-17 23:16:13';
options.sequenceUntil = 1000;
options.sequenceSince = 1100;
options.limit = 100;

var results = mlSolutionVersion.getClusterAssignments(options);

gs.print(results);
```

Output:

```javascript
[{
  "cluster_id": "1",
  "rec_display_id": "Incident",
  "INC0014483", "rec_sys_id": "04e33e7adb401300864adfea5e961900", "group_by": "network"},
{
  "cluster_id": "1",
  "rec_display_id": "Incident",
  "INC0011133", "rec_sys_id": "5bd23af2db401300864adfea5e96194d", "group_by": "network"}
]```
ClusteringSolutionVersion - getClusterInfo(Object options)

Gets information for a specified clustering solution in the store. The purity measurement provides insights as a percentage for the clustering fields on which the purity is based.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional. Object containing properties that enable filtering results within a clustering solution.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;clusterId&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;groupBy&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;limit&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;recSysId&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;sequenceSince&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;sequenceUntil&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;updatedSince&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;updatedUntil&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Return cluster memberships for all clusters.</td>
</tr>
<tr>
<td>options.groupBy</td>
<td>String</td>
<td>Optional. Identifies the segmentation field for which to retrieve cluster memberships, for example, assignment_group.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field provides the same grouping as options provided in the <strong>Use Group By</strong> check box in the <strong>Clustering Definition</strong> form. The information provided varies based on the table selected in the <strong>Table</strong> field. For more information, see <a href="#">Create and train a clustering solution</a>.</td>
</tr>
<tr>
<td>options.limit</td>
<td>Number</td>
<td>Optional. Maximum number of cluster assignments to retrieve.</td>
</tr>
<tr>
<td>options.recSysId</td>
<td>String</td>
<td>Optional. Sys_id of a record by which to find cluster information.</td>
</tr>
</tbody>
</table>
| options.sequenceSince | Number| Starting position in a table sequence. Returns information for clusters with insert_sequence
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options.sequenceUntil</td>
<td>String</td>
<td>Optional. Ending position in a table sequence. Returns assignments for clusters with <code>insert_sequence</code> values occurring before this value in the ML Cluster Detail [ml_cluster_detail] table. The sequence position starting point value is 1.</td>
</tr>
<tr>
<td>options.updatedSince</td>
<td>String</td>
<td>Optional. Date and time. Returns information for clusters with <code>sys_updated_on</code> after the value provided in GlideDateTime format.</td>
</tr>
<tr>
<td>options.updatedUntil</td>
<td>String</td>
<td>Optional. Date and time. Returns information for clusters with <code>sys_updated_on</code> before the value provided in GlideDateTime.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of objects containing cluster information.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;cluster_concept&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;cluster_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;cluster_quality&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;cluster_size&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;group_by&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;insert_sequence&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;purity&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_updated_on&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

<object>.cluster_concept

Set of words that describe the cluster in descending order of frequency.
Data type: String

<object>.cluster_id

Unique cluster number within a solution of clusters.
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;object&gt;.cluster_quality</td>
<td>Number from 0 to 100. Higher numbers indicate higher cluster density. Data type: String</td>
</tr>
<tr>
<td>&lt;object&gt;.cluster_size</td>
<td>Number of records in a cluster. Data type: String</td>
</tr>
<tr>
<td>&lt;object&gt;.group_by</td>
<td>If grouped, name of the segmentation field associated with this cluster. Data type: String</td>
</tr>
<tr>
<td>&lt;object&gt;.insert_sequence</td>
<td>Sequential position number of the cluster in the Cluster Summary [ml_cluster_summary] table. Data type: String</td>
</tr>
<tr>
<td>&lt;object&gt;.purity</td>
<td>Percentage value representing the purity of cluster quality.</td>
</tr>
<tr>
<td>&lt;object&gt;.sys_updated_on</td>
<td>System GlideDateTime value representing the date and time at which this cluster was last updated. Data type: String</td>
</tr>
</tbody>
</table>

The following example shows how to set the options object parameter and print the filtered cluster results.

```javascript
var mlSolution = sn_ml.ClusteringSolutionStore.get("ml_x_snc_global_global_clustering_solution");

var mlSolutionVersion = mlSolution.getActiveVersion();

var options = {
  updatedSince: '2020-05-28 02:09:53',
  updatedUntil: '2020-05-28 03:15:00',
  sequenceSince: 1,
  limit: 10,
};

var results = mlSolutionVersion.getClusterInfo(options);

gs.print(JSON.stringify(JSON.parse(results), null, 2));
```

Output:
[  
  {  
    "cluster_id": "1",  
    "cluster_quality": "100",  
    "cluster_size": "17",  
    "purity": "",  
    "insert_sequence": "8",  
    "group_by": "VPN Customer",  
    "sys_updated_on": "2020-05-28 02:09:53",  
    "cluster_concept": "vpn instance connection ldap user log unable usability tunnel"  
  },  
  {  
    "cluster_id": "1",  
    "cluster_quality": "100",  
    "cluster_size": "10",  
    "purity": "",  
    "insert_sequence": "24",  
    "group_by": "Live Feed",  
    "sys_updated_on": "2020-05-28 02:09:53",  
    "cluster_concept": "feed live user note work disable group default usability sort"  
  },  
  {  
    "cluster_id": "1",  
    "cluster_quality": "100",  
    "cluster_size": "18",  
    "purity": "",  
    "insert_sequence": "40",  
    "group_by": "Integrations",  
    "sys_updated_on": "2020-05-28 02:09:53",  
    "cluster_concept": "integrate usability certificate error"  
  }  
]

### ClusteringSolutionVersion - getProperties()

Gets solution object properties and version number.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Contents of the Dataset and <strong>Clusteringsolution</strong> version details. Results vary by object property setup.</td>
</tr>
</tbody>
</table>

```
{
    "algorithmConfig": {Object},
    "datasetProperties": {Object},
    "domainName": "String",
    "encoder": {Object},
    "groupByFieldName": "String",
    "inputFieldNames": [Array],
    "isActive": "Boolean",
    "label": "String",
    "minRecordsPerCluster": Number,
    "name": "String",  "processingLanguage": "String",
    "scope": "String",
    "stopwords": [Array],
    "trainingFrequency": "String",
    "updateFrequency": "String",
    "versionNumber": "Number"
}
```

**<Object>.algorithmConfig**

JavaScript object containing algorithm configuration properties. Property results vary by the value set in the **algorithm** property.

```
'algorithimConfig' : {
    "algorithm": "String",
    // See algorithmConfig.algorithm setting description for property settings based on algorithm
}
```

Data type: Object.

**<Object>.algorithmConfig.algorithm**

JavaScript object containing your solution. Properties for **dbscan**:

```
'algorithimConfig': {
    "algorithm": "dbscan",
    "distanceMetric": "String",
    "epsilon": Number,
    "minimumNeighbours": Number
}
```
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Properties for **kmeans:** | 'algorithmConfig': {   
|                       |   "algorithm": "kmeans",  
|                       |   "targetCoverage": Number                                                                                         |
|                       | Data type: String.                                                                                                   |
| <Object>.algorithmConfig.distanceMetric | DBSCAN algorithm only. Distance metric to scan for similar data objects. Data type: String.                       |
| <Object>.algorithmConfig.epsilon | DBSCAN algorithm only. Decimal value between 0 and 1 representing the size of the neighborhood search radius. Data type: Number. |
| <Object>.algorithmConfig.minimumNeighbours | DBSCAN algorithm only. Minimum number of neighbors required in a point to be a part of a cluster. For levenshteinDistance, the value must be 1 so that no points are excluded from the dataset. Data type: Number. |
| <Object>.algorithmConfig.targetCoverage | K-means algorithm only. Percentile field to filter out records that are less similar to each other. Data type: Number. |
| <Object>.datasetProperties | properties of the DatasetDefinition object associated with the solution.                                          |
|                       | {   
|                       |   "encodedQuery": "String",  
|                       |   "fieldDetails": [Array],  
|                       |   "fieldNames": [Array],  
|                       |   "tableName": "String"                                                                                         |
|                       | Data type: Object.                                                                                                   |
| <Object>.datasetProperties.tableName | Name of the table for the dataset. For example, "tableName" : "Incident". Data type: String. |

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.datasetProperties.fieldNames</td>
<td>List of field names from the specified table as strings. For example, &quot;fieldNames&quot; : [&quot;short_description&quot;, &quot;priority&quot;]. Data type: Array.</td>
</tr>
<tr>
<td>&lt;Object&gt;.datasetProperties.fieldDetails</td>
<td>List of JavaScript objects that specify field properties.</td>
</tr>
<tr>
<td>{ }</td>
<td></td>
</tr>
<tr>
<td>&quot;name&quot;: &quot;String&quot;, &quot;type&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td>Data type: Array.</td>
</tr>
<tr>
<td>&lt;Object&gt;.datasetProperties.fieldNames.fieldDetails.&lt;object&gt;.name</td>
<td>Name of the field defining the type of information to restrict this dataset to. Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.datasetProperties.fieldNames.fieldDetails.&lt;object&gt;.type</td>
<td>Machine-learning field type. Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.domainName</td>
<td>Domain name associated with this dataset. See Domain separation and Predictive Intelligence. Data type: String.</td>
</tr>
<tr>
<td>Object.encoderProperties</td>
<td>Encoder object assigned to this solution. See Encoder - Encoder(Object config). Data type: Object.</td>
</tr>
<tr>
<td>Object.groupByNameField</td>
<td>Field name by which the system groups records into one or more clusters. Data type: String</td>
</tr>
<tr>
<td>Object.inputFieldNames</td>
<td>List of input field names as strings. The model uses these fields used to make predictions.</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.isActive</td>
<td>Flag that indicates whether this version is active.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Version is active.</td>
</tr>
<tr>
<td></td>
<td>• false: Version is not active.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.label</td>
<td>Identifies the prediction task.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;label&quot;: &quot;my first prediction&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.minRecordsPerCluster</td>
<td>Number of records to allow in any cluster.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>&lt;Object&gt;.name</td>
<td>System-assigned name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.processingLanguage</td>
<td>Language in two-letter ISO 639-1 language code format.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.scope</td>
<td>Object scope. Currently the only valid value is global.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.stopwords</td>
<td>Optional. Preset list of strings that the system automatically generates</td>
</tr>
<tr>
<td></td>
<td>based on the language property setting.</td>
</tr>
<tr>
<td></td>
<td>For details, see <a href="#">Create a custom stopwords list</a>.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>&lt;Object&gt;.trainingFrequency</td>
<td>Frequency to retrain the model.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• every_30_days</td>
</tr>
<tr>
<td></td>
<td>• every_60_days</td>
</tr>
<tr>
<td></td>
<td>• every_90_days</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>every_120_days</td>
<td></td>
</tr>
<tr>
<td>every_180_days</td>
<td></td>
</tr>
<tr>
<td>run_once</td>
<td>Default: run_once</td>
</tr>
</tbody>
</table>

Data type: String.

<Object>.updateFrequency

The frequency at which the model for the solution definition must be rebuilt. Possible values:

- do_not_update
- every_1_day
- every_1_hour
- every_6_hours
- every_12_hours
- every_1_minute
- every_15_minutes
- every_30_minutes

Default: do_not_update

Datatype: String

<Object>.versionNumber

Version number of the ClusteringSolution object.

The following example gets properties of the active object version in the store.

```javascript
// Get properties
var mlSolution = 
    sn_ml.ClusteringSolutionStore.get('ml_x_snc_global_global_clustering_solution');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getProperties()), null, 2));
```

**Output:**

```json
*** Script: {
    "algorithmConfig": {
```
"algorithm": "kmeans",
"targetCoverage": "90"
},
"datasetProperties": {
"tableName": "incident",
"fieldNames": [
"category",
"short_description",
"state",
"description"
],
"encodedQuery": "activeANYTHING"
},
"domainName": "global",
"encoderProperties": {
"datasetsProperties": [
{
"tableName": "incident",
"fieldNames": [
"assignment_group",
"short_description",
"description"
],
"encodedQuery": "activeANYTHING"
}
],
"domainName": "global",
"isActive": "true",
"label": "my encoder definition",
"name": "ml_x_snc_global_global_my_encoder_definition",
"processingLanguage": "en",
"stopwords": ["Default English Stopwords"],
"versionNumber": "1"
},
"groupByFieldName": "category",
"inputFieldNames": [
"short_description"
],
"isActive": "true",
"label": "clustering solution",
"minRecordsPerCluster": 2,
"name": "ml_x_snc_global_global_clustering_solution"
ClusteringSolutionVersion - getStatus(Boolean includeDetails)

Gets training completion status.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>includeDetails</td>
<td>Boolean</td>
<td>Flag that indicates whether to return status details. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Return additional details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not return additional details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: False</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JavaScript object containing training status information for a ClusteringSolution object.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;state&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;percentComplete&quot;: &quot;Number as a String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;hasJobEnded&quot;: &quot;Boolean value as a String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;details&quot;: {Object}</td>
</tr>
<tr>
<td>&lt;Object&gt;.state</td>
<td>Training completion state. If the training job reaches a terminal state, the job does not leave that state. If the state is terminal, the hasJobEnded property is set to true. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• fetching_files_for_training</td>
</tr>
<tr>
<td></td>
<td>• preparing_data</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• retry</td>
<td></td>
</tr>
<tr>
<td>• solution_cancelled <em>(terminal)</em></td>
<td></td>
</tr>
<tr>
<td>• solution_complete <em>(terminal)</em></td>
<td></td>
</tr>
<tr>
<td>• solution_error <em>(terminal)</em></td>
<td></td>
</tr>
<tr>
<td>• solution_incomplete</td>
<td></td>
</tr>
<tr>
<td>• training_request_received</td>
<td></td>
</tr>
<tr>
<td>• training_request timed_out <em>(terminal)</em></td>
<td></td>
</tr>
<tr>
<td>• training_solution</td>
<td></td>
</tr>
<tr>
<td>• uploading_solution</td>
<td></td>
</tr>
<tr>
<td>• waiting_for_training</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

<Object>.hasJobEnded

Flag that indicates whether training is complete.

Valid values:
• true: Training is complete.
• false: Training is incomplete.

Data type: Boolean value as a String

<Object>.percentComplete

Number between zero and 100 representing training percent complete. If the completion percentage is less than 100, the job might be in a terminal state. For example, if training times out.

Data type: Number as a String

<Object>.details

Object containing a list of additional training details.

Data type: Object

The following example shows a successful result with training complete.

```javascript
// Get status
var mlSolution =
    sn_ml.ClusteringSolutionStore.get('ml_x_snc_global_global_cluster_solution');
```
The following example shows an unsuccessful result with training complete.

```
// Get status
var solutionName = 'ml_x_snc_global_global_cluster_solution';
var mlSolution = sn_ml.ClusteringSolutionStore.get(solutionName);
var trainingStatus = mlSolution.getLatestVersion().getStatus();
gs.print(JSON.stringify(JSON.parse(trainingStatus), null, 2));
```

Output:

```
{
  "state":"solution_error",
  "percentComplete":"100",
  "hasJobEnded":"true",
}
```

**ClusteringSolutionVersion - getTopNPurityInfo(Object options)**

Gets the top purity results for a clustering solution. The purity measurement provides insights as a percentage for the clustering fields on which the purity is based.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>options</td>
</tr>
</tbody>
</table>
|          |          | [{
  "clusterIds": [Array],
  "groupBy": [Array],
  "purityFields": [Array],
|          |          |                   |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>options.clusterIds</code></td>
<td>Array</td>
<td>Optional. List of cluster IDs provided as strings. A cluster ID is provided in the Cluster Summary [ml_cluster_summary] table. If provided, this method returns purity cluster information for each specified cluster. Default: This method returns purity information for all clusters.</td>
</tr>
<tr>
<td><code>options.groupBy</code></td>
<td>Array</td>
<td>Optional. List of group_by field strings from your table to help the system identify the class which is most frequent in the cluster. If provided, returns purity information for cluster solutions using group_by fields. The Cluster Summary [ml_cluster_summary] table lists clusters and any associated Group by values. This field only applies to clusters that use group by in the clustering definition. Eligible fields are listed in the Table field. For details on the group by feature, refer to Create and train a clustering solution.</td>
</tr>
<tr>
<td><code>options.purityFields</code></td>
<td>Array</td>
<td>Optional. List of purity field strings. If provided, this method only returns information for these purity fields. You can view purity values for a cluster in the Cluster Summary [ml_cluster_summary] table lists. <strong>Note:</strong> If both <code>purity_fields</code> and <code>topN_fields</code> are provided, this method returns <code>topN_fields</code> first, then selects fields from <code>purity_fields</code>. Default: Return purity information for all purity fields saved for the cluster.</td>
</tr>
<tr>
<td><code>options.topN</code></td>
<td>Number</td>
<td>Optional. Restricts the number of predictions to return to the highest values for each purity field. Maximum value: 10</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options.topNFields</td>
<td>Number</td>
<td>Optional. Restricts the number of purity fields returned to fields with the highest purity for each cluster. Maximum value: 10 Default: Return all fields in the cluster.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>JSON-formatted string containing a list of clusters with purity fields and details.</td>
</tr>
</tbody>
</table>

**Note:** Results vary based on settings made in the `options` input parameter.

The following information illustrates how cluster purity results are sorted and categorized. Refer to the example for actual output.

```json
{
    "<clusterID>"[:
        // List sorted by purity per field
        { "<fieldName>"[:
            // List sorted by purity per field value
            {"<field_val1_1>"":"<purity>",
             // Additional field values
             }],
            // Additional fields
         ],
        // Additional clusters
    }
```
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String.&lt;clusterID&gt;.&lt;field&gt; of objects representing purity fields arranged in descending order by purity value per field. The <code>options.top_n_fields</code> input object parameter property determines the number of fields returned.</td>
<td></td>
</tr>
</tbody>
</table>

```
"<field>:{{"<field_val>":"<purity>"}}}
```

Data type: Array

| String.<clusterID>.<field>.<values> of objects containing field value and purity. For example, `{"priority":{{"5":"100"}}}` is a priority field with a score of 5 and a purity value of 100 percent. Results list in descending order by purity percentage. The `options.top_n` input object parameter property determines the number of results returned. |

Data type: Array

The following example shows how to get the top two purity results for the category field in specific cluster solutions.

```javascript
var solution = new GlideRecord('ml_solution');
solution.addQuery('sys_id', '<clustering_solution_sys_id>');
solution.addQuery('active', 'true');
solution.query();

while (solution.next()) {

    var options = {};
    options.clusterIds = ['1', '3', '5'];
    options.purityFields = ['category'];
    options.topN = '2';
    options.topNFields = '2';

    var clustering = new sn_ml.ClusteringSolutionVersion(solution);
    var results = clustering.getTopNPurityInfo(options);

    gs.info(results);
}
```

Output displays purity insights based on the settings provided in the options parameter.
**ClusteringSolutionVersion - getUpdateStatus()**

Gets the status of the most recent clustering solution update job.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>State of the clustering solution update. States:</td>
</tr>
<tr>
<td></td>
<td>• Configuration or network error</td>
</tr>
<tr>
<td></td>
<td>• Error while updating solution</td>
</tr>
<tr>
<td></td>
<td>• Fetching files for updating</td>
</tr>
<tr>
<td></td>
<td>• Preparing data</td>
</tr>
<tr>
<td></td>
<td>• Unauthorized</td>
</tr>
<tr>
<td></td>
<td>• Update complete</td>
</tr>
<tr>
<td></td>
<td>• Updating is cancelled</td>
</tr>
<tr>
<td></td>
<td>• Updating request received</td>
</tr>
<tr>
<td></td>
<td>• Updating request timed out</td>
</tr>
<tr>
<td></td>
<td>• Updating solution</td>
</tr>
<tr>
<td></td>
<td>• Uploading</td>
</tr>
<tr>
<td></td>
<td>• Waiting</td>
</tr>
</tbody>
</table>

The following example shows how to get the update status of a clustering solution.

```javascript
var myCluster = new
  sn_ml.ClusteringSolutionStore.get("ml_x_snc_global_global_clustering_solution");
gs.print(JSON.stringify(myCluster.getActiveVersion().getUpdateStatus()));
```
Output:

"Update Complete"

**ClusteringSolutionVersion - getVersionNumber()**

Gets the version number of a solution object.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Version number.</td>
</tr>
</tbody>
</table>

The following example shows how to get a version number.

```javascript
// Get version number
var mlSolution =
    sn_ml.ClusteringSolutionStore.get('ml_x_snc_global_global_clustering_solution');
gs.print("Version number:
    " + JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getVersionNumber()), null, 2));
```

Output:

Version number: 1

**ClusteringSolutionVersion - predict(Object input, Object options)**

Gets the input data for a prediction.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>input</td>
</tr>
<tr>
<td>options</td>
</tr>
</tbody>
</table>

```javascript
{  
    "apply_threshold": Boolean, 
}
```
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>options.apply_threshold</strong></td>
<td>Boolean</td>
<td>Flag that indicates whether to check the threshold value for the solution and apply it to the result set. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- true: Return results in which confidence is greater than threshold.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- false: Return all results.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: True</td>
</tr>
<tr>
<td><strong>options.top_n</strong></td>
<td>Number</td>
<td>If provided, returns the top results, up to the specified number of predictions.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JSON object containing the prediction results sorted by sys_id or record_number.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&lt;identifier&gt;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;</td>
<td>List of objects with details for each prediction result.</td>
</tr>
<tr>
<td></td>
<td>&lt;identifier&gt;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;confidence&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;predictedSysId&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;predictedValue&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;threshold&quot;: Number</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;.object</td>
<td>object of the confidence associated with the prediction.</td>
</tr>
<tr>
<td></td>
<td>For example, 53.84.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;.&lt;object&gt;.predictedSysId</td>
<td>The sys_id of the predicted value. Results can be from any table on which information is being predicted. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;.&lt;object&gt;.predictedValue</td>
<td>Value representing the prediction result. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;.&lt;object&gt;.threshold</td>
<td>Value of the configured threshold associated with the prediction. Data type: Number</td>
</tr>
</tbody>
</table>

The following example shows how to display prediction results for a `predict()` method that takes a GlideRecord by sys_id for input and includes optional parameters to restrict to top three results and exclude the threshold value.

```javascript
var mlSolution = sn_ml.ClusteringSolutionStore.get('ml_incident_categorization');

// single GlideRecord input
var input = new GlideRecord("incident");
input.get("<sys_id>");

// configure optional parameters
var options = {};
options.top_n = 3;
options.apply_threshold = false;

var results = mlSolution.getVersion(1).predict(input, options);
// pretty print JSON results
gs.print(JSON.stringify(JSON.parse(results), null, 2));
```

```javascript
{
  "<sys_id/gr>": [
    {
      "confidence": 62.10782320780268,
      "threshold": 20.36,
      "predictedValue": "Clone Issues",
      "predictedSysId": ""
    },
    {
      "confidence": 6.945237375770391,
      "threshold": 20.36,
      "predictedValue": "","n
```
The following example shows how to display prediction results for a `predict()` method that takes an array of field names as key-value pairs for input and includes optional parameters to restrict to top three results and exclude the threshold value.

```javascript
var mlSolution = sn_ml.ClusteringSolutionStore.get("ml_incident_categorization");

// key-value pairs input
var input = [{short_description:"my email is not working"}, {short_description:"need help with password"}];

// configure optional parameters
var options = {};
options.top_n = 3;
options.apply_threshold = false;
var results = mlSolution.predict(input, options);

// pretty print JSON results
gs.print(JSON.stringify(JSON.parse(results), null, 2));
```
ClusteringSolutionVersion - submitUpdateJob(Object options)

Submits clustering update jobs with options to narrow results to a specific table and filter for matching records.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>JavaScript object containing options on which to base a clustering solution update.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options.filter</td>
<td>String</td>
<td>Encoded query string in standard Glide format. See Encoded query strings. Enables running an update job based on the filter provided.</td>
</tr>
<tr>
<td>options.table</td>
<td>String</td>
<td>Name of the table on which to run an update job.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to submit an update job.

```javascript
var myCluster = new
  sn_ml.ClusteringSolutionStore.get("ml_x_snc_global_global_clustering_solution");

var options = {
  "filter": "precision",
  "table": "incident"
};

myCluster.getActiveVersion().submitUpdateJob(options);
```

**CMDBDuplicateTaskUtils - Global**

Use the CMDBDuplicateTaskUtils API to create a De-duplication task that contains duplicate CIs that are of independent type.

A de-duplication task can only contain CMDB CIs that are not in another de-duplication task.

**CMDBDuplicateTaskUtils - CMDBDuplicateTaskUtils()**

Creates an instance of the CMDBDuplicateTaskUtils class.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### CMDBDuplicateTaskUtils - `createDuplicateTask(String sysIDs)`

Create a **De-duplication** task in which the sys_ids of duplicate tasks are specified.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysIDs</td>
<td>String</td>
<td>A comma separated list of sys_IDs.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_ID of the created task. Returns null if the input string is empty, the input parameter contains a sys_ID that is not in the cmdb_ci table, or the input parameter contains a sys_ID that is in an open de-duplication task.</td>
</tr>
</tbody>
</table>

```java
// where sys-id1 and sys-id2 are sys_IDs of CIs in the cmdb_ci table
var sysIDs = 'sys-id1, sys-id2';
var dupTaskUtil = new CMDBDuplicateTaskUtils();
var deDupTaskID = dupTaskUtil.createDuplicateTask(sysIDs);
gs.info(deDupTaskID);
```

### CMDBGroupAPI - Scoped

The **CMDBGroupAPI** provides methods for performing actions on Configuration Management Database (CMDB) groups.

The **CMDBGroupAPI** is a scoped static class. To use the class you must include the namespace identifier `sn_cmdbgroup` before the CMDBGroupAPI object. For example:

```java
var response = sn_cmdbgroup.CMDBGroupAPI.getManualCIList(groupSysId, false);
```

To access this API you must have the itil or asset role.
CMDBGroupAPI - getAllCI(String groupId, Boolean requireCompleteSet, Boolean requireAllQueryNodesCis)

Returns all configuration items (CIs) for the specified group. The results include all manual CIs and the list of CIs from the query builder's saved query.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupId</td>
<td>String</td>
<td>Sys_id of the Configuration Management Database (CMDB) group.</td>
</tr>
</tbody>
</table>
| requireCompleteSet    | Boolean       | Optional, but must be passed if `requireAllQueryNodesCis` is passed. Flag that indicates whether an empty string is returned if any CIs are filtered out by access control list (ACL) restrictions. Valid values:  
  • true: Return empty string.  
  • false: Don't return value.  
  Default: false |
| requireAllQueryNodesCis| Boolean       | Optional. Flag that indicates whether to return CIs from all CMDB classes of the query. Valid values:  
  • true: Return CIs from all columns.  
  • false: Only return CIs from the starting node of the query builder query.  
  Default: false |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errors</td>
<td>List of errors for a failed operation. Data type: Array</td>
</tr>
</tbody>
</table>

*errors*: [
  
]
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errors.error</td>
<td>Error name. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• GROUP_SYS_ID_IS_NOT_FOUND</td>
</tr>
<tr>
<td></td>
<td>• GROUP_SYS_ID_IS_EMPTY</td>
</tr>
<tr>
<td></td>
<td>• FAIL_TO_INSERT_GROUP_CI_PAIR</td>
</tr>
<tr>
<td></td>
<td>• FAIL_TO_INSERT_GROUP_QUERY_ID_PAIR</td>
</tr>
<tr>
<td></td>
<td>• CI_CAN_NOT_FOUND</td>
</tr>
<tr>
<td></td>
<td>• SAVED_QUERY_ID_NOT_FOUND</td>
</tr>
<tr>
<td></td>
<td>• ERROR_DURING_QUERY_BUILDER_PROCESS_QUERY</td>
</tr>
<tr>
<td></td>
<td>• TIMEOUT_DURING_QUERY_BUILDER_PROCESS_QUERY</td>
</tr>
<tr>
<td></td>
<td>• NOT_COMPLETE_DURING_QUERY_BUILDER_PROCESS_QUERY</td>
</tr>
<tr>
<td></td>
<td>• MAX_LIMIT_DURING_QUERY_BUILDER_PROCESS_QUERY</td>
</tr>
<tr>
<td></td>
<td>• GROUP_API_TIMEOUT</td>
</tr>
<tr>
<td></td>
<td>• EXCEPTION_FROM_EXECUTE_QUERY</td>
</tr>
<tr>
<td></td>
<td>• SOME_CI_NOT_VISIBLE_DUE_TO_SECURITY_CONSTRAINT</td>
</tr>
<tr>
<td>errors.message</td>
<td>Brief description of the error message. Data type: String</td>
</tr>
<tr>
<td>idList</td>
<td>List of CMDB CI sys_ids. Data type: Array</td>
</tr>
<tr>
<td>partialCIListDueToACLFlag</td>
<td>Flag that indicates if the list of returned CIs is incomplete due to ACL restrictions.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This is not considered an error condition, and no corresponding error information is returned.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: <code>idList</code> is incomplete.</td>
</tr>
<tr>
<td></td>
<td>• false: <code>idList</code> is complete.</td>
</tr>
<tr>
<td>result</td>
<td>Flag that indicates whether the method completed successfully.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Success</td>
</tr>
<tr>
<td></td>
<td>• false: Error</td>
</tr>
</tbody>
</table>

// Script example:
```
var getAllCIFunc = function(groupSysId) {
    var parser = new JSONParser();
    var response = sn_cmdbgroup.CMDBGroupAPI.getAllCI(groupSysId, false);
    var parsed = parser.parse(response);
    if (parsed.result) {
        gs.info("succeed to retrieve ci list: " + parsed.idList);
    } else {
        gs.info("fail to retrieve list, errors: " + JSON.stringify(parsed.errors));
    }
}
var groupExists = "d0d2d25113152200eef2dd828144b0e4";
var groupContainsInvalidSavedQuery = "e685a2c3d7012200de92a5f75e610387";
getAllCIFunc(groupExists);
getAllCIFunc(groupContainsInvalidSavedQuery);
```

Output:
```
// Successful response
{
    'result':true,
    'partialCIListDueToACLFlag': false,
    'idList':[
        '3a5dd3db0a8ce0100655f1ec66ed42c',
        '6b43105c373010000deabfc8bcbe5db2'
    ]
}

// Error response
{
"errors": null,
"partialCIListDueToACLFlag": false,
"idList":null
}
```
CMDBGroupAPI - getAllCIFromQueryBuilder(String groupId, Boolean requireCompleteSet, Boolean requireAllQueryNodesCis)

Returns all configuration items (CIs) returned from all saved query builders' query IDs for the specified group.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupId</td>
<td>String</td>
<td>Sys_id of the Configuration Management Database (CMDB) group.</td>
</tr>
<tr>
<td>requireCompleteSet</td>
<td>Boolean</td>
<td>Optional, but must be passed if requireAllQueryNodesCis is passed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Flag that indicates whether an empty string is returned if any CIs are filtered out by access control list (ACL) restrictions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Return empty string.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Don’t return value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>requireAllQueryNodesCis</td>
<td>Boolean</td>
<td>Optional. Flag that indicates whether to return CIs from all CMDB classes of the query.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Return CIs from all columns.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Only return CIs from the starting node of the query builder query.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errors</td>
<td>List of errors for a failed operation. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>errors.error</td>
<td>Error name. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• GROUP_SYS_ID_IS_NOT_FOUND</td>
</tr>
<tr>
<td></td>
<td>• GROUP_SYS_ID_IS_EMPTY</td>
</tr>
<tr>
<td></td>
<td>• FAIL_TO_INSERT_GROUP_CI_PAIR</td>
</tr>
<tr>
<td></td>
<td>• FAIL_TO_INSERT_GROUP_QUERY_ID_PAIR</td>
</tr>
<tr>
<td></td>
<td>• CI_CAN_NOT_FOUND</td>
</tr>
<tr>
<td></td>
<td>• SAVED_QUERY_ID_NOT_FOUND</td>
</tr>
<tr>
<td></td>
<td>• ERROR_DURING_QUERY_BUILDER_PROCESS_QUERY</td>
</tr>
<tr>
<td></td>
<td>• TIMEOUT_DURING_QUERY_BUILDER_PROCESS_QUERY</td>
</tr>
<tr>
<td></td>
<td>• NOT_COMPLETE_DURING_QUERY_BUILDER_PROCESS_QUERY</td>
</tr>
<tr>
<td></td>
<td>• MAX_LIMIT_DURING_QUERY_BUILDER_PROCESS_QUERY</td>
</tr>
<tr>
<td></td>
<td>• GROUP_API_TIMEOUT</td>
</tr>
<tr>
<td></td>
<td>• EXCEPTION_FROM_EXECUTE_QUERY</td>
</tr>
<tr>
<td></td>
<td>• SOME_CI_NOT_VISIBLE_DUE_TO_SECURITY_CONSTRAINT</td>
</tr>
<tr>
<td>errors.message</td>
<td>Brief description of the error message.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>idList</td>
<td>List of CMDB CI sys_ids. Data type: String</td>
</tr>
<tr>
<td>partialCIListDueToACLFlag</td>
<td>Flag that indicates if the list of returned CIs is incomplete due to ACL restrictions. Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Note: This is not considered an error condition, and no corresponding error information is returned.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: idList is incomplete.</td>
</tr>
<tr>
<td></td>
<td>• false: idList is complete.</td>
</tr>
<tr>
<td>result</td>
<td>Flag that indicates whether the method completed successfully. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Success</td>
</tr>
<tr>
<td></td>
<td>• false: Error</td>
</tr>
</tbody>
</table>

// Script example:
```javascript
var getAllCIFromQueryBuilderFunc = function(groupSysId) {
  var parser = new JSONParser();
  var response = sn_cmdbgroup.CMDBGroupAPI.getAllCIFromQueryBuilder(groupSysId, false);
  var parsed = parser.parse(response);
  if (parsed.result) {
    gs.info("succeed to retrieve ci list: " + parsed.idList);
  } else {
    gs.info("fail to retrieve list, errors: " + JSON.stringify(parsed.errors));
  }
}
var groupExists = "d0d2d25113152200eef2dd828144b0e4";
var groupContainsInvalidSavedQuery = "e685a2c3d7012200de92a5f75e610387";
getAllCIFromQueryBuilderFunc(groupExists);
getAllCIFromQueryBuilderFunc(groupContainsInvalidSavedQuery);
```

Output:
CMDBGroupAPI - getManualCIList(String groupId, Boolean requireCompleteSet)

Returns the CMDB group’s manual CI list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupId</td>
<td>String</td>
<td>The sysId of the CMDB group.</td>
</tr>
<tr>
<td>requireCompleteSet</td>
<td>Boolean</td>
<td>When true, returns an error string if any CIs are filtered out by ACL restrictions.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A JSON formatted string in the format</td>
</tr>
</tbody>
</table>

```json
{
  'result':false,
  'errors':[
    {'message':'Group does not exist','error':'GROUP_SYS_ID_IS_NOT_FOUND'}
  ],
  'partialCIListDueToACLFlag':false,
  'idList':[]
}
```
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>}, 'partialCIListDueToACLFlag':false,</td>
</tr>
<tr>
<td></td>
<td>'idList':['sys_id_1', 'sys_id2'] }</td>
</tr>
</tbody>
</table>

Where

- **result** - a boolean flag. When true the method was successful.
- **errors** - a list of errors with a message and error code.
- **partialCIListDueToACLFlag** - a Boolean flag. When true, the idList is incomplete due to an ACL restriction. When false, the idList is complete.
- **idList** - an array of cmdb_ci sys_ids

When not successful, returns one of the errors

- GROUP_SYS_ID_IS_NOT_FOUND
- GROUP_SYS_ID_IS_EMPTY
- FAIL_TO_INSERT_GROUP_CI_PAIR
- FAIL_TO_INSERT_GROUP_QUERY_ID_PAIR
- CI_CAN_NOT_FOUND
- SAVED_QUERY_ID_NOT_FOUND
- ERROR DURING QUERY_BUILDER_PROCESS_QUERY
- TIMEOUT DURING QUERY_BUILDER_PROCESS_QUERY
- NOT_COMPLETE DURING QUERY_BUILDER_PROCESS_QUERY
- MAX_LIMIT DURING QUERY_BUILDER_PROCESS_QUERY
- GROUP_API_TIMEOUT
- EXCEPTION FROM_EXECUTE_QUERY
- SOME_CI_NOT_VISIBLE DUE TO SECURITY CONSTRAINT

// Script example for requireCompleteSet being false:
```
var getManualCIList = function(groupSysId) {
  var parser = new JSONParser();
  var response = sn_cmdbgroup.CMDBGroupAPI.getManualCIList(groupSysId, false);
  var parsed = parser.parse(response);
  if (parsed.result) {
    gs.info("succeed to retrieve ci list: " + parsed.idList);
  } else {
    gs.info("fail to retrieve list, errors: " + JSON.stringify(parsed.errors));
  }
}
```

// create a group in cmdb_group, and add CIs to this group in Edit Manual CI form
var groupExists = "d0d2d25113152200eef2dd828144b0e4";
// use a non-exist group
var groupDoesNotExist = "d0d2d25113152200eef2dd828144b0e4111";
getManualCIList(groupExists);
getManualCIList(groupDoesNotExists);

Output: (Line breaks added for formatting.)
succeed to retrieve ci
list: 6b43105c37301000deeabfc8bcbe5db2,2dfd7c8437201000deeabfc8bcbe5d56
fail to retrieve list, errors:
[{"message":"Group does not exist","error":"GROUP_SYS_ID_IS_NOT_FOUND"}]

// Script example for requireCompleteSet being true
var getManualCIList = function(groupSysId) {
    var parser = new JSONParser();
    var response = sn_cmdbgroup.CMDBGroupAPI.getManualCIList(groupSysId, true);
    var parsed = parser.parse(response);
    if (parsed.result) {
        gs.info("succeed to retrieve ci list: " + parsed.idList);
    } else {
        gs.info("fail to retrieve list, errors: " + JSON.stringify(parsed.errors));
    }
}

// create a group in cmdb_group, and add CIs to this group in Edit Manual CI form
var groupExists = "d0d2d25113152200eef2dd828144b0e4";
getManualCIList(groupExists);

Output: (Line breaks added for formatting.)
fail to retrieve list, errors:
[{"message":"Some CI(s) not visible due to security constraint","error":"SOME_CI_NOT_VISIBLE_DUE_TO_SECURITY_CONSTRAINT"}]

**CMDBGroupAPI - getSavedQueryIdList(String groupId, Boolean requireCompleteSet)**

Returns the query builder's query IDs for the specified CMDB group.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupId</td>
<td>String</td>
<td>The sysId of the CMDB group.</td>
</tr>
<tr>
<td>requireCompleteSet</td>
<td>Boolean</td>
<td>When true, returns an empty string if any CIs are filtered out by ACL restrictions.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A JSON formatted string in the format</td>
</tr>
</tbody>
</table>

```json
{ 'result':false,
  'errors':[ {'message':'Group does not exist',
              'error':'GROUP_SYS_ID_IS_NOT_FOUND'},
            ],  // another error if it exists
  'partialCIListDueToACLFlag':false,
  'idList':['sys_id_1', 'sys_id2'] }
```

Where

- result - a boolean flag. When true the method was successful.
- errors - a list of errors with a message and error code.
- partialCIListDueToACLFlag - a Boolean flag. When true, the idList is incomplete due to an ACL restriction. When false, the idList is complete.
- idList - an array of cmdb_ci sys_ids

When not successful, returns one of the errors

GROUP_SYS_ID_IS_NOT_FOUND, GROUP_SYS_ID_IS_EMPTY,
FAIL_TO_INSERT_GROUP_CI_PAIR,
FAIL_TO_INSERT_GROUP_QUERY_ID_PAIR,
CI_CAN_NOT_FOUND, SAVED_QUERY_ID_NOT_FOUND,
ERROR_DURING_QUERY_BUILDER_PROCESS_QUERY,
TIMEOUT_DURING_QUERY_BUILDER_PROCESS_QUERY,
NOT_COMPLETE_DURING_QUERY_BUILDER_PROCESS_QUERY,
MAX_LIMIT_DURING_QUERY_BUILDER_PROCESS_QUERY,
GROUP_API_TIMEOUT, EXCEPTION_FROM_EXECUTE_QUERY,
SOME_CI_NOT_VISIBLE_DUE_TO_SECURITY_CONSTRAINT

// Script example:
var getSavedQueryIdList = function(groupSysId) {
  var parser = new JSONParser();
  var response = sn_cmdbgroup.CMDBGroupAPI.getSavedQueryIdList(groupSysId, false);
  var parsed = parser.parse(response);
  if (parsed.result) {
    gs.info("succeed to retrieve saved query id list: " + parsed.idList);
  } else {
    gs.info("fail to retrieve list, errors: " + JSON.stringify(parsed.errors));
}

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
var groupExists = "d0d2d25113152200eef2dd828144b0e4";
var groupDoesNotExists = "d0d2d25113152200eef2dd828144b0e4111";
getSavedQueryIdList(groupExists);
getSavedQueryIdList(groupDoesNotExists);

Output: (Line breaks added for formatting.)
succeed to retrieve saved query id list: 5d498532d7c12200de92a5f75e6103ce
fail to retrieve list, errors:

  [{"message":"Group does not exist","error":"GROUP_SYS_ID_IS_NOT_FOUND"}]

CMDBGroupAPI - setManualCIList(String groupid, String ciSysIds)
Sets the manual CI list for the specified group. The existing manual CI list is overwritten. CI sysIds not found in the cmdb_ci table are ignored.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupId</td>
<td>String</td>
<td>The sysId of the CMDB group.</td>
</tr>
<tr>
<td>ciSysIds</td>
<td>String</td>
<td>Comma separated list of CI sysIds.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A JSON formatted string in the format</td>
</tr>
</tbody>
</table>

```json
{
  'result':false,
  'errors':[ {'message':'Group does not exist',
               'error':'GROUP_SYS_ID_IS_NOT_FOUND'},
              {} // another error if it exists
  ],
  'partialCIListDueToACLFlag':false,
  'idList':['sys_id_1', 'sys_id2']
}
```

Where

- result - a boolean flag. When true the method was successful.
- errors - a list of errors with a message and error code.
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• partialCIListDueToACLFlag - a Boolean flag. When true, the idList is incomplete due to an ACL restriction. When false, the idList is complete.</td>
<td></td>
</tr>
<tr>
<td>• idList - an array of cmdb_ci sys_ids</td>
<td></td>
</tr>
</tbody>
</table>

When not successful, returns one of the errors:
GROUP_SYS_ID_IS_NOT_FOUND, GROUP_SYS_ID_IS_EMPTY, FAIL_TO_INSERT_GROUP_CI_PAIR, FAIL_TO_INSERT_GROUP_QUERY_ID_PAIR, CI_CAN_NOT_FOUND, SAVED_QUERY_ID_NOT_FOUND, ERROR_DURING_QUERY_BUILDER_PROCESS_QUERY, TIMEOUT_DURING_QUERY_BUILDER_PROCESS_QUERY, NOT_COMPLETE_DURING_QUERY_BUILDER_PROCESS_QUERY, MAX_LIMIT_DURING_QUERY_BUILDER_PROCESS_QUERY, GROUP_API_TIMEOUT, EXCEPTION_FROM_EXECUTE_QUERY, SOME_CI_NOT_VISIBLE_DUE_TO_SECURITY_CONSTRAINT

// Script example:
var setManualCIListFunc = function(groupSysId, manualCIList) {
    var parser = new JSONParser();
    var response = sn_cmdbgroup.CMDBGroupAPI.setManualCIList(groupSysId, manualCIList);
    var parsed = parser.parse(response);
    if (parsed.result) {
        gs.info("succeed to set manual ci list");
    } else {
        gs.info("fail to set manual ci list, errors: " + JSON.stringify(parsed.errors));
    }
}
var group = "d0d2d25113152200eef2dd828144b0e4";
var groupDoesNotExist = "1234";
var manualCIList = "b4fd7c8437201000deeabfc8bcbe5dc1, affd3c8437201000deeabfc8bcbe5dc3";
setManualCIListFunc(group, manualCIList);
setManualCIListFunc(groupDoesNotExist, manualCIList);

Output: (Line breaks added for formatting.)
succeed to set manual ci list
fail to set manual ci list, errors: [{"message":"Group does not exist","error": "GROUP_SYS_ID_IS_NOT_FOUND"}]
CMDBGroupAPI - setSavedQueryIdList(String groupld, String queryIds)

Sets the saved query ID list for the specified group. The existing query ID list is overwritten. Query sysIds not found in the qb_saved_query table are ignored.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupld</td>
<td>String</td>
<td>The sysId of the CMDB group.</td>
</tr>
<tr>
<td>queryIds</td>
<td>String</td>
<td>Comma separated list of saved query sysIds.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A JSON formatted string in the format</td>
</tr>
</tbody>
</table>

```
{ 'result':false,
  'errors':[ {'message':'Group does not exist',
    'error':'GROUP_SYS_ID_IS_NOT_FOUND'},
  { } // another error if it exists
},
'partialCIListDueToACLFlag':false,
'idList':['sys_id_1', 'sys_id2'] }
```

Where

- result - a boolean flag. When true the method was successful.
- errors - a list of errors with a message and error code.
- partialCIListDueToACLFlag - a Boolean flag. When true, the idList is incomplete due to an ACL restriction. When false, the idList is complete.
- idList - an array of cmdb_ci sys_ids

When not successful, returns one of the errors

GROUP_SYS_ID_IS_NOT_FOUND, GROUP_SYS_ID_IS_EMPTY,
FAIL_TO_INSERT_GROUP_CI_PAIR,
FAIL_TO_INSERT_GROUP_QUERY_ID_PAIR,
CI_CAN_NOT_FOUND, SAVED_QUERY_ID_NOT_FOUND,
ERROR_DURING_QUERY_BUILDER_PROCESS_QUERY,
TIMEOUT_DURING_QUERY_BUILDER_PROCESS_QUERY,
NOT_COMPLETE_DURING_QUERY_BUILDER_PROCESS_QUERY,
MAX_LIMIT_DURING_QUERY_BUILDER_PROCESS_QUERY,
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GROUP_API_TIMEOUT, EXCEPTION_FROM_EXECUTE_QUERY, SOME_CI_NOT_VISIBLE_DUE_TO_SECURITY_CONSTRAINT</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
// Script example:
var setSavedQueryIdListFunc = function(groupSysId, queryIdList) {
    var parser = new JSONParser();
    var response = sn_cmdbgroup.CMDBGroupAPI.setSavedQueryIdList(groupSysId, queryIdList);
    var parsed = parser.parse(response);
    if (parsed.result) {
        gs.info("succeed to set saved query id list");
    } else {
        gs.info("fail to set saved query id list, errors: " + JSON.stringify(parsed.errors));
    }
}

var group = "d0d2d25113152200eef2dd828144b0e4";
var savedQueryBuilderIdList = "394585fed7812200de92a5f75e61038e";
var savedQueryBuilderIdNotExistList = "b4fd7c8437201000deabfc8bcbe5dc1, affd3c8437201000deabfc8bcbe5dc3";
setSavedQueryIdListFunc(group, savedQueryBuilderIdList);
setSavedQueryIdListFunc(group, savedQueryBuilderIdNotExistList);
```

Output: (Line breaks added for formatting.)

```
succeed to set saved query id list
fail to set saved query id list, errors: [{"message": "Saved query id(b4fd7c8437201000deabfc8bcbe5dc1) is not found for group(d0d2d25113152200eef2dd828144b0e4)"},"error": "SAVED_QUERY_ID_NOT_FOUND"},{"message": "Saved query id(affd3c8437201000deabfc8bcbe5dc3) is not found for group(d0d2d25113152200eef2dd828144b0e4)"},"error": "SAVED_QUERY_ID_NOT_FOUND"]
```

**CMDBQueryBuilderAPI - Global**

Use the `CMDBQueryBuilderAPI` to retrieve Query Builder saved query execution details in server-side scripts.

To access this API you must activate the Configuration Management (CMDB) (com.snc.cmdb) plugin.

For more information on the CMDB Query Builder, see [Querying the CMDB](Querying the CMDB).
CMDBQueryBuilderAPI - getSavedQueryExecutionDetails(String savedQueryName, Boolean executeQuery, Number timeout)

Returns the execution details for a specified Configuration Management Database (CMDB) Query Builder saved query.

Optionally you can specify whether to execute the query or not, in order to return its most recent query execution details. You can also override the default timeout limit for the execution of the query.

In the event of a timeout or memory issue while executing the query, you can fine tune the query batch size to help alleviate these problems. For additional information, see Batch size for Query Builder saved queries.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>executeQuery</td>
<td>Boolean</td>
<td>Optional. Flag that indicates whether to execute the specified query or not.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Execute the specified query and return its execution details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not execute the specified query. Return the most recent execution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>details of the query.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: False</td>
</tr>
<tr>
<td>savedQueryName</td>
<td>String</td>
<td>Name of the CMDB Query Builder saved query for which to return query</td>
</tr>
<tr>
<td></td>
<td></td>
<td>execution details.</td>
</tr>
<tr>
<td>timeout</td>
<td>Number</td>
<td>Optional. Integer value that overrides the default query execution timeout</td>
</tr>
<tr>
<td></td>
<td></td>
<td>limit of five minutes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unit: Seconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Five minutes</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>Error code and message. Returned when the query execution fails or the</td>
</tr>
<tr>
<td></td>
<td>specified query is not found.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query_id</td>
<td>Saved query Execution ID (sys_id). Obtained from the Query Status [qb_query_status] table.</td>
</tr>
<tr>
<td>table_name</td>
<td>Name of the table that contains the execution query details.</td>
</tr>
<tr>
<td>status</td>
<td>Status of the query execution.</td>
</tr>
<tr>
<td>COMPLETE</td>
<td>Execution is complete.</td>
</tr>
<tr>
<td>FAILED</td>
<td>Error occurred while executing query.</td>
</tr>
<tr>
<td>TIME_OUT</td>
<td>Query execution timed out.</td>
</tr>
<tr>
<td>MAX_LIMIT</td>
<td>Maximum number of return results reached.</td>
</tr>
</tbody>
</table>

Error examples:
- "error": "QUERY_EXECUTION_NOT_FOUND", "message": "Query execution details not found. Please execute the query."
- "error": "QUERY_NOT_FOUND", "message": "Given saved query not found. Please make sure you are passing the name of a saved query."
- "error": "QUERY_EXECUTION_FAILED", "message": "Query execution failed. Please use a valid query."

This example shows how to call the `getSavedQueryExecutionDetails()` method.

```java
SNC.CMDBQueryBuilderAPI.getSavedQueryExecutionDetails('Test', true);
```

Output:
```
{"query_id":"024fd53a7773330033b5270bba106141",
"table_name":"u_cmdb_qb_result_598918aacb4dbf3f1a8dce6a5d57151a4e2",
"status":COMPLETE}
```

This example shows how to call the `getSavedQueryExecutionDetails()` method and override the default query execution timeout value to 10 minutes.

```java
SNC.CMDBQueryBuilderAPI.getSavedQueryExecutionDetails('Test', true, 600);
```

Output:
```
{"query_id":"024fd53a7773330033b5270bba106141",
"table_name":"u_cmdb_qb_result_598918aacb4dbf3f1a8dce6a5d57151a4e2",
"status":COMPLETE}
```
CMDBTransformUtil - Global

The CMDBTransformUtil class uses the Identification and Reconciliation (IRE) framework to minimize creation of duplicate configuration items (CI).

The IRE framework also reconciles CI attributes by only accepting information from authorized sources when updating the CMDB in onBefore transform map scripts. This class cannot be used in other scripts. For additional information on how the Identification and Reconciliation framework operates, see CMDB Identification and Reconciliation.

CMDBTransformUtil - CMDBTransformUtil()

Creates a CMDBTransformUtil object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var cmdbUtil = new CMDBTransformUtil();
```

CMDBTransformUtil - getError()

Returns any errors generated by the previous identifyAndReconcile() or identifyAndReconcileEnhanced() method call.

Use this method in onBefore transform map scripts to obtain errors. Use the hasError() method to first detect any errors from the identifyAndReconcile() or identifyAndReconcileEnhanced() method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Error message from identifyAndReconcile() or identifyAndReconcileEnhanced().</td>
</tr>
</tbody>
</table>
// Add this code to the onBefore transform map script
// Call CMDB API to do Identification and Reconciliation of the current row
var cmdbUtil = new CMDBTransformUtil();
cmdbUtil.setDataSource('ImportSet');
cmdbUtil.identifyAndReconcileEnhanced(source, map, log);
ignore = true;

if (cmdbUtil.hasError()) {
    var errorMessage = cmdbUtil.getError();
    log.error(errorMessage);
} else {
    log.info('IE Output Payload: ' + cmdbUtil.getOutputPayload());
    log.info('Imported CI: ' + cmdbUtil.getOutputRecordSysId());
}

Scoped equivalent
There is no scoped equivalent for this method.

CMDBTransformUtil - getOutputPayload()

Returns the JSON payload from the previous identifyAndReconcile() or identifyAndReconcileEnhanced() method call.

Use this method in onBefore transform map scripts.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;</td>
<td>JSON formatted string that is a list of results for the configuration items in the input string. Each result string is in the format 'items: [[]], relations: [[]]', where each item within items and relations lists contains name-value pairs.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedItems</code></td>
<td>No values are currently returned.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations</code></td>
<td>No values are currently returned.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items</code></td>
<td>List of objects that describe created or updated CIs. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"items": {
  "additionalRelatedItems": [Array],
  "className": "String",
  "duplicateIndices": [Array],
  "errorCount": Number,
  "errors": [Array],
  "identificationAttempts": [Array],
  "identifierEntrySysId": "String",
  "info": [Array],
  "inputIndices": [Array],
  "maskedAttributes": [Array],
  "operation": "String",
  "relatedItems": [Array],
  "relatedSysIds": [Array],
  "sysId": "String"
}
```

- `<String>.items.additionalRelatedItems` List of JSON objects that provides information about additional lookup and related items that were processed but not provided as part of the input payload. These items come from partial payloads. This information is not currently returned. Data type: Array
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| "additionalRelatedItems": [  
  {  
    "className": "String",  
    "inputIndices": [Array],  
    "operation": "String",  
    "sysId": "String"  
  }  
] | |
| <String>.items.additionalRelatedItems.className | Class/table name (sys_class_name) of the CI that was created or updated.  
Data type: String |
| <String>.items.additionalRelatedItems.inputIndices | Index of the corresponding input item. For top-level items it is a list of integers. For related or lookup items, it is list of JSON objects.  
Data type: Array of Numbers or Array of objects |
| "inputIndices": [  
  {  
    "mainIndex": Number,  
    "subIndex": Number  
  }  
] | |
| <String>.items.additionalRelatedItems.inputIndices.mainIndex | Index value from the request body items array that corresponds to the CI parent the additional related item.  
Data type: Number |
| <String>.items.additionalRelatedItems.inputIndices.subIndex | Index value from the request body items.lookup array that corresponds to the additional related item.  
Data type: Number |
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| `<String>.items.additionalRelatedItems.operation` | Type of operation. Possible values:  
- INSERT: New CI was inserted into the database.  
- NO_CHANGE: No CI changes were made.  
- UPDATE: Existing CI was updated.  
  Data type: String |
| `<String>.items.additionalRelatedItems.sysId` | Sys_id of the additional related items.  
  Data type: String |
| `<String>.items.className` | Class/table name (sys_class_name) of the CI that was created or updated.  
  Data type: String |
| `<String>.items.duplicateIndices` | List of indexes of CIs that are duplicates of the current item.  
  Data type: Array |
| `<String>.items.errorCount` | Number of errors.  
  Data type: Number |
| `<String>.items.errors` | Array of objects in which each object describes an error encountered while processing this CI.  
  Data type: Array |

```
"errors": [  
  {  
    "error": "String",  
    "message": "String"
  }
]```
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| `<String>.items.errors.error` | Type of error encountered while processing the CI.  
Data type: String |
| `<String>.items.errors.message` | Error message associated with the error.  
Data type: String |
| `<String>.items.identificationAttempts` | List of attempts that were made to identify the CIs.  
Data type: Array |
| `<String>.items.identificationAttempts.attemptResult` | Results of the attempt to identify the CI.  
Possible values: 
- **MATCHED**: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes. 
- **MULTI_MATCH**: Identification failed with an error. Duplicate CIs were found in the identifier rule table when |

© 2021 ServiceNow, Inc. All rights reserved. 
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>matching against the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>- <strong>NO_MATCH</strong>: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>- <strong>SKIPPED</strong>: Identification not attempted. The attributes required for this identifier rule table search were not provided, so the rule was not applied.</td>
</tr>
</tbody>
</table>

Data type: String

`<String>.items.identificationAttempts.attributes` List of CI identifier entry attributes that were used during the identification process. Data type: Array

Attribute names and types depend on the request body data and the identifier in use, such as:

```
"attributes": {
  "serial_number": "String",
  "serial_number_type": "String"
}
```

`<String>.items.identificationAttempts.hybridEntryCiAttributes` No values are currently returned.

`<String>.items.identificationAttempts.identifierName` Identifier rule used for this CI identification attempt. Data type: String

`<String>.items.identificationAttempts.searchOnTable` Name of the table searched during the identification process.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.identifierEntrySysId</code></td>
<td>Sys_id for the identifier rule used to identify the CI. Located in the Identifier Entry [cmdb_identifier_entry] table.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.info</code></td>
<td>List of objects that contain additional information about the processing of the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;info&quot;: [</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.info.code</code></td>
<td>Reclassification type that was skipped. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• SKIPPED_CLASS_SWITCH</td>
</tr>
<tr>
<td></td>
<td>• SKIPPED_CLASS_DOWNGRADE</td>
</tr>
<tr>
<td></td>
<td>• SKIPPED_CLASS_UPGRADE</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.info.message</code></td>
<td>Message that provides additional insights into the reason for skipping the reclassification.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.info.ruleSysId</code></td>
<td>Sys_id of the reclassification restriction rule that was matched. Applicable only when IRE skips reclassification.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.inputIndices</code></td>
<td>Index values for CIs from the request body <code>items</code> array that correspond to this CI. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.maskedAttributes</code></td>
<td>List of attributes whose update by a non-authoritative data source was skipped as defined by the Reconciliation Rules. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems</code></td>
<td>List of JSON objects that provides information about processed related items. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.className</code></td>
<td>Class/table name (sys_class_name) of the related item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.errors</code></td>
<td>List of errors that occurred during processing.</td>
</tr>
</tbody>
</table>

```
"relatedItems": [ 
  
  
  
  
]
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data type: Array</strong></td>
<td></td>
</tr>
<tr>
<td>&quot;errors&quot;: [</td>
<td>Number of errors detected during processing.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Indexes of the corresponding related items.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array or Numbers</td>
</tr>
<tr>
<td>&quot;inputIndices&quot;: [</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Integer value from the request body <strong>items</strong> array that corresponds to the CI parent of the related item.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Integer value from the request body <strong>items.lookup</strong> array that corresponds to the related item.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Type of operation.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| • INSERT: New CI was inserted into the database.  
• NO_CHANGE: No CI changes were made.  
• UPDATE: Existing CI was updated. | |
| <String>.items.relatedSysIds | List of the sys_id values for related items (table lookup items) from the request body items.lookup array.  
Notable values:  
• null: No sys_id was identified for this related item. |
| <String>.items.sys_id | Sys_id of the CI that was updated or created.  
Data type: String |
| <String>.relations | List of JSON objects that provides information about processed relations.  
Data type: Array |
| "relations":[
   {
      "className": "String",
      "errorCount": Number,
      "inputIndices": [Array],
      "operation": "String",
      "sysId": "String"
   }
] | |
| <String>.relations.className | Sys_class_name of this dependent relationship CI. |
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only supported value:</td>
<td></td>
</tr>
<tr>
<td>• cmdb_rel_ci: CI Relationship table.</td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.relations.errorCount</td>
<td>Number of errors.</td>
</tr>
<tr>
<td>Data type: Number</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.relations.inputIndices</td>
<td>Indexes of the corresponding input relations.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.relations.operation</td>
<td>Type of operation performed.</td>
</tr>
<tr>
<td>Possible values:</td>
<td></td>
</tr>
<tr>
<td>• INSERT</td>
<td></td>
</tr>
<tr>
<td>• UPDATE</td>
<td></td>
</tr>
<tr>
<td>• NO_CHANGE</td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.relations.sysId</td>
<td>Sys_id of the dependent relationship CI.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

// Add this code to the onBefore transform map script
// Call CMDB API to do Identification and Reconciliation of the current row
var cmdbUtil = new CMDBTransformUtil();
cmdbUtil.setDataSource('ImportSet');
cmdbUtil.identifyAndReconcileEnhanced(source, map, log);
ignore = true;

if (cmdbUtil.hasError()) {
    var errorMessage = cmdbUtil.getError();
    log.error(errorMessage);
} else {
    log.info('IE Output Payload: ' + cmdbUtil.getOutputPayload());
}
Scoped equivalent

There is no scoped equivalent for this method.

**CMDBTransformUtil - getOutputRecordSysId()**

Returns the sys_id of the configuration item (CI) that was inserted or updated.

Use this method in onBefore transform map scripts.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the CI that was inserted or updated.</td>
</tr>
</tbody>
</table>

// Add this code to the onBefore transform map script
// Call CMDB API to do Identification and Reconciliation of current row
var cmdbUtil = new CMDBTransformUtil();
cmdbUtil.setDataSource('ImportSet');
cmdbUtil.identifyAndReconcileEnhanced(source, map, log);
ignore = true;

if (cmdbUtil.hasError()) {
    var errorMessage = cmdbUtil.getError();
    log.error(errorMessage);
} else {
    log.info('IE Output Payload: ' + cmdbUtil.getOutputPayload());
    log.info('Imported CI: ' + cmdbUtil.getOutputRecordSysId());
}
**CMDBTransformUtil - getPayload(Object source, Object map, Object log)**

Returns the JSON payload generated from the specified input.

You can then modify the payload and use the `setPayload()` method to store the updated payload. Once the payload is modified and stored, use `identifyAndReconcile()` or `identifyAndReconcileEnhanced()` to process the payload.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>Object</td>
<td>Global object available in transform map scripts that holds the incoming record.</td>
</tr>
<tr>
<td>map</td>
<td>Object</td>
<td>Map object available in transform map scripts that holds the current transform map record.</td>
</tr>
<tr>
<td>log</td>
<td>Object</td>
<td>Log object available in transform map scripts.</td>
</tr>
</tbody>
</table>

### Returns

JSON format for the results for the payload.

Data type:

```json
{
  "additionalCommittedItems": [Array],
  "additionalCommittedRelations": [Array],
  "hasError": "Boolean",
  "hasWarning": "Boolean",
  "items": [Array],
  "relations": [Array],
  "summary": {Object}
}
```

**<String>.additionalCommittedItems**

List of CIs that were committed during the IRE processing of the current payload.
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>not present in the current input payload.</td>
<td></td>
</tr>
</tbody>
</table>

Data type:

```json
"additionalCommittedItems": [
  {
    "className": "String",
    "errorCount": Number,
    "operation": "String",
    "identificationAttempts": [Array],
    "inputIndices": [Array],
    "markers": [Array],
    "mergedPayloads": [Array],
    "sysId": "String"
  }
]```

### <String>.additionalCommittedItems.className

Sys_class name of this additional CI.

**Data type:** String

### <String>.additionalCommittedItems.errorCount

Number of errors encountered while processing this additional CI.

**Data type:** Number

### <String>.additionalCommittedItems.errors

Array of objects that describes errors encountered while processing this additional CI.

**Data type:** Array

```json
"errors": [
  {
    "error": "String",
    "message": "String"
  }
]```

### <String>.additionalCommittedItems.errors.error

Type of error encountered while processing the additional CI.

**Data type:** String
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.errors.message</td>
<td>Error message encountered while processing the additional CI. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.identificationAttempts</td>
<td>Array of objects in which each object describes an attempt made to identify this additional CI. Data type: Array</td>
</tr>
</tbody>
</table>

```
"identificationAttempts": [
  {
    "attemptResult": "String",
    "attributes": [Array],
    "hybridEntryCiAttributes": [Array],
    "identifierName": "String",
    "searchOnTable": [Array]
  }
]
```

<table>
<thead>
<tr>
<th>&lt;String&gt;.additionalCommittedItems.identificationAttempts.attemptResult</th>
<th>Outcome of this additional CI identification attempt. Data type: String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Possible values:</td>
<td></td>
</tr>
<tr>
<td>• MATCHED: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.</td>
<td></td>
</tr>
<tr>
<td>• MULTI_MATCH: Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.</td>
<td></td>
</tr>
<tr>
<td>• NO_MATCH: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.identificationAttempts.attributes</td>
<td>Array of CI identifier entry attributes used during this additional CI identification attempt. Data type: Array. Attribute names and types depend on the request body data and the identifier in use, such as:</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.identificationAttempts.hybridEntryCiAttributes</td>
<td>Array of CI identifier entry attributes used during this additional CI identification attempt. Data type: Array. Attribute names and types depend on the request body data and the identifier in use, such as:</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.identificationAttempts.identifierName</td>
<td>Identifier rule used for this additional CI identification attempt. Data type: String.</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.identificationAttempts.searchOnTable</td>
<td>Name of the table searched for this additional CI identification attempt. Data type: String.</td>
</tr>
</tbody>
</table>

SKIPPED: Identification not attempted. The attributes required for this identifier rule table search were not provided, so the rule was not applied.
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| `<String>.additionalCommittedItems.identifierEntrySysId` | Sys_id for the identifier rule used to identify this additional CI. Notable values:  
- Unknown: Identification of this additional CI failed. See errors for details. |
| `<String>.additionalCommittedItems.inputIndices` | Array of index values for CIs from the request body items array that correspond to this additional CI. Data type: Array |
| `<String>.additionalCommittedItems.markers` | Array of marker values for internal use. Data type: Array |
| `<String>.additionalCommittedItems.mergedPayloads` | Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads table that were merged during processing of this additional CI. Data type: Array |
| `<String>.additionalCommittedItems.operation` | Operation performed for this additional CI. Data type: String  
Possible values:  
- ... |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• DELETE: An existing CI is removed from the target table.</td>
<td></td>
</tr>
<tr>
<td>• INSERT: The additional CI is inserted into the target table as a new record.</td>
<td></td>
</tr>
<tr>
<td>• NO_CHANGE: No operation is performed for the additional CI.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE: An existing CI in the target table is updated.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE_WITH_DOWNGRADE: An existing CI in the target table is updated and its class is changed to a more generic class.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE_WITH_SWITCH: An existing CI in the target table is updated and its class is changed to another class that is not an ancestor or descendent class.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE_WITH_UPGRADE: An existing CI in the target table is updated and its class is changed to a more specialized class.</td>
<td></td>
</tr>
</tbody>
</table>

<String>.additionalCommittedItems.sysId

Sys_id found for this additional CI through identification.

Data type: String

Notable values:
• Unknown: Identification of this additional CI failed. See errors for details.

<String>.additionalCommittedItems.warnings

Array of objects that describe a warning encountered while processing this additional CI.

Data type: Array

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.additionalCommittedItems.warnings.error</code></td>
<td>Type of warning encountered while processing this additional CI. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedItems.warnings.message</code></td>
<td>Warning message encountered while processing this additional CI. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations</code></td>
<td>Array of objects that describe a dependent relationship CI that was not included in the request body relations list to insert or update. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations.className</code></td>
<td>The sys_class_name of this additional dependent relationship CI. Only supported value: <code>cmdb_rel_ci</code>: The CI Relationship table. Data type: String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations.errorCount</code></td>
<td>Number of errors encountered while processing this additional dependent CI. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations.errors</code></td>
<td>Array of objects that describe errors encountered while processing this additional dependent CI. Data type: Array</td>
</tr>
</tbody>
</table>
|                                                    | `{  
|                                                    |   "error": "String", 
|                                                    |   "message": "String"  
|                                                    | }  
| `<String>.additionalCommittedRelations.errors.error` | Type of error encountered while processing this additional dependent CI. Data type: String |
| `<String>.additionalCommittedRelations.errors.message` | Error message encountered while processing this additional dependent CI. Data type: String |
| `<String>.additionalCommittedRelations.inputIndices` | Index values for dependent relationship CI objects in the request body that correspond to this additional dependent CI. Data type: Array |
| `<String>.additionalCommittedRelations.markers`    | Marker values for internal use. |
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations.mergedPayloadIds</code></td>
<td>Sys_id values for partial payloads from the [cmdb_ire_partial_payloads] table that were merged during processing of this additional dependent relationship CI.</td>
</tr>
</tbody>
</table>
| `<String>.additionalCommittedRelations.operation` | Operation performed for the additional dependent relationship CI. Possible values:  
  - INSERT: The dependent relationship CI is inserted into the target table as a new record.  
  - INSERT_AS_INCOMPLETE: The dependent relationship CI had errors and is inserted into the CMDB IRE Incomplete Payloads ([cmdb_ire_incomplete_payloads]) table.  
  - INSERT_AS_PARTIAL: The dependent relationship CI had errors and is inserted into the CMDB IRE Partial Payloads ([cmdb_ire_partial_payloads]) table. |
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• NO_CHANGE: No</td>
<td>operation is performed for the dependent relationship CI.</td>
</tr>
<tr>
<td>• UPDATE: An existing</td>
<td>dependent relationship CI in the target table is updated.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.hasError</td>
<td>Flag that indicates whether any item or relation has errors.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.hasWarning</td>
<td>Flag that indicates whether any item or relation has warnings.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items</td>
<td>Array of objects that describe the created or updated CIs.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
</tbody>
</table>

```json
"items": [
  {
    "add": 
    "class": 
    "dup": 
    "err": 
    "err": 
    "id": 
    "id": 
    "in": 
    "in": 
    "inp": 
    "mas": 
    "op": 
    "pa": 
    "re": 
    "rep": 
    "sys": 
```
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.additionalRelatedItems</td>
<td>List of JSON objects that provide information about additional lookup and related items that were processed but not provided as part of the input payload. These items are from partial payloads. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.items.additionalRelatedItems.className</td>
<td>Class/table name (sys_class_name) of the CI that was created or updated. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.additionalRelatedItems.inputIndices</td>
<td>Index values for CIs from the request body items array that correspond to this related item. Data type: Array of Numbers</td>
</tr>
<tr>
<td>&lt;String&gt;.items.additionalRelatedItems.mergedPayloadIds</td>
<td>List of sys_ids of the partial payloads that were merged into the related item. Located in the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.items.additionalRelatedItems.operation</td>
<td>Type of operation. Possible values:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• INSERT</td>
<td>New CI was inserted into the database.</td>
</tr>
<tr>
<td>• NO_CHANGE</td>
<td>No CI changes were made.</td>
</tr>
<tr>
<td>• UPDATE</td>
<td>Existing CI was updated.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.additionalRelatedItems.sysId</td>
<td>Sys_id of the CI that was updated or created.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.className</td>
<td>Class/table name (sys_class_name) of the CI that was created or updated.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.duplicateIndices</td>
<td>List of indexes of CIs that are duplicates of the current item.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.errorCount</td>
<td>Number of errors encountered while processing the item.</td>
</tr>
<tr>
<td>Data type: Number</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.errors</td>
<td>Array of objects in which each object describes an error encountered while processing this CI.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>*errors:</td>
<td></td>
</tr>
<tr>
<td>*: {</td>
<td></td>
</tr>
<tr>
<td>*error:</td>
<td></td>
</tr>
<tr>
<td>*message:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.errors.error</td>
<td>Type of error encountered while processing the CI.</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.errors.message</code></td>
<td>Error message associated with the error.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.identificationAttempts</code></td>
<td>List of attempts that were made to identify the CIs.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.identificationAttempts.attemptResult</code></td>
<td>Results of the attempt to identify the CI. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• MATCHED: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• MULTI_MATCH: Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• NO_MATCH: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• SKIPPED: Identification not attempted. The attributes were not specified in the query.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.attributes</td>
<td>List of CI identifier entry attributes that were used during the identification process. Data type: Array. Attribute names and types depend on the request body data and the identifier in use, such as:</td>
</tr>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.hybridEntryCiAttributes</td>
<td>List of CI identifier entry attributes that were used during the identification process. Data type: Array. Attribute names and types depend on the request body data and the identifier in use, such as:</td>
</tr>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.identifierName</td>
<td>Identifier rule used for this CI identification attempt. Data type: String.</td>
</tr>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.searchOnTable</td>
<td>Name of the table searched during the identification process. Data type: String.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.identifierEntrySysId</td>
<td>Sys_id for the identifier used to identify the CI. Located in the Identifier Entry [cmdb_identifier_entry] table. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.incompleteSysIds</td>
<td>If the item was saved as an incomplete payload, this parameter contains the sys_id of the record in the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.info</td>
<td>List of objects that contains additional information about the processing of the item. Data type: Array</td>
</tr>
</tbody>
</table>

```
"info": [
  {
    "code": "String",
    "message": "String",
    "ruleSysId": "String"
  }
]
```

<p>| &lt;String&gt;.items.info.code | Reclassification type that was skipped. Possible values: SKIPPED_CLASS_SWITCH, SKIPPED_CLASS_DOWNGRADE, SKIPPED_CLASS_UPGRADE. Data type: String |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.info.message</code></td>
<td>Message that provides additional insights into the reason for skipping the reclassification. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.info.ruleSysId</code></td>
<td>Sys_id of the reclassification restriction rule that was matched. Applicable only when IRE skips reclassification due to reclassification restriction rule. This value is empty if the reclassification is skipped due to a payload or a global flag. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.inputIndices</code></td>
<td>Indexes of the corresponding input CI. For top-level items, it is a list of integers. For related or lookup CIs, it is list of JSON objects. Data type: Array of Numbers</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.maskedAttributes</code></td>
<td>List of attributes whose update by a non-authoritative data source was skipped as defined by the Reconciliation Rules. Data type: Array</td>
</tr>
</tbody>
</table>
| `<String>.items.operation` | Operation that took place. Possible values:  
- INSERT: New CI was inserted into the database.  
- INSERT_AS_INCOMPLETE: Item was saved in cmdb_ire_incomplete_payloads table.  
- INSERT_AS_PARTIAL: Item was saved in cmdb_ire_partial_payloads table. |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.partialSysIds</td>
<td>If the item had errors and was saved as a partial payload, this parameter contains the sys_id of the partial payload record. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems</td>
<td>List of JSON objects that describe a related CI from the request body. Data type: Array</td>
</tr>
</tbody>
</table>

- **NO_CHANGE**: No CI changes were made.
- **UPDATE**: Existing CI was updated.
- **UPDATE_WITH_DOWNGRADE**: CI was updated and the class changed to a more generic class (ancestor class).
- **UPDATE_WITH_SWITCH**: CI was updated and the class changed to a class that is not ancestor or descendent.
- **UPDATE_WITH_UPGRADE**: CI was updated and the class changed to a more specialized class (descendent class).
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.relatedItems.className</td>
<td>Class/table name (sys_class_name) of the related item. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.errors</td>
<td>List of errors that occurred during processing of the related item. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.errors.error</td>
<td>Type of error encountered while processing the related item. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.errors.message</td>
<td>Error message associated with the error. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.errorCount</td>
<td>Number of errors detected while processing the related items. Data type: Number</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.incompleteSysIds</td>
<td>If the related item was saved as an incomplete payload, this value is the sys_id of the record in the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.inputIndices</code></td>
<td>Index of the corresponding input item. For top-level items, it is a list of integers. For related or lookup items, it is a list of JSON objects. Data type: Array of Numbers or Array of Objects</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.inputIndices.mainIndex</code></td>
<td>Index value from the request body <code>items</code> array that corresponds to the CI parent of the related item. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.inputIndices.subIndex</code></td>
<td>Index value from the request body <code>items.lookup</code> array that corresponds to the related item. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.mergedPayloadIds</code></td>
<td>List of sys_ids of the partial payloads that were merged into the CI during processing. Data type: Array</td>
</tr>
</tbody>
</table>
| `<String>.items.relatedItems.operation` | Operation that took place. Possible values:  
  - INSERT: New related CI was inserted into the database.  
  - INSERT_AS_INCOMPLETE: Item was saved in `cmdb_ire_incomplete_payloads` table.  
  - UPDATE: Related CI was updated or replaced.  
  - DELETE: Related CI was deleted.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• INSERT:</td>
<td>Item was saved in cmdb_ire_partial_payloads table.</td>
</tr>
<tr>
<td>• NO_CHANGE:</td>
<td>No related CI changes were made.</td>
</tr>
<tr>
<td>• UPDATE:</td>
<td>Existing related CI was updated.</td>
</tr>
<tr>
<td>• UPDATE_WITH_DOWNGRADE:</td>
<td>Related CI was updated and the class changed to a more generic class (ancestor class).</td>
</tr>
<tr>
<td>• UPDATE_WITH_SWITCH:</td>
<td>Related CI was updated and the class changed to a class that is not ancestor or descendent.</td>
</tr>
<tr>
<td>• UPDATE_WITH_UPGRADED:</td>
<td>Related CI was updated and the class changed to a more specialized class (descendent class).</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.partialSysIds</code></td>
<td>If the related item had errors and was saved as a partial payload, this contains a list of the sys_ids of the associated records in the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.sysId</code></td>
<td>The sys_id of the related item.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.warningCount</code></td>
<td>Number of warnings encountered when processing the related items.</td>
</tr>
<tr>
<td>Data type: Number</td>
<td></td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.warnings</code></td>
<td>Array of objects that describes a warning encountered while processing the related items. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.warnings.error</code></td>
<td>Type of warning encountered while processing the related item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.warnings.message</code></td>
<td>Message associated with the warning. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedSysIds</code></td>
<td>List of the CIs used during lookup-based identification. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.sys_id</code></td>
<td>Sys_id of the CI that was updated or created. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations</code></td>
<td>List of JSON objects that describe a dependent relationship CI from the request body relations array. Data type: Array</td>
</tr>
</tbody>
</table>

---

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| `<String>.relations.className` | Sys_class_name of this dependent relationship CI. Only supported value:  
  - cmdb_rel_ci: CI Relationship table. Data type: String |
| `<String>.relations.errorCount` | Number of errors encountered when processing the dependent relationship CI. Data type: Number |
| `<String>.relations.errors` | Array of objects that describe errors that were encountered while processing the dependent relationship CI. Data type: Array |
| `<String>.relations.errors.error` | Type of error encountered while processing the dependent relationship CI. Data type: String |
| `<String>.relations.errors.message` | Error message encountered while processing the dependent relationship CI. Data type: String |
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.relations.incompleteSysIds</code></td>
<td>If the relation was saved as an incomplete payload of the request, this value is the sys_id of the record in the CMDB IRE Incomplete Payloads table. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations.inputIndices</code></td>
<td>Indexes for the dependent relationship CI objects in the request body relations array that correspond to this dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations.operation</code></td>
<td>Type of operation performed. Possible values: • INSERT: The dependent relationship CI was inserted into the target table as a new record. • INSERT_AS_INCOMPLETE: The dependent relationship CI had errors and was inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table. • INSERT_AS_PARTIAL: The dependent relationship CI had errors and was inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• NO_CHANGE: No operation was performed.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE: An existing dependent relationship CI in the target table was updated.</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

<String>.relations.partialSysIds

If the relation had errors and was saved as a partial payload, this value is the sys_id of the record in the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.

Data type: String

<String>.relations.sysId

Sys_id of the dependent relationship CI.

Data type: String

<String>.summary

List of JSON properties that provide statistics on how many items were inserted, updated, and such, per class.

Data type: Array

<String>.summary.<class_name>

Statistics for a specific class.

Data type: Object

```json
<class_name>: {
  "additionalInsertedItemCount": Number,
  "errorCount": Number,
  "incompleteItemCount": Number,
  "insertedItemCount": Number,
  "partialItemCount": Number,
  "skippedItemCount": Number,
  "unchangedItemCount": Number,
  "updatedItemCount": Number,
}
```
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.additionalInsertedItemCount</code></td>
<td>Number of items inserted due to processing of partial payloads. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.errorCount</code></td>
<td>Number of errors encountered when processing items. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.incompleteItemCount</code></td>
<td>Number of items inserted in the CMDB IRE Incomplete Payloads table. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.insertedItemCount</code></td>
<td>Number of items created. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.partialItemCount</code></td>
<td>Number of items saved in the Partial Payload table [cmdb_ire_partial_payloads]. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.skippedItemCount</code></td>
<td>Number of items that were skipped. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.unchangedItemCount</code></td>
<td>Number of items that had entries but were not modified. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.updatedItemCount</code></td>
<td>Number of items updated. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.warningCount</code></td>
<td>Number of items that generated a warning when processed. Data type: Number</td>
</tr>
</tbody>
</table>
This code example shows how to obtain the payload of an ImportSet, where you can then update the payload, and then set the new payload before calling the cmdbUtil.identifyAndReconcileEnhanced() method to process the payload.

```javascript
// Add this code to the onBefore transform map script
// Ability to modify the payload using get and set APIs
// Call cmdbUtil API to do Identification and Reconciliation of current row
var cmdbUtil = new CMDBTransformUtil();
cmdbUtil.setDataSource('ImportSet');
var payload = cmdbUtil.getPayload(source, map, log);
// Modify the payload here
cmdbUtil.setPayload(payload);
cmdbUtil.identifyAndReconcileEnhanced(source, map, log);
ignore = true;

if (cmdbUtil.hasError()) {
    var errorMessage = cmdbUtil.getError();
    log.error(errorMessage);
} else {
    log.info('IE Output Payload: ' + cmdbUtil.getOutputPayload());
    log.info('Imported CI: ' + cmdbUtil.getOutputRecordSysId());
}
```

**CMDBTransformUtil - hasError()**

Determines if an error occurred in the previous identifyAndReconcile() or identifyAndReconcileEnhanced() method call.

Use this method in onBefore transform map scripts.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Boolean</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• true</td>
<td>An error occurred</td>
</tr>
<tr>
<td>• false</td>
<td>No error occurred</td>
</tr>
</tbody>
</table>

// Add this code to the onBefore transform map script
// Call CMDB API to do Identification and Reconciliation of current row
var cmdbUtil = new CMDBTransformUtil();
cmdbUtil.setDataSource('ImportSet');
cmdbUtil.identifyAndReconcileEnhanced(source, map, log);
ignore = true;

if (cmdbUtil.hasError()) {
    var errorMessage = cmdbUtil.getError();
    log.error(errorMessage);
} else {
    log.info('IE Output Payload: ' + cmdbUtil.getOutputPayload());
    log.info('Imported CI: ' + cmdbUtil.getOutputRecordSysId());
}

Scoped equivalent
There is no scoped equivalent for this method.

CMDBTransformUtil - identifyAndReconcile(Object source, Object map, Object log)

Inserts or updates a configuration item (CI) in the Configuration Management Database (CMDB).

Use this method in onBefore transform map scripts. Using this method instead of relying on the transform to insert or update the configuration item record reduces duplicate entries in the CMDB. For additional information, see Apply CI Identification and Reconciliation to Import Sets.

To prevent the transform from adding or updating the record a second time, set ignore = true.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>Object</td>
<td>Global object available in transform map scripts that holds the incoming record.</td>
</tr>
<tr>
<td>map</td>
<td>Object</td>
<td>Map object available in transform map scripts that holds the current transform map record.</td>
</tr>
<tr>
<td>log</td>
<td>Object</td>
<td>Log object available in transform map scripts.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This code example shows how to call the `identifyAndReconcile()` method to process the current row of the input payload.

```java
// Add this code to the onBefore transform map script
// Call CMDBTransformUtil API to do Identification and Reconciliation of the current row
var cmdbUtil = new CMDBTransformUtil();
cmdbUtil.identifyAndReconcile(source, map, log);
ignore = true;
```

### CMDBTransformUtil - identifyAndReconcileEnhanced(Object source, Object map, Object log)

Inserts or updates a configuration item (CI) in the Configuration Management Database (CMDB).

In addition to providing the functionality of the `CMDBTransformUtil.identifyAndReconcile()` method, this method also supports:

- Handling partial payloads
- Handling partial commits
- Removing duplicate items within a payload
- Generating output summaries

For additional information on Identification and Reconciliation and more detailed explanations of the data used by this method, see [Identification and Reconciliation (IRE)](https:// servicenow.com).
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>Object</td>
<td>Global object available in transform map scripts that holds the incoming record.</td>
</tr>
<tr>
<td>map</td>
<td>Object</td>
<td>Map object available in transform map scripts that holds the current transform map record.</td>
</tr>
<tr>
<td>log</td>
<td>Object</td>
<td>Log object available in transform map scripts.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This code example shows how to call the `identifyAndReconcileEnhanced()` method to process the passed in payload.

```java
// Add this code to the onBefore transform map script
// Call CMDBTransformUtil API to do Identification and Reconciliation
var cmdbUtil = new CMDBTransformUtil();
cmdbUtil.identifyAndReconcileEnhanced(source, map, log);
ignore = true;
```

**CMDBTransformUtil - logTransformStats(Object log)**

Logs the number of configuration items (CI) inserted, updated, skipped, or that had errors.

Use this method in onComplete transform map scripts.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>log</td>
<td>Object</td>
<td>Log object available in transform map scripts.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var cmdbUtil = new CMDBTransformUtil();
cmdbUtil.logTransformStats(log);

Scoped equivalent
There is no scoped equivalent for this method.

CMDBTransformUtil - setDataSource(String source)
Sets the data source to use when the identifyAndReconcile() or identifyAndReconcileEnhanced() method is called.

Use this method in onBefore transform map scripts. If this method is not called, the default value ImportSet.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>String</td>
<td>Data source for the source configuration item record.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

// Add this code to the onBefore transform map script
// Call CMDB API to do Identification and Reconciliation of current row
var cmdbUtil = new CMDBTransformUtil();
cmdbUtil.setDataSource('ImportSet');
cmdbUtil.identifyAndReconcileEnhanced(source, map, log);
ignore = true;

if (cmdbUtil.hasError()) {
    var errorMessage = cmdbUtil.getError();
    log.error(errorMessage);
} else {
    log.info('IE Output Payload: ' + cmdbUtil.getOutputPayload());
    log.info('Imported CI: ' + cmdbUtil.getOutputRecordSysId());
}

Scoped equivalent
There is no scoped equivalent for this method.
CMDBUtil - Global

The CMDBUtil API provides utility methods for creating and managing table relationships in the configuration management database (CMDB) and managing CMDB baselines.

CMDBUtil is a JavaScript-accessible ScriptableObject. The CMDBUtil API has dynamic and static methods. You access dynamic methods by creating a SNC.CMDBUtil object. You access static methods by using SNC.CMDBUtil global object to call the methods.

This example creates a CMDBUtil object.

```javascript
var cu = new SNC.CMDBUtil();
cu.baselineProposedChangesGenDIFF(current, action.get('sysparm_changeset'));
```

This example calls a static method.

```javascript
var output = SNC.CMDBUtil.getAllChildrenOfAsCommaList('cmdb_ci_computer');
```

Use these methods to manage CMDB table relationships and baselines.

CMDBUtil - baselineProposedChangesApplyChanges(GlideRecord proposedChanges)

Applies proposed changes in a task_ci record that represent an update set for all CIs associated with the task.

This is a dynamic method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>proposedChanges</td>
<td>GlideRecord</td>
<td>The collection of proposed changes in the change request (CHG).</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
//Where current is a GlideRecord and action is the current UI action
var base = new SNC.CMDBUtil();
base.baselineProposedChangesApplyChanges(current);
```
CMDBUtil - baselineProposedChangesGenDiff(GlideRecord current, String changeSet)
Generates the XML for proposed changes diff, and adds it to the corresponding task_ci record.
This is a dynamic method.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>current</td>
<td>GlideRecord</td>
<td>The GlideRecord that contains the collection of proposed changes.</td>
</tr>
<tr>
<td>changeSet</td>
<td>String</td>
<td>The sysid of the task_ci record that represents the change set to use.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

//Where current is a GlideRecord and action is the current UI action
var base = new SNC.CMDBUtil();
base.baselineProposedChangesGenDiff(current, action.get('sysparm_changeset'));

CMDBUtil - bootstrap(String dictFile)
Bootstrap the environment by creating the database, creating the system dictionary table, and having the system dictionary table describe itself.
This is a static method.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dictFile</td>
<td>String</td>
<td>The file name including path, of the dictionary to load.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
CMDBUtil - createCIRelationship(String tableName, String parentField, String childField, String parentDesc, String childDesc)

Creates the specified CI relationship using the specified invocation parameters.

If called without the first parameter, passing only four parameters, defaults to the CI Relationship [cmdb_rel_ci] table.

This is a dynamic method.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>The table name.</td>
</tr>
<tr>
<td>parentField</td>
<td>String</td>
<td>The parent field.</td>
</tr>
<tr>
<td>childField</td>
<td>String</td>
<td>The child field</td>
</tr>
<tr>
<td>parentDesc</td>
<td>String</td>
<td>The parent relationship descriptor.</td>
</tr>
<tr>
<td>childDesc</td>
<td>String</td>
<td>The child relationship descriptor.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CMDBUtil - getAllChildrenOfAsCommaList(String baseTable)

Gets all the child tables of the specified table as a comma-separated list.

This is a static method.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseTable</td>
<td>String</td>
<td>The base table name.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A comma-separated list of sys_ids of tables extending the base table.</td>
</tr>
</tbody>
</table>
//Where cmdv_ci_computer is a table.
var output = SNC.CMDBUtil.GetAllChildrenOfAsCommaList('cmdb_ci_computer');
gs.print(output);

CMDBUtil - getCMDBViews()

Gets all the records in the CMDB View [cmdb_view] table.

This is a dynamic method.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArrayList</td>
<td>The records in the CMDB view table.</td>
</tr>
</tbody>
</table>

CMDBUtil - getTables0(String tableName)

Gets a list of all the parents of a table.

This is a static method.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>tableName</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArrayList</td>
<td>List of tables that are parents of the specified table.</td>
</tr>
</tbody>
</table>
**CMDBUtil - isExcludedFromBR(String className)**

Determines whether a CI class is defined in the Business Rule Exclusion Lists [cmdb_business_rule_exclusions] table. Use this method in an advanced condition to prevent a business rule from executing on excluded CI classes.

For example, the Create Asset on insert business rule uses this method in the advanced condition to prevent the rule from creating assets for CI classes such as cmdb_ci_qualifier, cmdb_ci_endpoint, cmdb_ci_storage_volume, and cmdb_ci_vcenter_datastore_disk. Define classes for exclusion in the Business Rule Exclusion Lists [cmdb_business_rule_exclusions] table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>className</td>
<td>String</td>
<td>Name of the class to check for exclusion.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the class is defined in the Business Rule Exclusion Lists [cmdb_business_rule_exclusions] table; otherwise, false.</td>
</tr>
</tbody>
</table>

//Returns true if the cmdb_ci_endpoint class is defined in the Business Rule Exclusion Lists table
var exclusion = SNC.CMDBUtil.isExcludedFromBR("cmdb_ci_endpoint");

//Advanced condition that prevents business rules from executing on classes in the Business Rule Exclusion Lists table
(!SNC.CMDBUtil.isExcludedFromBR(current.getTableName()))

**CMDBUtil - processCIChange(GlideRecord event, GlideRecord target)**

Wraps the call to RelationshipEventProcessor(), which processes any changes to CI relationships.

This is a dynamic method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
<td>GlideRecord</td>
<td>The event record.</td>
</tr>
</tbody>
</table>
CMDBUtil - processRelChange(GlideRecord event, GlideRecord current, String relType, String triplet)

Wraps the call to `RelationshipEventProcessor()`, which processes any changes to CI relationships, with the specified type and triplet.

This is a dynamic method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
<td>GlideRecord</td>
<td>The event record</td>
</tr>
<tr>
<td>current</td>
<td>GlideRecord</td>
<td>The current record, which is either the relation record or a user record if the current process is a deletion.</td>
</tr>
<tr>
<td>relType</td>
<td>String</td>
<td>The type of relation that changed.</td>
</tr>
<tr>
<td>triplet</td>
<td>String</td>
<td>The child, parent, and class name from the relation that changed.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CMDBUtil - removeCIRelationship(String tableName, String parentField, String childField, String parentDesc, Object childDesc)

Deletes the specified CI relationship.
If called without the first parameter, passing only four parameters, defaults to the CI Relationship [cmdb_rel_ci] table.

This is a dynamic method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>The table name</td>
</tr>
<tr>
<td>parentField</td>
<td>String</td>
<td>The parent field</td>
</tr>
<tr>
<td>childField</td>
<td>String</td>
<td>The child field</td>
</tr>
<tr>
<td>parentDesc</td>
<td>String</td>
<td>The parent relationship descriptor</td>
</tr>
<tr>
<td>childDesc</td>
<td>Object</td>
<td>The child relationship descriptor</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**CMDBUtil - reParentTable(String table, String oldParent, String newParent)**

Changes the parent of the given table to the new parent.

This is a static method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>The table to re-parent</td>
</tr>
<tr>
<td>oldParent</td>
<td>String</td>
<td>The old parent</td>
</tr>
<tr>
<td>newParent</td>
<td>String</td>
<td>The new parent</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True, if successful; otherwise, false.</td>
</tr>
</tbody>
</table>
Color - Scoped, Global

Creates a Color object used to define color attributes that you can apply to elements in a PDF; such as cells, tables, and lines.

This API is part of the ServiceNow PDF Generation Utilities plugin (com.snc.apppdfgenerator) and is provided within the sn_pdfgeneratorutils namespace. The plugin is activated by default.

This API is a component used with the Document API to generate a PDF.

Color - Color(Array colors)

Instantiates a new Color object with RGB values.

Color can be applied to the following scenarios:

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>colors</td>
</tr>
</tbody>
</table>

The following example shows how to create a Color object.

```javascript
var color = new sn_pdfgeneratorutils.Color([0.1, 0.9, 0.5]); //given as object containing RGB values
```

Color – equals(Color color)

Indicates whether the values of two different color objects match.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>color</td>
</tr>
<tr>
<td>Returns</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td><strong>Type</strong></td>
</tr>
</tbody>
</table>
| Boolean | Flag that indicates whether the values of two color objects match.  
Valid values:  
• true: The colors match.  
• false: The colors do not match. |

The following example shows how to create two color objects and determine if the colors match.

```javascript
var color1 = new sn_pdfgeneratorutils.Color([1,0.5,0.5]);  // given as an array of RGB values
var color2 = new sn_pdfgeneratorutils.Color([0.8,0.5,0.5]);         // given as an array of RGB values
var isequal = color1.equals(color2);
```

**Color – getGrayColor(Number grayScale)**

Returns a black, gray, or white color object.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>grayScale</td>
</tr>
</tbody>
</table>

The following example shows how to create a color object that is 50% grayscale.

```javascript
var grayColor = new sn_pdfgeneratorutils.Color.getGrayColor(0.5);
```

**Color – setColorValue(Array colors)**

Creates color with given values and enables you to change the values of an existing color. Each of the values must be from 0 through 1.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>colors</td>
<td>Array</td>
<td>Three numbers indicating RGB values using a decimal value from 0 through 1. For example, in [0.1, 0.9, 0.5], the value of the first position is red, second is green, and third is blue. Also, [0, 0, 0] is solid black, [0.5, 0.5, 0.5] is solid gray, and [1, 1, 1] is solid white.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to change the values of an existing color.

```javascript
var color = new sn_pdfgeneratorutils.Color([1,0.5,0.5]); //given as array of RGB values;
color.setColorValue(color);
```

**Color – setOpacity(Number opacity)**

Sets the level of color opacity.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>Color</td>
<td>Floating decimal value from 0 through 1, in which 0 is fully transparent and 1 is fully opaque.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to create a color object and set its opacity to 50 percent.

```javascript
var color = new Color([1,0.5,0.5]);
color.setOpacity(0.5);
```
ConnectActionResponse - Global

The ConnectActionResponse API provides a method to create a GlideRecord. ConnectActionResponse methods are accessed through a global object (response) that is available only in Connect action scripts.

ConnectActionResponse - newRecord(String table, Object values, String view)

Create a GlideRecord in the specified table with the specified values.

Opens a form in the UI based upon the table, which replaces the Table and Document fields in the Live Group Profile record for the associated conversation.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>Name of the table where the record is to be added.</td>
</tr>
<tr>
<td>values</td>
<td>Object</td>
<td>An object of name value pairs where the names are field names in the table and the values are the field values.</td>
</tr>
<tr>
<td>view</td>
<td>String</td>
<td>The form view to load. This parameter is optional.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
response.newRecord("incident", {
  short_description: conversation.document.short_description || ",
  caller_id: conversation.document.opened_by
});
```

ConnectionInfo - Scoped, Global

Use the ConnectionInfo API to get connection attribute information through the connection and credential alias.

You can use this API in scoped applications, or within the global scope. In scoped scripts, use the sn_cc namespace identifier.

For more information on connections and credentials, see Credentials and connection information.
This function retrieves connection attribute information identified by the given connection and credential alias.

```java
var provider = new sn_cc.ConnectionInfoProvider();

// get a jdbc connection in the current domain with the alias ID
//     "6219afbf9f03320021dd7501942e70fc"
var connectionInfo = provider.getConnectionInfo("6219afbf9f03320021dd7501942e70fc");
if (connectionInfo != null) {
    // get data map
    var datamap = connectionInfo.getDataMap();
    gs.info(datamap["name"]);
    gs.info(datamap["connection_url"]);

    // get the same values using getAttribute
    gs.info(connectionInfo.getAttribute("name"));
    gs.info(connectionInfo.getAttribute("connection_url"));

    // get credential attributes
    gs.info(connectionInfo.getCredentialAttribute("user_name"));
    gs.info(connectionInfo.getCredentialAttribute("password"));

    // get extended attributes
    var extendedAttributes = connectionInfo.getExtendedAttributes();
    gs.info(extendedAttributes["name1"]);
}

// get a jdbc connection in the ACME domain with the alias ID
//     "cd5923ff9f03320021dd7501942e70bb"
connectionInfo = provider.getConnectionInfoByDomain("cd5923ff9f03320021dd7501942e70bb",
                                                  "c90d4b084a362312013398f0512720c0d");
if (connectionInfo != null) {
    // get data map
    var datamap = connectionInfo.getDataMap();
    gs.info(datamap["name"]);
}
```

**Scoped ConnectionInfo - getAttribute(String name)**

Returns the value of a `ConnectionInfo` attribute with the specified property name.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of a ConnectionInfo object property.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of a specified ConnectionInfo property.</td>
</tr>
</tbody>
</table>

```java
// get the same values using getAttribute
gs.info(connectionInfo.getAttribute("name"));
gs.info(connectionInfo.getAttribute("connection_url"));
```

---

**Scoped ConnectionInfo - getCredentialAttribute()**

Returns the value of credential attributes for a specified connection.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Key-value pair map of credential attributes.</td>
</tr>
</tbody>
</table>

```java
// get credential attributes
gs.info(connectionInfo.getCredentialAttribute("user_name"));
gs.info(connectionInfo.getCredentialAttribute("password"));
```

---

**Scoped ConnectionInfo - getDataMap()**

Returns the connection attributes as a collection of key-value pairs.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Key-value pair map of connection attributes.</td>
</tr>
</tbody>
</table>

```java
// get the same values using getDataMap
gs.info(connectionInfo.getDataMap());
```
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Key-value pair map of connection attributes.</td>
</tr>
</tbody>
</table>

```javascript
// get data map
var datamap = connectionInfo.getDataMap();
gs.info(datamap["name"]);
gs.info(datamap["connection_url"]);
```

Scoped ConnectionInfo - getExtendedAttributes()

Returns the extended attributes as a collection of key-value pairs.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Key-value pair map of extended attributes.</td>
</tr>
</tbody>
</table>

```javascript
// get extended attributes
var extendedAttributes = connectionInfo.getExtendedAttributes();
gs.info(extendedAttributes["name1"]);
```

ConnectionInfoProvider - Scoped, Global

Use the ConnectionInfoProvider API to select connection information through the connection alias.

You can use this API in scoped applications, or within the global scope. In scoped scripts, use the `sn_cc` namespace identifier.
This function retrieves connection information identified by the given connection alias.

```
var provider = new sn_cc.ConnectionInfoProvider();

// get a jdbc connection in the current domain with the alias ID
// "6219afbf9f03320021dd7501942e70fc"
var connectionInfo = provider.getConnectionInfo("6219afbf9f03320021dd7501942e70fc");
if (connectionInfo != null) {
    // get data map
    var datamap = connectionInfo.getDataMap();
    gs.info(datamap["name"]);
    gs.info(datamap["connection_url"]);

    // get the same values using getAttribute
    gs.info(connectionInfo.getAttribute("name"));
    gs.info(connectionInfo.getAttribute("connection_url"));

    // get credential attributes
    gs.info(connectionInfo.getCredentialAttribute("user_name");
    gs.info(connectionInfo.getCredentialAttribute("password");

    // get extended attributes
    var extendedAttributes = connectionInfo.getExtendedAttributes();
    gs.info(extendedAttributes["name1"]);
}

// get a jdbc connection in the ACME domain with the alias ID
// "cd5923ff9f03320021dd7501942e70bb"
connectionInfo = provider.getConnectionInfoByDomain("cd5923ff9f03320021dd7501942e70bb",
                                                "c90d4b084a362312013398f051272c0d");
if (connectionInfo != null) {
    // get data map
    var datamap = connectionInfo.getDataMap();
    gs.info(datamap["name"]);
}
```

**ConnectionInfoProvider - ConnectionInfoProvider()**

Selects connection information through the connection alias.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**ConnectionInfoProvider - getConnectionInfo(String aliasID)**

Retrieves a ConnectionInfo object identified by the given aliasID in the current domain.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aliasID</td>
<td>String</td>
<td>Sys_id of a connection alias.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConnectionInfo</td>
<td>Information about the connection.</td>
</tr>
</tbody>
</table>

```javascript
var provider = new sn_cc.ConnectionInfoProvider();

// get a jdbc connection in the current domain with the alias ID
//   "6219afbf9f03320021dd7501942e70fc"
var connectionInfo = provider.getConnectionInfo("6219afbf9f03320021dd7501942e70fc");
```

**ConnectionInfoProvider - getConnectionInfoByDomain(String aliasID, String domainID)**

Retrieves a ConnectionInfo object identified by the given aliasID for a specific domain.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aliasID</td>
<td>String</td>
<td>Sys_id of a connection alias.</td>
</tr>
<tr>
<td>domainID</td>
<td>String</td>
<td>Sys_id of a domain or global.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConnectionInfo</td>
<td>Connection information.</td>
</tr>
</tbody>
</table>

```javascript
var provider = new sn_cc.ConnectionInfoProvider();

// get a jdbc connection in the ACME domain with the alias ID
//    "cd5923ff9f03320021dd7501942e70bb"
connectionInfo = provider.getConnectionInfoByDomain("cd5923ff9f03320021dd7501942e70bb", "c90d4b084a362312013398f051272c0d");
```

Conversation - Scoped

The Conversation API enables you to create or modify Connect conversations. To use this class in a scoped application, use the `sn_connect` namespace identifier. The Connect Scriptable APIs plugin (ID: com.glide.connect.scriptable) should be enabled to access the Conversation API.

Scoped Conversation - get(String sysID)

Get an existing Connect conversation by sys_id.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysID</td>
<td>String</td>
<td>The sys_id of the conversation record.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Conversation object</td>
</tr>
</tbody>
</table>

```javascript
var conversation = sn_connect.Conversation.get("27b9844c1385030034bb58a12244b037");
```

Scoped Conversation - create(String name)

Create a Connect conversation.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Create a conversation with a specific name.</td>
</tr>
</tbody>
</table>
| type | String | Include a specific conversation type. The type is determined by the type choice list. The base system includes the following type options:  
• connect  
• support  
• group  
• peer  
• qanda  
• team |

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Scriptable Conversation</td>
</tr>
</tbody>
</table>

```javascript
var conversation = sn_connect.Conversation.create({  
    name: "Hello world",  
    type: "connect"  
});
```

## Scoped Conversation - addSubscriber(String sysID)

Add a user to a conversation.

## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysID</td>
<td>String</td>
<td>The sys_ID of the user you want to add to a conversation.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
**Scoped Conversation - removeSubscriber(String sysID)**
Remove a user from a conversation.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>SysID</td>
<td>String</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Scoped Conversation - sendMessage(String body, String field)**
Send a message to a conversation.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>Body</td>
<td>String</td>
</tr>
<tr>
<td>Field</td>
<td>String</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
CostPlanBatchOperations - Global

Provides methods that enable the creation of a large number of cost plan records for demands and projects using batch processing. Use this API instead of GlideRecord to more efficiently create multiple cost plan records.

Use the CostPlanBatchOperations.add() method to add any number of cost plans to a batch queue. Once you have added all the required cost plans, use the CostPlanBatchOperations.process() method to actually create/insert the cost plans in your instance. Until you call the CostPlanBatchOperations.process() method, the add requests remain in the batch queue (they are not yet added to your instance). If for some reason you need to remove all of the cost plans in the batch queue, use the CostPlanBatchOperations.clear() method.

To use this API you must activate the com.snc.financial_planning_pmo plugin.

CostPlanBatchOperations - add(Object costPlan)

Adds one or more cost plans to a specified task (project or demand). Use this method when you want to create multiple cost plans.

Once processed, the cost plans are inserted into the specified project or demand. You can then view and modify them within your instance. For additional information, see Create a project cost plan and Create a demand cost plan.

Note: This method only defines the cost plans to add to your instance. The cost plans are not inserted into your instance until the CostPlanBatchOperations.process() is called.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>costPlan</td>
<td>Object</td>
<td>One or more objects or an array of objects that describe each of the cost plans to add to an existing task.</td>
</tr>
<tr>
<td>costPlan.name</td>
<td>String</td>
<td>Name of the cost plan. Maximum length: 130 characters</td>
</tr>
<tr>
<td>costPlan.task</td>
<td>String</td>
<td>Sys_id of the project or demand to associate with this cost plan. You can</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>locate this value in one of the following tables:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Project [pm_project]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Project Task [pm_project_task]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Demand [dmn_demand]</td>
</tr>
<tr>
<td>costPlan.unit_cost</td>
<td>Number</td>
<td>Cost of a single unit of the specified resource.</td>
</tr>
<tr>
<td>costPlan.resource_type</td>
<td>String</td>
<td>Sys_id of the record that defines the cost type associated with this cost plan. The available values for this parameter are defined in the Cost Type Definition [resource_type_definition] table.</td>
</tr>
<tr>
<td>costPlan.start_fiscal_period</td>
<td>String</td>
<td>Sys_id of the record that defines the starting fiscal period to associate with this cost plan. The available values for this parameter are defined in the Fiscal period [fiscal_period] table.</td>
</tr>
<tr>
<td>costPlan.end_fiscal_period</td>
<td>String</td>
<td>Sys_id of the record that defines the ending fiscal period to associate with this cost plan. The available values for this parameter are defined in the Fiscal period [fiscal_period] table.</td>
</tr>
<tr>
<td>costPlan.&lt;optional&gt;</td>
<td>Varied</td>
<td>Optional. You can pass in additional parameters to add to a cost plan. The available parameters depend on the type of cost plan that you are creating. Refer to the Cost Plan [cost_plan] table for the list of additional parameters that you can pass.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
This example shows how to add a simple batched cost plan.

```javascript
// Define Array of Cost Plan records in JSON format
var costPlanRecords = [];
costPlanRecords.push(
    {
        name: 'Capital Expense',
        task: 'f7a36d1bd5b8001025c85a35dc96193a', // sys_id of the task
        unit_cost: 1000.00, // decimal
        resource_type: 'a546eaf79330120064f572edb67ff60', // sys_id of the cost type definition
        start_fiscal_period: '091b6e60cb111200f2de77a4634c9c2e', // sys_id of the start fiscal period record
        end_fiscal_period: '0d1b6e60cb111200f2de77a4634c9c2f', // sys_id of the end fiscal period record
        quantity: 1 // Optional cost plan record attributes
    });

var costPlanBatchOperations = new CostPlanBatchOperations();
costPlanBatchOperations.add(costPlanRecords);
var costPlanSysIds = costPlanBatchOperations.process();
```

**CostPlanBatchOperations - clear()**

Removes all cost plan objects that were added using the CostPlanBatchOperations.add() method.

ℹ️ **Note:** This method only works for cost plans that have been added. Once a cost plan is processed, this method cannot remove it. Processed cost plans must be removed manually through the UI.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to clear the batch queue after adding cost plans.
//Define Array of Cost Plan records in JSON format
var costPlanRecords= [];
costPlanRecords.push({
    name:'Capital Expense',
    task:'f7a36d1b958001025c85a35dc96193a', // sys_id of the task
    unit_cost:1000.00, //decimal
    resource_type:'a546eaf79330120064f572edb67ff70', // sys_id of the cost type definition
    start_fiscal_period:'091b6e60cb11200f2de77a46d34c9e2', // sys_id of the start fiscal period record
    end_fiscal_period:'0d1b6e60cb11200f2de77a4634c9c2f', // sys_id of the end fiscal period record
    quantity:1 // Optional cost plan record attributes
});

var costPlanBatchOperations = new CostPlanBatchOperations();
costPlanBatchOperations.add(costPlanRecords);
var costPlanSysIds = costPlanBatchOperations.process();
costPlanBatchOperations.clear();

**CostPlanBatchOperations - process()**

Processes all of the cost plans that were added using the CostPlanBatchOperations.add() method and creates corresponding cost plans and relevant rollups.

Once the cost plans are successfully processed, the cost plan queue is cleared.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Array</td>
</tr>
</tbody>
</table>

This example shows how to process a simple batched cost plan.

//Define Array of Cost Plan records in JSON format
var costPlanRecords= [];
costPlanRecords.push({
name: 'Capital Expense',
task: 'f7a36d1dbd58001025c85a35dc96193a', // sys_id of the task
unit_cost: 1000.00, // decimal
resource_type: 'a546eaf79330120064f572ed572e67ffeb70', // sys_id of the cost type definition
start_fiscal_period: '091b6e60cb111200f2de77a4634c9c2e', // sys_id of the start fiscal period record
end_fiscal_period: '0d1b6e60cb111200f2de77a4634c9c2f', // sys_id of the end fiscal period record
quantity: 1 // Optional cost plan record attributes
});

costPlanBatchOperations = new CostPlanBatchOperations();
costPlanBatchOperations.add(costPlanRecords);
costPlanSysIds = costPlanBatchOperations.process();

CriteriaEvaluator - Global

Implements a general-purpose criteria evaluator.

The general use pattern is to construct the class, add as many criteria as needed, then evaluate.

CriteriaEvaluator - addCriterion(String lhComparand, String operator, String rhComparand)

Adds the criteria to be evaluated.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lhComparand</td>
<td>String</td>
<td>The left hand comparand. Must be a named value in the left hand values input into evaluate().</td>
</tr>
<tr>
<td>operator</td>
<td>String</td>
<td>The operator. Must be one of: equals, starts with, contains, does not contain, ends with, or regex matches.</td>
</tr>
<tr>
<td>rhComparand</td>
<td>String</td>
<td>The right hand comparand. Can be evaluated either as one of the right hand values input to evaluate() or as a literal string.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

CriteriaEvaluator - evaluate(Object lhValues, String rhValues, Boolean any)

Evaluates the criteria.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lhValues</td>
<td>Object</td>
<td>A hashmap of name-value pairs to evaluate the left hand comparands of the criteria.</td>
</tr>
<tr>
<td>rhValues</td>
<td>String</td>
<td>Optional right hand values to use instead of the literal value in evaluating the right hand comparands.</td>
</tr>
<tr>
<td>any</td>
<td>Boolean</td>
<td>If true, any criterion matching suffices for a true result. If false, all criteria must match for a true result.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the supplied values satisfy the criteria, false otherwise.</td>
</tr>
</tbody>
</table>

CSVParser - Scoped

This API provides methods for parsing comma-separated value (CSV) formatted records into an object or an array.

The CSVParser API is in the `sn_impex` namespace.

CSVParser - parseLineToArray(String cvsLine, String delimiter, String quoteCharacter)

Parses passed in CSV formatted content into an array.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>csvLine</td>
<td>String</td>
<td>CSV content to parse.</td>
</tr>
<tr>
<td>delimiter</td>
<td>String</td>
<td>Optional. Character used to delineate the fields in the source CSV content. Default: Comma ','</td>
</tr>
<tr>
<td>quoteCharacter</td>
<td>String</td>
<td>Optional. Character used as the quote character in the source CSV content. Default: Double quote &quot;&quot;</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array containing the parsed values for each element in the passed-in CSV content. For example:</td>
</tr>
</tbody>
</table>

```
{
  Joe,
  Smith,
  470 W Carmen, Chicago IL, 60640
}
```

This example shows simple CSV formatted content parsed into a returned array.

```js
var csvLine = '"Joe","Smith","1470 W Carmen, Chicago IL, 60640"';
var delimiter = ',';
var quoteCharacter = '"';

var x = new sn_impex.CSVParser().parseLineToArray(csvLine, delimiter, quoteCharacter);

gs.log(x[0]);
gs.log(x[1]);
gs.log(x[2]);
```

Output:

```
Joe
Smith
1470 W Carmen, Chicago IL, 60640
```
CSVParser - parseLineToObject(String cvsLine, Array headers, String delimiter, String quoteCharacter)

Parses passed in CSV formatted content into an object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>csvLine</td>
<td>String</td>
<td>CSV content to parse.</td>
</tr>
<tr>
<td>headers</td>
<td>Array of Strings</td>
<td>Headers associated with the CSV content. These headers must be specified in the same order as the corresponding content in <code>cvsLine</code>. For example, \var headers = ['first_name', 'last_name', 'address'];\</td>
</tr>
<tr>
<td>delimiter</td>
<td>String</td>
<td>Optional. Character used to delineate the fields in the source CSV content. Default: Comma ','</td>
</tr>
<tr>
<td>quoteCharacter</td>
<td>String</td>
<td>Optional. Character used as the quote character in the source CSV content. Default: Double quote &quot;&quot;</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing the header and corresponding value for each element in the passed-in CSV content. For example:</td>
</tr>
</tbody>
</table>

```
{
  first_name: Joe,
  last_name: Smith,
  address: 1470 W Carmen, Chicago IL, 60640
}
```

This example shows CSV formatted content parsed into a returned object.

```
var csvLine = '\"Joe\",\"Smith\",\"1470 W Carmen, Chicago IL, 60640\"';
var headers = ['first_name', 'last_name', 'address'];
var delimiter = ',';
var quoteCharacter = '"';
```
var x = new sn_impex.CSVParser().parseLineToObject(csvLine, headers, delimiter, quoteCharacter);

gs.log(x.first_name);
gs.log(x.last_name);
gs.log(x.address);

Output:

Joe
Smith
1470 W Carmen, Chicago IL, 60640

This example shows an exception response because of an invalid pass of the header parameter.

var csvLine = '"Joe","Smith","1470 W Carmen, Chicago IL, 60640"';
var headers = null;
var delimiter = ',';
var quoteCharacter = '"';
try {
var x = new sn_impex.CSVParser().parseLineToObject(csvLine, headers, delimiter, quoteCharacter);
}
catch(ex) {
    gs.info(ex.message);
}

Output:

CSVParser: Header list is empty: no thrown error
*** Script: CSVParser: Header list is empty

CTIOperationRequest - Scoped, Global

The CTIOperationRequest API provides methods to set and get data on the current CTIOperationRequest object.

You use CTIOperationRequest objects to pass information between a message transformer and an operation handler within the Cloud Call Center framework.

Typically, a message transformer is responsible for parsing an incoming CTI-specific payload and setting the payload data on an associated CTIOperationRequest object. An operation handler is then responsible for
getting the information from the CTIOperationRequest object and using the
data to process the requested operation.

The ServiceNow base system provides working operation handlers and message
transformers that enable connection to Amazon Connect. When building
contact flows within Amazon Connect, there are two integration points
between Amazon services and a ServiceNow instance:

- Amazon Web Services (AWS) Lambda Proxy (Invoke AWS Lambda function)
- AWS Lex Bot (Get Customer Input)

You can find the available operation handlers and message transformers for
these integration points in the Operation Handlers [sn_cti_operation_handler]
and Provider Message Transformer [sn_cti_provider_msg_transormer] tables.

A payload is passed into a ServiceNow instance from an external source, such
as from a computer telephony integration (CTI) provider that is making an
operation request. When a request is received, a message transformer parses
the payload and uses the CTIOperationRequest set methods to set payload
values, such as the operation handler name, onto an operation and contact-
specific CTIOperationRequest object.

For example, the following message transformer script parses the passed in JSON
payload and sets the values required by the associated operation handler on
the CTIOperationRequest object.

```javascript
{
    try {
        var jsonPayload = httpRequest.body.data;
        if (jsonPayload) {
            var event = jsonPayload.event;
            if (event) {
                var details = event.Details;
                if (details) {
                    operationRequest.setOperationName(details.Parameters['sn_operation']);
                    var contactData = details.ContactData;
                    if(contactData) {
                        // Call Id should be surfaced in openframe call log so it can be tied to transcription
                        later
                        operationRequest.setParameter('contact.call_id', contactData.ContactId);
                        for(var ck in contactData.CustomerEndpoint) {
                            operationRequest.setParameter('contact.' + ck, contactData.CustomerEndpoint[ck]);
                        }
                        operationRequest.setParameter('contact.id', contactData.ContactId);
                        operationRequest.setParameter('contact.phone',
                        contactData.CustomerEndpoint.Address);
```
```javascript
var parameters = details.Parameters;
if (parameters) {
  for (var pk in parameters) {
    switch(pk) {
      case 'interactionId':
        operationRequest.setInteractionSysId(parameters[pk]);
        break;
      case 'authToken':
        operationRequest.setAuthToken(parameters[pk]);
        break;
      case 'language':
        operationRequest.setLanguage(parameters[pk]);
        break;
      default:
        operationRequest.setParameter(pk, parameters[pk]);
    }
  }

  try {
    // Returns the sys_id of the interaction record.
    var interactionSysId = request.getInteractionSysId();
    var number = request.getParameter('number');
    var addComment = request.getParameter('add_comment');
    addComment = addComment ? addComment.toLowerCase() : addComment;
    addComment = 'yes' === addComment || 'true' === addComment || '1' === addComment;

    } catch(e) {
      ctx.setError(e);
    }
  })(httpRequest, operationRequest, ctx);
```

After the message transformer finishes parsing the passed in payload, the Cloud Call Center framework instantiates the specified operation handler. The operating handler uses the `CTIOperationResponse` API get methods to obtain the information that it needs from the associated `CTIOperationRequest` object to process the requested operation.

For example, the following operation handler script stores values that were set on the `CTIOperationRequest` object in the interaction record associated with the call.

```javascript
(function(/*CTIOperationRequest*/ request, /*CTIOperationResponse*/ response, ctx) {

  // Returns the sys_id of the interaction record.
  var interactionSysId = request.getInteractionSysId();
  var number = request.getParameter('number');
  var addComment = request.getParameter('add_comment');
  addComment = addComment ? addComment.toLowerCase() : addComment;
  addComment = 'yes' === addComment || 'true' === addComment || '1' === addComment;

  })(httpRequest, operationRequest, ctx);
```
```javascript
var workNotes = request.getParameter('work_notes');
var message = gs.getMessage('Thankyou');
var interactionGr = request.getInteractionRecord();
var openedFor = interactionGr ? interactionGr.getValue('opened_for') : null;
if (openedFor && number && addComment && workNotes) {
    var now_GR = new GlideRecordSecure('incident');
    now_GR.addQuery('caller_id', openedFor);
    now_GR.addQuery('number', number);
    now_GR.query();
    if (now_GR.next()) {
        now_GR.work_notes += workNotes;
        now_GR.update();
        message = now_GR.getMessage('Your comment was added');
    }
}
response.setStatusCode(200);
response.setMessage(message);
})(request, response, ctx);
```

For additional information on creating operation handlers, see Configure a contact flow for an automated caller interaction.

This API runs in the sn_cti_core namespace. Before you are able to access the CTIOperationRequest API, the Cloud Call Center Core (sn_cti_core) plugin must be activated. For information on activating the Cloud Call Center, see Install Cloud Call Center applications.

For additional information on the Cloud Call Center, see Cloud Call Center.

**CTIOperationRequest - CTIOperationRequest(String origin)**

Instantiates a CTIOperationRequest object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>origin</td>
<td>String</td>
<td>Optional. Origin of the request. Usually the name of the computer telephony integrator provider. Default: null</td>
</tr>
</tbody>
</table>

```javascript
var request = new sn_cti_core.CTIOperationRequest('AWS Connect');
request.setSessionAttribute('contact.phone', '+15552222345');
request.setSessionAttribute('contact.okToCall', true);
request.setSessionAttribute('contact.address', { |
CTIOperationRequest - getAuthToken()

Returns the authentication token set on the associated CTIOperationRequest object.

This token is used by the Cloud Call Center framework to authenticate the current Cloud Call Center user before executing the requested operation handler if the handler's auth_required flag is set to true. The auth_required flag is a field in the Operation Handler [sn_cti_operation_handler] table. The life of an authentication token should be for the life of the call session, but is determined by the CTI provider.

You can define whatever authentication/authorization handling required by your implementation by creating your own authentication operation handler. Regardless of how the authentication token is generated, the message translator must pass the token back in the CTI payload. Additionally, the CTI provider must store this authentication token locally and pass it in each operation request that requires authentication.

If using the instance provided authenticate operation handler, the handler initiates the creation of the authentication token based on a four-digit user-entered pin. It then sets the authentication token in the sessionAttributes object of the CTIOperationResponse object. The associated message transformer translates the sessionAttributes object into the CTI-specific payload and then sends it to the CTI provider.

⚠️ Note: If the authentication token is not passed, all requests to execute operation handlers with the auth_required flag set will fail. If you do not use authentication, you do not need to maintain authentication tokens.

The ServiceNow base system provides working operation handlers and message transformers that enable connection to Amazon Connect. When building contact flows within Amazon Connect, there are two integration points between Amazon services and a ServiceNow instance:

- Amazon Web Services (AWS) Lambda Proxy (Invoke AWS Lambda function)
- AWS Lex Bot (Get Customer Input)

You can find the available operation handlers and message transformers for these integration points in the Operation Handlers [sn_cti_operation_handler] and Provider Message Transformer [sn_cti_provider_msg_transformer] tables.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The authentication token associated with the current Cloud Call Center user.</td>
</tr>
</tbody>
</table>

```javascript
(function( /*CTIProviderResponseExecutionContext*/ ctx, /*CTIOperationRequest*/ operationRequest, /*CTIOperationResponse*/ operationResponse, /*HTTPResponse*/ httpResponse) {
    var sessionAttributes = {},
        intentResponse;
    var statusCode = -1;
    var message = 'Unprocessed';
    if (operationResponse) {
        statusCode = operationResponse.getStatusCode();
        sessionAttributes = operationResponse.getSessionAttributes();
        message = operationResponse.getMessage();
        var error = operationResponse.getError();
        if (error) {
            sessionAttributes.error = error;
        }
    }
    if (sn_cti_core.CTIRequestDispatcher.Constants.HTTP.Status.AUTH_REQUIRED.code === statusCode) {
        // first call for a secure action without authentication token
        sessionAttributes.statusCode = 401;
        sessionAttributes.message = message;
        intentResponse = {
            sessionAttributes,  // Additional session attributes
            dialogAction: {
                type: 'Close',
                fulfillmentState: 'Fulfilled',
                message: {
                    contentType: 'PlainText',
                    content: 'This operation requires authentication. Say authenticate for authentication.'
                }
            }
        }
    }
})(ctx, operationRequest, operationResponse, httpResponse);
```
} else if ('DialogCodeHook' === operationRequest.getParameter('$$invocationSource')) {
    // pass back authToken and interactionId
    sessionAttributes.authToken = operationRequest.getAuthToken();
    sessionAttributes.interactionId = operationRequest.getInteractionSysId();
    var originalSlots = operationRequest.getParameter('$$slots');
    var responseParameters = operationResponse ? operationResponse.getParameters() : {};
    responseParameters = responseParameters ? responseParameters : {};
    // See if processing happened and we got a dialogAction
    var responseDialogAction = responseParameters['dialogAction'];
    // Default dialog action
    var dialogAction = {
        type: 'Delegate',
        slots: originalSlots
    };
    // If the response doesn't contain an override replay the inputs
    if (responseDialogAction) {
        // We bypassed dialoghook so delegate and pass through inputs
        dialogAction = responseDialogAction;
    }
    intentResponse = {
        sessionAttributes: sessionAttributes,
        dialogAction: dialogAction
    };
} else {
    // Override required session attributes
    sessionAttributes.statusCode = statusCode;
    sessionAttributes.message = message;
    // Build response template
    intentResponse = {
        sessionAttributes: sessionAttributes,
        dialogAction: {
            type: 'Close',
            fulfillmentState: 'Fulfilled',
            message: {
                contentType: 'PlainText',
                content: message
            }
        }
    };
}
CTIOperationRequest - getInteractionRecord()

Returns the interaction GlideRecord associated with the operation request.

An interaction represents a customer request for assistance made through a chat, phone call, or in-person. Interactions can be routed to queues for assignment or assigned to agents directly. Interaction records are stored in the Interaction [interaction] table and can contain any type of data that describes service account interactions with a customer for a specific session.

Interaction records are associated with a CTIOperationRequest object through the CTIOperationRequest - setInteractionSysId(String Id) method. This method is typically called by the message transformer and must be called prior to calling the getInteractionRecord() method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>Interaction GlideRecord object associated with the current CTIOperationRequest object. If the interaction record sys_id has not been set on the CTIOperationRequest object, returns null.</td>
</tr>
</tbody>
</table>

```javascript
(function(/*CTIOperationRequest*/ request, /*CTIOperationResponse*/ response, /*Context*/ ctx) {
    var notes = '', lang = request.getLanguage();
    try {
        // Returns the interaction record (GlideRecord) set on the passed in CTIOperationRequest object
        var interactionGr = request.getInteractionRecord();

        var now_GR = new GlideRecord('sys_user');
        if(interactionGr && now_GR.get(interactionGr.opened_for.toString())) {
            if(now_GR.locked_out == true) {
                notes += gs.getMessageLang("User record found locked. Unlocking the account.
```
now_GR.locked_out=false;
now_GR.update();
response.setStatusCode(200);
response.setMessage(gs.getMessageLang("Your account has been unlocked.", lang));
} else {
    notes += gs.getMessageLang("User record not locked.
", lang);
    response.setStatusCode(200);
    response.setMessage(gs.getMessageLang("Your account doesn't seem to be locked.", lang));
}
} else {
    notes += gs.getMessageLang("User record not found. Transfer to agent
", lang);
    response.setStatusCode(302);
    response.setMessage(gs.getMessageLang("I am unable to find your record. Let me
transfer to someone who can help.", lang));
}
if(interactionGr) {
    interactionGr.work_notes = notes;
    interactionGr.update();
}
} catch(e) {
    ctx.setError(e);
}
})(request, response, ctx);

**CTIOperationRequest - getInteractionSysId()**

Returns the sys_id of the interaction record associated with the current CTIOperationRequest object.

The **CTIOperationRequest - setInteractionSysId(String Id)** method must have been called prior to calling this method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_id of the interaction record set on the current CTIOperationRequest object.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>If the sys_id was not set on the associated CTIOperationRequest object, the method returns null.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
(function( /*CTIOperationRequest*/ request, /*CTIOperationResponse*/ response, ctx) {
    // Returns the sys_id of the interaction record associated with the CTIOperationRequest object.
    var interactionSysId = request.getInteractionSysId();
    var number = request.getParameter('number');
    var addComment = request.getParameter('add_comment');
    addComment = addComment.toLowerCase() || addComment || '1';
    var workNotes = request.getParameter('work_notes');
    var message = gs.getMessage('Thankyou');
    var interactionGr = request.getInteractionRecord();
    var openedFor = interactionGr ? interactionGr.getValue('opened_for') : null;
    if (openedFor && number && addComment && workNotes) {
        var now_GR = new GlideRecordSecure('incident');
        now_GR.addQuery('caller_id', openedFor);
        now_GR.addQuery('number', number);
        now_GR.query();
        if (now_GR.next()) {
            now_GR.work_notes += workNotes;
            now_GR.update();
            message = gs.getMessage('Your comment was added');
        }
    }
    response.setStatusCode(200);
    response.setMessage(message);
})(request, response, ctx);
```

**CTIOperationRequest - getLanguage()**

Returns the ISO 639.1 language code that was set on the current CTIOperationRequest object.

Using the customer preferred language enables you to use platform-provided internationalization and localization capabilities to customize messages and other communications with the associated caller.
Typically an operation handler calls the `getLanguage()` method and the message transformer sets the language on the CTIOperationRequest object by calling the `CTIOperationRequest - setLanguage(String languageCode)` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The ISO 639.1 language code set on the current CTIOperationRequest object. If the language code is not set, returns <code>en</code>.</td>
</tr>
</tbody>
</table>

The following example shows an operation handler calling `getLanguage()` to obtain the caller's preferred language.

```javascript
(function(/*CTIOperationRequest*/ request, /*CTIOperationResponse*/ response) {
    // Obtain the contact's preferred language
    var notes = "", announcement, lang = request.getLanguage();
    try {
        var interactionGr = request.getInteractionRecord();

        announcement = gs.getMessageLang("We are currently experiencing issues with VPN. If you are calling about VPN issues, you may hang up the call.", lang);
        // Use the contact's language to customize the response messages and notes
        if(announcement) {
            notes = gs.getMessageLang("Listened to announcement: {0}", lang, [announcement]);
            response.setStatusCode(200);
        } else {
            announcement = gs.getMessageLang("There are currently no known outage announcements", lang);
            // use the status case as the flag to check for announcements in contact flow
            response.setStatusCode(404);
        }
        response.setMessage(announcement);

        if(interactionGr & notes) {
            interactionGr.work_notes = notes;
            interactionGr.update();
        }
    }
});
```
CTIOperationRequest - getMajorVersion()

Returns the major version of the computer telephony integrator (CTI) software set on the current CTIOperationRequest object.

Use this method if the CTI providers connected to your ServiceNow instance are running multiple versions of their software, as different software versions may require different processing behavior. If the processing behavior is only slightly different between the software versions, it may make sense to have only a single operation handler. You can then handle the processing differences by just checking the version of software making the request and process the request/data within your operation handler. If the required processing is significantly different between versions, it may be more effective to use multiple operation handlers.

Typically the message transformer sets the software version on the CTIOperationRequest object by calling the CTIOperationRequest - setMajorVersion(Number majorVersion) method and an operation handler then consumes the set value using this method. You can also get/set the minor version value of a provider's software using the CTIOperationRequest - getMinorVersion() and CTIOperationRequest - setMinorVersion(Number minorVersion) methods.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Number| Major version number of the CTI software.  
If the major version was never set, returns 1. |
var request = new sn_cti_core.CTIOperationRequest('origin');
var majorVersion = request.getMajorVersion();

**CTIOperationRequest - getMinorVersion()**

Returns the minor version of the computer telephony integrator (CTI) software associated with a CTIOperationRequest object.

Use this method if the CTI providers connected to your ServiceNow instance are running multiple versions of their software, as different software versions may require different processing behavior. If the processing behavior is only slightly different between the software versions, it may make sense to have only a single operation handler. You can then handle the processing differences by just checking the version of software making the request and process the request/data within your operation handler. If the required processing is significantly different between versions, it may be more effective to use multiple operation handlers.

Typically the message transformer sets the software version on the CTIOperationRequest object by calling the `CTIOperationRequest - setMinorVersion(Number minorVersion)` method and an operation handler then consumes the set value using this method. You can also get/set the major version value of a provider's software using the `CTIOperationRequest - getMajorVersion()` and `CTIOperationRequest - setMajorVersion(Number majorVersion)` methods.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Minor version number of the CTI software.</td>
</tr>
<tr>
<td></td>
<td>If the minor version was never set, returns 0.</td>
</tr>
</tbody>
</table>

var request = new sn_cti_core.CTIOperationRequest('origin');
var minorVersion = request.getMinorVersion();
CTIOperationRequest - getOperationName()

Returns the name of the operation that the computer telephony integration (CTI) provider is requesting to execute.

The operation name determines the operation handler that is used to process a request. Typically the message transformer sets the operation name on the CTIOperationRequest object using the CTIOperationRequest - setOperationName(String name) method.

**Note:** The getOperationName() method does not instantiate the associated operation handler. This is done by the Cloud Call Center framework. This method simply returns the associated operation handler name.

**Note:** For Amazon Connect Lex Bot implementations, intent names are appended with a five letter suffix as different Amazon intents cannot have the same intent name. In the Cloud Call Center framework, the intent name equals the operation handler name. An operation handler can either be specific for an intent, having an operation handler name that includes this suffix (myOperation_SUFFX), or it can be generic and used by multiple CTI providers or different use cases, such as using the same handler for processing an HR request and a helpdesk request (myOperation).

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

```javascript
var request = new sn_citi_core.CTIOperationRequest('origin');
var name = request.getOperationName();
```

CTIOperationRequest - getOperationSubStepName()

Returns the substep operation handler name set on the current CTIOperationRequest object.

Use substeps to perform actions such as initialization and validation before processing an operation. Unlike operation handlers that are meant to be
generic and used by all CTI providers, substeps are CTI provider specific and use terminology and naming conventions that are specific to the CTI provider.

The Cloud Call Center framework performs the following steps to locate the required operation handler and any associated substep operation handlers.

1. Tries to locate the operation handler associated with the request.
   - First checks for an operation handler called `myOperation_SUFFX`.
   - If not found, checks for the operation handler `myOperation`. If not found, errors out.

2. If the handler is found, checks the `auth_required` flag on the operation handler. If set, it checks that a valid authentication token (obtained through the `getAuthToken()` method) is present on the request. If present and valid, continue processing, else throw an error.

3. Looks up the operation handler for the passed in substep. For this example let's assume `subStepName=SubStep`.
   - First checks for an operation handler named `myOperation_SUFFX.SubStep`. If found, executes the handler.
   - If not found, checks for the handler `myOperation.SubStep`. If found executes the handler.
   - If not found, checks if a `subStepNotFoundBehavior` was set on the request object.
     - If set, executes the behavior, otherwise, errors out.

4. Executes the primary operation handler (`myOperation_SUFFX` or `myOperation`).

Typically the message transformer sets the substep name on the CTIOperationRequest object using the `CTIOperationRequest - setOperationSubStepName(String name)` method. You can override this behavior if you are creating your own transformer by calling the `CTIOperationRequest - setSubStepNotFoundBehaviour(Object behaviour)` method.

Note: For Amazon Connect Lex Bot implementations, intent names are appended with a five letter suffix as different Amazon intents cannot have the same intent name. In the Cloud Call Center framework, the intent name equals the operation handler name. An operation handler can either be specific for an intent, having an operation handler name that includes this suffix (myOperation_SUFFX), or it can be generic and used by multiple CTI providers or different use cases, such as using the same handler for processing an HR request and a helpdesk request (myOperation).
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the substep operation handler set on the CTIOperationRequest object. If it does not exist, returns null.</td>
</tr>
</tbody>
</table>

```javascript
var request = new sn_cti_core.CTIOperationRequest('origin');
var name = request.getOperationSubStepName();
```

#### CTIOperationRequest - getParameter(String key)

Returns the value of a specified key previously set on the current CTIOperationRequest object.

Using the get/set parameter methods enables the passing of virtually any string or number value between a message transformer and an operation handler. The parameter object can contain zero or more key/value pairs of data that directly correlate to the current operation request. The operation handler determines what key/value pairs are required as it consumes these values. Typically the message transformer sets these parameters on the CTIOperationRequest object by calling the `setParameter(String key, Object value)` method and the operation handler consumes them using this method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>The name of the key value to return.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String or Number</td>
<td>Value of the specified key. If no such key exists, returns null.</td>
</tr>
</tbody>
</table>
function(/*CTIOperationRequest*/ request, /*CTIOperationResponse*/ response, ctx) {
    var interactionSysId = request.getInteractionSysId();
    // Returns the number and add_comments parameters from the CTIOperationRequest object
    var number = request.getParameter('number');
    var addComment = request.getParameter('add_comment');
    addComment = addComment ? addComment.toLowerCase() : addComment;
    addComment = 'yes' === addComment || 'true' === addComment || '1' === addComment;
    var workNotes = request.getParameter('work_notes');
    var message = gs.getMessage('Thankyou');
    var interactionGr = request.getInteractionRecord();
    var openedFor = interactionGr ? interactionGr.getValue('opened_for') : null;
    if (openedFor && number && addComment && workNotes) {
        var now_GR = new GlideRecordSecure('incident');
        now_GR.addQuery('caller_id', openedFor);
        now_GR.addQuery('number', number);
        now_GR.query();
        if (now_GR.next()) {
            now_GR.work_notes += workNotes;
            now_GR.update();
            message = gs.getMessage('Your comment was added');
        }
    }
    response.setStatusCode(200);
    response.setMessage(message);
})(request, response, ctx);

CTIOperationRequest - getParameters()

Returns the key-value pairs for all parameters that were previously set on the current CTIOperationRequest object.

Using the get/set parameter methods enables the passing of virtually any string or number value between a message transformer and an operation handler. The parameter object can contain zero or more key/value pairs of data that directly correlate to the current operation request. The operation handler determines what key/value pairs are required as it consumes these values. Typically, the message transformer sets these parameters on the CTIOperationRequest object by calling the CTIOperationRequest - setParameter(String key, Object value) method and the operation handler consumes them using this method.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>A map of key-value pairs that were previously set on the CTIOperationRequest object. This key-value pairs are free-form and defined by the needs of the operation handler. The returned values are either Strings or Numbers.</td>
</tr>
</tbody>
</table>

```javascript
var request = new sn_cti_core.CTIOperationRequest('origin');
var parms = request.getParameters();
for(var key in parms) {
    gs.info(key + '=' + parms[key]);
}
```

**CTIOperationRequest - getSessionAttribute(String key)**

Returns the value of a specified session attribute key set on the current CTIOperationRequest object.

Using the get/set session attribute methods enables the passing of virtually any string or number value between a message transformer and an operation handler. The sessionAttribute object can contain zero or more key/value pairs of data that are valid for the duration of a computer telephony integration provider defined session, such as the contact's phone number. Session attributes can also be accessed within a contact flow. The operation handler determines what session attribute key/value pairs are needed as it consumes these values. Typically the message transformer sets these attributes on the CTIOperationRequest object by calling the **CTIOperationRequest - setSessionAttribute(String key, Object value)** method. Session attributes are different than parameters in that they persist for the life of the session (such as the entire call) and should be passed back by the CTI provider with each operation request within that call session.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>The name of the key value to return.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String or Number</td>
<td>Value of the specified key. If no such key exists, returns null.</td>
</tr>
</tbody>
</table>

```
var request = new sn_cti_core.CTIOperationRequest('origin');
var phoneAttribute = request.getSessionAttribute('contact.phone');
```

**CTIOperationRequest - getSessionAttributes()**

Returns a key-value pair map of all session attributes set on the current CTIOperationRequest object.

Using the get/set session attribute methods enables the passing of virtually any string or number value between a message transformer and an operation handler. The sessionAttribute object can contain zero or more key/value pairs of data that are valid for the duration of a computer telephony integration provider defined session, such as the contact's phone number. Session attributes can also be accessed within a contact flow. The operation handler determines what session attribute key/value pairs are needed as it consumes these values. Typically the message transformer sets these attributes on the CTIOperationRequest object by calling the `CTIOperationRequest - setSessionAttribute(String key, Object value)` method. Session attributes are different than parameters in that they persist for the life of the session (such as the entire call) and should be passed back by the CTI provider with each operation request within that call session.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>A map of session attribute key-value pairs that were set on the associated CTIOperationResponse object. This map is free-form and defined by the needs of the operation handler.</td>
</tr>
</tbody>
</table>

```javascript
var request = new sn_cti_core.CTIOperationRequest('origin');
var parms = request.getSessionAttributes();
for(var key in parms) {
  gs.info(key + '=' + parms[key]);
}
```

CTIOperationRequest - getSubStepNotFoundBehaviour

Returns the handling behavior for a subsep when a substep's operation handler is not found.

Use substeps to perform actions such as initialization and validation before processing an operation. Unlike operation handlers that are meant to be generic and used by all CTI providers, substeps are CTI provider specific and use terminology and naming conventions that are specific to the CTI provider.

If the requested substep operation handler cannot be found by the Cloud Call Center framework, it attempts to use the "step not found behavior". This behavior is typically set by the message transformer when an operation is initially requested.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OperationNotFoundBehaviours</td>
<td>The behavior/handling to perform when an operation handler cannot be found for the operation/substep combination (&lt;operation_name&gt;.SubStep&gt;). If not defined, returns null. Possible values:</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• OperationNotFoundBehaviours.PassThrough: Do not perform any further processing on the request, just return.</td>
</tr>
<tr>
<td></td>
<td>• OperationNotFoundBehaviours.RouteToOperation: Use the parent operation handler &lt;operation_name&gt;.</td>
</tr>
</tbody>
</table>

```javascript
var request = new sn_cti_core.CTIOperationRequest('origin');
var behavior = request.getSubStepNotFoundBehaviour();
```

**CTIOperationRequest - setAuthToken(String id)**

Sets an authentication token on the current CTIOperationRequest object.

This token is used by the Cloud Call Center framework to authenticate the current Cloud Call Center user before executing the requested operation handler if the handler's `auth_required` flag is set to true. The `auth_required` flag is a field in the Operation Handler [sn_cti_operation_handler] table. The life of an authentication token should be for the life of the call session, but is determined by the CTI provider.

You can define whatever authentication/authorization handling required by your implementation by creating your own authentication operation handler. Regardless of how the authentication token is generated, the message translator must pass the token back in the CTI payload. Additionally, the CTI provider must store this authentication token locally and pass it in each operation request that requires authentication.

If using the instance provided authenticate operation handler, the handler initiates the creation of the authentication token based on a four-digit user-entered pin. It then sets the authentication token in the sessionAttributes object of the CTIOperationResponse object. The associated message transformer translates the sessionAttributes object into the CTI-specific payload and then sends it to the CTI provider.

⚠️ **Note:** If the authentication token is not passed, all requests to execute operation handlers with the `auth_required` flag set will fail. If you do not use authentication, you do not need to maintain authentication tokens.
The ServiceNow base system provides working operation handlers and message transformers that enable connection to Amazon Connect. When building contact flows within Amazon Connect, there are two integration points between Amazon services and a ServiceNow instance:

- Amazon Web Services (AWS) Lambda Proxy (Invoke AWS Lambda function)
- AWS Lex Bot (Get Customer Input)

You can find the available operation handlers and message transformers for these integration points in the Operation Handlers [sn_cti_operation_handler] and Provider Message Transformer [sn_cti_provider_msg_transormer] tables.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>The authentication token for the associated Cloud Call Center user.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example of message transformer script that saves the passed in authentication token.

```javascript
{
  try {
    var jsonPayload = httpRequest.body.data;
    if (jsonPayload) {
      var event = jsonPayload.event;
      if (event) {
        var details = event.Details;
        if (details) {
          operationRequest.setOperationName(details.Parameters['sn_operation']);
          var contactData = details.ContactData;
          if(contactData) {
            // Call id should be surfaced in openframe call log so it can be tied to transcription later
            operationRequest.setParameter('contact.call_id', contactData.ContactId);
            for(var ck in contactData.CustomerEndpoint) {
              operationRequest.setParameter('contact.' + ck, contactData.CustomerEndpoint[ck]);
            }
          }
        }
      }
    }
  }
}
```
operationRequest.setParameter('contact.id', contactData.ContactId);
operationRequest.setParameter('contact.phone',
contactData.CustomerEndpoint.Address);

var parameters = details.Parameters;
if (parameters) {
  for (var pk in parameters) {
    switch(pk) {
      case 'interactionId':
        operationRequest.setInteractionSysId(parameters[pk]);
        break;
      case 'authToken':
        operationRequest.setAuthToken(parameters[pk]);
        break;
      case 'language':
        operationRequest.setLanguage(parameters[pk]);
        break;
      case 'statusCode':
        case 'message':
        break;
      default:
        operationRequest.setParameter(pk, parameters[pk]);
    }
  }
}

try {
  (httpRequest, operationRequest, ctx);
}

} catch(e) {
  ctx.setError(e);
}

CTIOperationRequest - setInteractionSysId(String Id)
Sets the sys_id of the interaction record associated with the operation request on the current CTIOperationRequest object.

If you want to maintain contact interaction information within a contact-specific interaction record, the interaction record sys_id must be maintained for the entire Cloud Call Center call session.

Typically, an operation handler creates an interaction record when a new contact call is initiated. It then sets the interaction record sys_id and table name
on the associated CTIOperationResponse object as session attributes using the
CTIOperationResponse - setSessionAttribute(String key, Object value) method.
The associated method transformer then translates this information into the
computer telephony integration (CTI) provider payload and sends it back to the
CTI provider.

The CTI provider must store the interaction record sys_id for the contact session
locally. Each time the CTI provider interacts with the Cloud Call Center for this
contact session, it must pass back the corresponding interaction record sys_id
in its payload. The receiving message transformer then parses the payload
and uses the CTIOperationRequest - setInteractionSysId() method to set the
interaction record sys_id on the associated CTIOperationRequest object.

The CTIOperationRequest - getInteractionRecord() method uses this sys_id to
obtain the correct interaction record for the contact session. Interaction records
are located in the Interaction [interaction] table. For additional information on
interaction records, see

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The sys_id of the interaction record to associate with the request.</td>
</tr>
</tbody>
</table>

Example of a message transformer script that extracts the interaction record
sys_id from the incoming payload (passed as a parameter) and sets it on the
associated CTIOperationRequest object.

```java
{ 
  try {
    var jsonPayload = httpRequest.body.data;
    if (jsonPayload) {
      var event = jsonPayload.event;
      if (event) {
        var details = event.Details;
        if (details) {
          operationRequest.setOperationName(details.Parameters['sn_operation']);
          var contactData = details.ContactData;
          if(contactData) {
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
// Call id should be surfaced in openframe call log so it can be tied to transcription later
operationRequest.setParameter('contact.call_id', contactData.ContactId);
for(var ck in contactData.CustomerEndpoint) {
    operationRequest.setParameter('contact.' + ck, contactData.CustomerEndpoint[ck]);
}
operationRequest.setParameter('contact.id', contactData.ContactId);
operationRequest.setParameter('contact.phone', contactData.CustomerEndpoint.Address);
}
var parameters = details.Parameters;
if (parameters) {
    for (var pk in parameters) {
        switch(pk) {
            // Associate the existing customer interaction record with the request
            case 'interactionId':
                operationRequest.setInteractionSysId(parameters[pk]);
                break;
            case 'authToken':
                operationRequest.setAuthToken(parameters[pk]);
                break;
            case 'language':
                operationRequest.setLanguage(parameters[pk]);
                break;
            case 'statusCode':
            case 'message':
                break;
        
        default:
            operationRequest.setParameter(pk, parameters[pk]);
        }
    }
} catch(e) {
    ctx.setError(e);
}())(httpRequest, operationRequest, ctx);
CTIOperationRequest - setLanguage(String languageCode)

Sets the ISO 639.1 language code to use when processing the associated operation on a CTIOperationRequest object.

Using the customer preferred language enables you to use platform-provided internationalization and localization capabilities to customize messages and other communications with the associated caller. Typically the message transformer sets the language on the CTIOperationRequest object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>languageCode</td>
<td>String</td>
<td>The ISO 639.1 language code to use when processing the associated operation.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
{
    try {
        var jsonPayload = httpRequest.body.data;
        if (jsonPayload) {
            var event = jsonPayload.event;
            if (event) {
                var details = event.Details;
                if (details) {
                    operationRequest.setOperationName(details.Parameters['sn_operation']);
                    var contactData = details.ContactData;
                    if (contactData) {
                        // Call id should be surfaced in openframe call log so it can be tied to transcription later
                        operationRequest.setParameter('contact.call_id', contactData.ContactId);
                        for (var ck in contactData.CustomerEndpoint) {
                            operationRequest.setParameter('contact.' + ck, contactData.CustomerEndpoint[ck]);
                        }
                        operationRequest.setParameter('contact.id', contactData>ContactData.ContactId);
                        operationRequest.setParameter('contact.phone', contactData.CustomerEndpoint.Address);
                    }
                }
            }
        }
    }
    catch (error) {
        console.error(error);
    }
}
```
var parameters = details.Parameters;
if (parameters) {
  for (var pk in parameters) {
    switch(pk) {
      case 'interactionId':
        operationRequest.setInteractionSysId(parameters[pk]);
        break;
      case 'authToken':
        operationRequest.setAuthToken(parameters[pk]);
        break;
      case 'language':
        operationRequest.setLanguage(parameters[pk]);
        break;
      case 'statusCode':
      case 'message':
        break;
      default:
        operationRequest.setParameter(pk, parameters[pk]);
    }
  }
} catch(e) {
  ctx.setError(e);
}
})(httpRequest, operationRequest, ctx);

CTIOperationRequest - setMajorVersion(Number majorVersion)

Sets the major version of the computer telephony integrator (CTI) provider software making the request on the associated CTIOperationRequest object.

Use this method if the CTI providers connected to your ServiceNow instance are running multiple versions of their software, as different software versions may require different processing behavior. If the processing behavior is only slightly different between the software versions, it may make sense to have only a single operation handler. You can then handle the processing differences by just checking the version of software making the request and process the request/data within your operation handler. If the required processing is significantly different between versions, it may be more effective to use multiple operation handlers.
Typically the message transformer sets the software version on the CTIOperationRequest object by calling this method and an operation handler then consumes the set value using the CTIOperationRequest - getMajorVersion() method. You can also get/set the minor version value of a provider's software using the CTIOperationRequest - getMinorVersion() and CTIOperationRequest - setMinorVersion(Number minorVersion) methods.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>majorVersion</td>
<td>Number</td>
<td>Major version of the CTI provider software making the operation request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this value was not previously set, returns 1.</td>
</tr>
</tbody>
</table>

```
var request = new sn_cti_core.CTIOperationRequest('origin');
request.setMajorVersion(2);
```

**CTIOperationRequest - setMinorVersion(Number minorVersion)**

Sets the minor version of the computer telephony integrator (CTI) provider software making the request on the associated CTIOperationRequest object.

Use this method if the CTI providers connected to your ServiceNow instance are running multiple versions of their software, as different software versions may require different processing behavior. If the processing behavior is only slightly different between the software versions, it may make sense to have only a single operation handler. You can then handle the processing differences by just checking the version of software making the request and process the request/data within your operation handler. If the required processing is significantly different between versions, it may be more effective to use multiple operation handlers.

Typically the message transformer sets the software version on the CTIOperationRequest object by calling this method and an operation handler then consumes the set value using the CTIOperationRequest - getMinorVersion() method. You can also get/set the major version value of a provider's software using the CTIOperationRequest - getMajorVersion() and CTIOperationRequest - setMajorVersion(Number majorVersion) methods.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>minorVersion</td>
<td>Number</td>
<td>Minor version of the CTI provider software making the operation request. If this value was not previously set, returns 0.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var request = new sn_cti_core.CTIOperationRequest('origin');
...
request.setMinorVersion(3);
```

### `CTIOperationRequest - setOperationName(String name)`

Sets the name of the operation that the computer telephony integration (CTI) provider is attempting to execute on the current CTIOperationRequest object.

The operation name determines the operation handler that is used to process the request. Call this method from the associated message transformer.

**Note:** For Amazon Connect Lex Bot implementations, intent names are appended with a five letter suffix as different Amazon intents cannot have the same intent name. In the Cloud Call Center framework, the intent name equals the operation handler name. An operation handler can either be specific for an intent, having an operation handler name that includes this suffix (myOperation_SUFFX), or it can be generic and used by multiple CTI providers or different use cases, such as using the same handler for processing an HR request and a helpdesk request (myOperation).

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the operation that the CTI provider is currently attempting to execute. This name must be the same as the name of the operation handler to use to process the request. If it is not, an error is</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>thrown. You can locate the available operation handlers in the Operation Handler [sn_cti_operation_handler] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
break;
case 'language':
operationRequest.setLanguage(parameters[pk]);
break;
case 'statusCode':
case 'message':
break;

default:
operationRequest.setParameter(pk, parameters[pk]);
}
}
}
}
}
) catch(e) {
ctx.setError(e);
}
})(httpRequest, operationRequest, ctx);

CTIOperationRequest - setOperationSubStepName(String name)

Sets the operation substep name on the current CTIOperationRequest object.

Use substeps to perform actions such as initialization and validation before processing an operation. Unlike operation handlers that are meant to be generic and used by all CTI providers, substeps are CTI provider specific and use terminology and naming conventions that are specific to the CTI provider.

The Cloud Call Center framework performs the following steps to locate the required operation handler and any associated substep operation handlers.

⚠️ Note: For Amazon Connect Lex Bot implementations, intent names are appended with a five letter suffix as different Amazon intents cannot have the same intent name. In the Cloud Call Center framework, the intent name equals the operation handler name. An operation handler can either be specific for an intent, having an operation handler name that includes this suffix (myOperation_SUFFX), or it can be generic and used by multiple CTI providers or different use cases, such as using the same handler for processing an HR request and a helpdesk request (myOperation).
1. Tries to locate the operation handler associated with the request.
   • First checks for an operation handler called `myOperation_SUFFIX`.
   • If not found, checks for the operation handler `myOperation`. If not found, errors out.

2. If the handler is found, checks the `auth_required` flag on the operation handler. If set, it checks that a valid authentication token (obtained through the `getAuthToken()` method) is present on the request. If present and valid, continue processing, else throw an error.

3. Looks up the operation handler for the passed in substep. For this example let's assume `subStepName=SubStep`.
   • First checks for an operation handler named `myOperation_SUFFIX.SubStep`. If found, executes the handler.
   • If not found, checks for the handler `myOperation.SubStep`. If found executes the handler.
   • If not found, checks if a `subStepNotFoundBehavior` was set on the request object.
   • If set, executes the behavior, otherwise, errors out.

4. Executes the primary operation handler (`myOperation_SUFFIX` or `myOperation`.)

You can set the default behavior of a substep by calling the `CTIOperationRequest - setSubStepNotFoundBehaviour(Object behaviour)` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the substep that the CTI provider is trying to execute.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
(function( /*HTTPRequest*/ httpRequest, /*CTIOperationRequest*/ operationRequest, /*Context*/ ctx) {
  try {
    var jsonPayload = httpRequest.body.data;
```
if (jsonPayload) {
    var event = jsonPayload.event;
    if (event) {
        var intent = event.currentIntent;
    var inputTranscript = event.inputTranscript;
    if (inputTranscript) {
        operationRequest.setParameter('$utterance', inputTranscript);
    }
    if (intent) {
        operationRequest.setOperationName(intent.name);
        // Add this because intent name gets changed through the pipeline
        operationRequest.setParameter('$original_intent', intent.name);
        var sessionAttributes = event.sessionAttributes;
        if (sessionAttributes) {
            for (var sk in sessionAttributes) {
                switch (sk) {
                    case 'interactionId':
                        operationRequest.setInteractionSysId(sessionAttributes[sk]);
                        break;
                    case 'authToken':
                        operationRequest.setAuthToken(sessionAttributes[sk]);
                        break;
                    case 'language':
                        operationRequest.setLanguage(sessionAttributes[sk]);
                        break;
                    case 'statusCode':
                    case 'message':
                        break;
                    default:
                        operationRequest.setSessionAttribute(sk, sessionAttributes[sk]);
                        break;
                }
            }
        }
        var slots = intent.slots;
        // Always copy in slots they are the parameters for the operation
        if (slots) {
            for (var pk in slots) {
                operationRequest.setParameter(pk, slots[pk]);
            }
        }
    var invocationSource = event.invocationSource;
    operationRequest.setParameter('$$invocationSource', invocationSource);
    // Specific handling for dialog hooks with a way to override for Amazon specific provider
    }
if (invocationSource === 'DialogCodeHook') {
    // Set substep to trigger forwarding to another handler IF it s present
    operationRequest.setOperationSubStepName(invocationSource);
    // Allow pass through behavior
    var behaviourToUse =
        sn_cti_core.CTIOperationRequest.OperationNotFoundBehaviours.PassThrough;
    operationRequest.setSubStepNotFoundBehaviour(behaviourToUse);
    // Reserved - copy slots in so we can just use delegate as default if the call isn't handled
    operationRequest.setParameter('$$slots', slots);
}

} catch (e) {
    ctx.setError(e);
}
return operationRequest;

CTIOperationRequest - setParameter(String key, Object value)
Sets the specified key-value pair on the parameter object of the current CTIOperationRequest object.

Using the get/set parameter methods enables the passing of virtually any string or number value between a message transformer and an operation handler. The parameter object can contain zero or more key/value pairs of data that directly correlate to the current operation request. The operation handler determines what key/value pairs are required as it consumes these values. Typically the message transformer sets these parameters on the CTIOperationRequest object by calling this method and the operation handler consumes them using the CTIOperationRequest - getParameter(String key) or CTIOperationRequest - getParameters() method.

⚠️ **Note:** You should only store objects that can survive a JSON.parse(JSON.stringify(object)) operation. Objects not meeting this criteria may not propagate correctly through the entire operation processing chain.

For example:

```javascript
var x = {
    "string": 'abc',
    "int": 123,
};
```
```javascript
"float": 1.234,
"number": new Number(3),
"boolean": true,
"date": new Date(2006, 0, 2, 15, 4, 5),
"object": {
  "string": 'abc',
  "int": 123,
  "float": 1.234,
  "number": new Number(3),
  "boolean": true,
  "date": new Date(2006, 0, 2, 15, 4, 5)
},
"function": function(abc) {
  console.log(abc);
}
}
var stringify = JSON.stringify(x);
gs.info(stringify);
var hydrate = JSON.parse(stringify);
var stringify2 = JSON.stringify(hydrate);
gs.info(stringify2);
```

Produces:

```javascript
```

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the key under which to store the associated value.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>Value to store. Valid data types:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- string</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- int</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- float</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- number</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>boolean</td>
<td></td>
</tr>
<tr>
<td></td>
<td>date</td>
<td></td>
</tr>
<tr>
<td></td>
<td>object</td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
```java
case 'authToken':
    operationRequest.setAuthToken(parameters[pk]);
    break;

case 'language':
    operationRequest.setLanguage(parameters[pk]);
    break;

case 'statusCode':
    case 'message':
        break;

default:
    operationRequest.setParameter(pk, parameters[pk]);
}
```

### CTIOperationRequest - `setSessionAttribute(String key, Object value)`

Sets the specified session attribute key-value pair on the current CTIOperationRequest object.

Using the get/set session attribute methods enables the passing of virtually any string or number value between a message transformer and an operation handler. The `sessionAttribute` object can contain zero or more key/value pairs of data that are valid for the duration of a computer telephony integration provider defined session. Session attributes can also be accessed within a contact flow. The operation handler determines what session attribute key/value pairs are needed as it consumes these values. Typically the message transformer sets these attributes on the CTIOperationRequest object by calling this method and the operation handler gets the attributes using the `CTIOperationRequest - getSessionAttribute(String key)` or `CTIOperationRequest - getSessionAttributes()` method.

**Note:** You should only store objects that can survive a `JSON.parse(JSON.stringify(object))` operation. Objects not meeting this criteria may not propagate correctly through the entire operation processing chain.
For example:

```javascript
var x = {
    "string": 'abc',
    "int": 123,
    "float": 1.234,
    "number": new Number(3),
    "boolean": true,
    "date": new Date(2006, 0, 2, 15, 4, 5),
    "object": {
        "string": 'abc',
        "int": 123,
        "float": 1.234,
        "number": new Number(3),
        "boolean": true,
        "date": new Date(2006, 0, 2, 15, 4, 5)
    },
    "function": function(abc) {
    }
};

var stringify = JSON.stringify(x);
gs.info(stringify);
var hydrate = JSON.parse(stringify);
var stringify2 = JSON.stringify(hydrate);
gs.info(stringify2);
```

Produces:

```
```

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the key under which to store the associated value.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>Value to store.</td>
</tr>
</tbody>
</table>

Valid data types:
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>• string</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• int</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• float</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• boolean</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• date</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• object</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var request = new sn_cti_core.CTIOperationRequest('origin');
request.setSessionAttribute('contact.phone', '+15552222345');
request.setSessionAttribute('contact.okToCall', true);
request.setSessionAttribute('contact.address', {
    street: '1234 Main St'
    city: 'API City',
    zip: 91335
});
```

CTIOperationRequest - setSubStepNotFoundBehaviour(Object behaviour)

Sets the behavior to perform if the current substep's operation handler is not found.

This method should be called by the input message transformer.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| behaviour  | OperationNotFoundBehaviours | Behavior to use if a handling behavior is not specified for the substep. This must be a behavior defined on the
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| CTIOperationRequest.OperationNotFoundBehaviours | object | Possible values:  
  • ThrowError: Throw an error.  
  • PassThrough: Do not perform any further processing on the request, just return.  
  • RouteToOperation: Use the parent operation handler `<operation_name>`.

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
(function (/*HTTPRequest*/ httpRequest, /*CTIOperationRequest*/ operationRequest, /*Context*/ ctx) {
  try {
    var jsonPayload = httpRequest.body.data;
    if (jsonPayload) {
      var event = jsonPayload.event;
      if (event) {
        var intent = event.currentIntent;
        var inputTranscript = event.inputTranscript;
        if (inputTranscript) {
          operationRequest.setParameter('utterance', inputTranscript);
        }
        if (intent) {
          operationRequest.setOperationName(intent.name);
          // Add this because intent name gets changed through the pipeline
          operationRequest.setParameter('original_intent', intent.name);
          var sessionAttributes = event.sessionAttributes;
          if (sessionAttributes) {
            for (var sk in sessionAttributes) {
              switch (sk) {
                case 'interactionId':
                  operationRequest.setInteractionSysId(sessionAttributes[sk]);
                  break;
              }
            }
          }
        }
      }
    }
  }
});
```
case 'authToken':
    operationRequest.setAuthToken(sessionAttributes[sk]);
    break;

case 'language':
    operationRequest.setLanguage(sessionAttributes[sk]);
    break;

    case 'status':
    case 'message':
        break;

    default:
        operationRequest.setSessionAttribute(sk, sessionAttributes[sk]);
        break;

var slots = intent.slots;
// Always copy in slots they are the parameters for the operation
if (slots) {
    for (var pk in slots) {
        operationRequest.setParameter(pk, slots[pk]);
    }
}

var invocationSource = event.invocationSource;
operationRequest.setParameter('$$invocationSource', invocationSource);
// Specific handling for dialog hooks with a way to override for Amazon specific
provider
    if (invocationSource === 'DialogCodeHook') {
        // Set substep to trigger forwarding to another handler IF it s present
        operationRequest.setOperationSubStepName(invocationSource);
        // Allow pass through behavior
        var behaviourToUse = sn_cti_core.CTIOperationRequest.OperationNotFoundBehaviours.PassThrough;
        operationRequest.setSubStepNotFoundBehaviour(behaviourToUse);
        // Reserved - copy slots in so we can just use delegate as default if the call isn't
        handled
        operationRequest.setParameter('$$slots', slots);
    }
}

try {
    // Act
}

} catch (e) {
    ctx.setError(e);
}
CTIOperationResponse - Scoped, Global

The CTIOperationResponse API provides methods to set and get data on the current CTIOperationResponse object.

You use CTIOperationResponse objects to pass information between an operation handler and a message transformer within the Cloud Call Center framework.

Typically, an operation handler is responsible for setting values on the CTIOperationResponse object as it processes the associated request. A message transformer is then responsible for getting the values from the CTIOperationRequest object and building the CTI-specific payload that is sent back to the computer telephony integrator (CTI) provider.

The ServiceNow base system provides working operation handlers and message transformers that enable connection to Amazon Connect. When building contact flows within Amazon Connect, there are two integration points between Amazon services and a ServiceNow instance:

- Amazon Web Services (AWS) Lambda Proxy (Invoke AWS Lambda function)
- AWS Lex Bot (Get Customer Input)

You can find the available operation handlers and message transformers for these integration points in the Operation Handlers [sn_cti_operation_handler] and Provider Message Transformer [sn_cti_provider_msg_transformer] tables.

The following operation handler script shows how to use various CTIOperationResponse methods to set values on the CTIOperationResponse object.

```javascript
(function(/*CTIOperationRequest*/ request, /*CTIOperationResponse*/ response) {
    try {
        var state = request.getParameter('$state');
        if(!state) {
            state = 'new';
        }
        var phone = request.getParameter('contact.phone');
        var username = request.getParameter('contact.username');
        var someDataPresent = phone || username;
        if(!someDataPresent) {
            throw 'phone or username must be supplied';
        }
    return operationRequest;
})(httpRequest, operationRequest, ctx);
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
if('new' === state) {
    var userGr = null;
    if(phone || username) {
        userGr = new GlideRecordSecure('sys_user');
        var qc = null;
        if(phone) {
            qc = userGr.addQuery('phone', phone);
            qc.addOrCondition('mobile_phone', phone);
            qc.addOrCondition("home_phone", phone);
        }
        if(username) {
            if(qc) {
                qc.addOrCondition('user_name', username);
            } else {
                qc = userGr.addQuery('user_name', username);
            }
        }
        // If there was criteria then run the query
        if(qc) {
            userGr.query();
        } else {
            userGr = null;
        }
    }
}

var foundUser = userGr && userGr.next();

var phonelogGr = new GlideRecord('sn_openframe_phone_log');
phonelogGr.initialize();
phonelogGr.setValue('call_id', request.getParameter('contact.id'));
phonelogGr.setValue('phone_number', phone);
if(foundUser) {
    phonelogGr.setValue('contact', userGr.getUniqueValue());
}
phonelogGr.insert();

var interactionGr = new GlideRecordSecure('interaction');
interactionGr.initialize();
interactionGr.setValue('type', 'phone');
interactionGr.setValue('short_description', 'User Contact via Phone: ' + phone);
interactionGr.setValue('channel_metadata_table', 'sn_openframe_phone_log');
interactionGr.setValue('channel_metadata_document', phonelogGr.getUniqueValue());
if(foundUser) {
interactionGr.setValue('opened_for', userGr.getUniqueValue());
response.setSessionAttribute('snc_user_first_name', userGr.getValue('first_name'));
response.setSessionAttribute('snc_user_last_name', userGr.getValue('last_name'));
response.setSessionAttribute('snc_user_sys_id', userGr.getUniqueValue());
}
else {
response.setSessionAttribute('snc_user_sys_id', "USER_NOT_FOUND");
}
interactionGr.setValue('state', 'new');

var interactionId = interactionGr.insert();

response.setStatusCode(200);
response.setMessage('success');
response.setSessionAttribute('interactionTable', 'interaction');
response.setSessionAttribute('interactionId', interactionId);

})

} catch(e) {
ctx.setError(e);
}

})(request, response);

For additional information on creating operation handlers, see Configure a contact flow for an automated caller interaction.

The following message transformer script gets the values set on the CTIOperationResponse object and stores them in the CTI-specific payload that is then sent back to the CTI provider.

(function /*CTIOperationResponse*/operationResponse, /*HTTPResponse*/ httpResponse) {
// AWS Connect Lambda expects name value pairs only
var out = {};
for(var k in operationResponse.getSessionAttributes()) {
    out[k] = operationResponse.getSessionAttributes()[k];
}
out.statusCode = operationResponse.getStatusCode();
out.message = operationResponse.getMessage();
httpResponse.setBody(out);
})(operationResponse, httpResponse);

This API runs in the sn_cti_core namespace. Before you are able to access the CTIOperationRequest API, the Cloud Call Center Core (sn_cti_core) application must be installed. For information on this installation, see Install Cloud Call Center applications.
For additional information on the Cloud Call Center, see Cloud Call Center.

CTIOperationResponse - CTIOperationResponse(String origin)

Instantiates a CTIOperationResponse object.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```java
var response = new sn_cti_core.CTIOperationResponse();
response.setMinorVersion(3);
```

CTIOperationResponse - getAuthToken()

Returns the authentication token that was set on the CTIOperationResponse object.

This token is used by the Cloud Call Center framework to authenticate the current Cloud Call Center user before executing the requested operation handler if the handler's `auth_required` flag is set to true. The `auth_required` flag is a field in the Operation Handler `[sn_cti_operation_handler]` table. The life of an authentication token should be for the life of the call session, but is determined by the CTI provider.

You can define whatever authentication/authorization handling required by your implementation by creating your own authentication operation handler. Regardless of how the authentication token is generated, the message translator must pass the token back in the CTI payload. Additionally, the CTI provider must store this authentication token locally and pass it in each operation request that requires authentication.

If using the instance provided authenticate operation handler, the handler initiates the creation of the authentication token based on a four-digit user-entered pin. It then sets the authentication token in the sessionAttributes object of the CTIOperationResponse object. The associated message transformer translates the sessionAttributes object into the CTI-specific payload and then sends it to the CTI provider.

⚠️ Note: If the authentication token is not passed, all requests to execute operation handlers with the `auth_required` flag set will fail. If you do not use authentication, you do not need to maintain authentication tokens.

The ServiceNow base system provides working operation handlers and message transformers that enable connection to Amazon Connect. When building
contact flows within Amazon Connect, there are two integration points between Amazon services and a ServiceNow instance:

- Amazon Web Services (AWS) Lambda Proxy (Invoke AWS Lambda function)
- AWS Lex Bot (Get Customer Input)

You can find the available operation handlers and message transformers for these integration points in the Operation Handlers [sn.cti_operation_handler] and Provider Message Transformer [sn.cti_provider_msg_transformer] tables.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

```javascript
var response = new sn.cti_core.CTIOperationResponse();
var token = response.getAuthToken();
```

**CTIOperationResponse - getError()**

Returns the error object associated with the current CTIOperationResponse object.

You might use this information to formulate the payload returned to the computer telephony integration (CTI) provider.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Operation error information. The format of this object is determined by the CTI provider. If an error does not exist, returns null.</td>
</tr>
</tbody>
</table>

```
(function( /*CTIProviderResponseExecutionContext*/ ctx, /*CTIOperationRequest*/ operationRequest, /*CTIOperationResponse*/ operationResponse, /*HTTPResponse*/ httpResponse) {

    var sessionAttributes = {},
        intentResponse;
    var statusCode = -1;
    var message = 'Unprocessed';
    if (operationResponse) {
        statusCode = operationResponse.getStatusCode();
        sessionAttributes = operationResponse.getSessionAttributes();
        message = operationResponse.getMessage();
        var error = operationResponse.getError();
        if (error) {
            sessionAttributes.error = error;
        }
    }

    if (sn_cti_core.CTIRequestDispatcher.Constants.HTTP.Status.AUTH_REQUIRED.code === statusCode) {
        // first call for a secure action without authentication token
        sessionAttributes.statusCode = 401;
        sessionAttributes.message = message;
        intentResponse = {
            sessionAttributes: sessionAttributes,
            dialogAction: {
                type: 'Close',
                fulfillmentState: 'Fulfilled',
                message: {
                    contentType: 'PlainText',
                    content: 'This operation requires authentication. Say authenticate for authentication.'
                }
            }
        };
    } else if ('DialogCodeHook' === operationRequest.getParameter('$$invocationSource')) {
        // pass back authToken and interactionId
        sessionAttributes.authToken = operationRequest.getAuthToken();
    }
}
```
sessionAttributes.interactionId = operationRequest.getInteractionSysId();
var originalSlots = operationRequest.getParameter('$$slots');
var responseParameters = operationResponse ? operationResponse.getParameters() : {};
responseParameters = responseParameters ? responseParameters : {};

// See if we processing happened and we got a dialogAction
var responseDialogAction = responseParameters['dialogAction'];

// Default dialog action
var dialogAction = {
    type: 'Delegate',
    slots: originalSlots
};

// If the response doesn't contain an override replay the inputs
if (responseDialogAction) {
    // We bypassed dialoghook so delegate and pass through inputs
dialogAction = responseDialogAction;
}
intentResponse = {
    sessionAttributes: sessionAttributes,
    dialogAction: dialogAction
};
} else {
    // Override required session attributes
    sessionAttributes.statusCode = statusCode;
    sessionAttributes.message = message;

    // Build response template
    intentResponse = {
        sessionAttributes: sessionAttributes,
        dialogAction: {
            type: 'Close',
            fulfillmentState: 'Fulfilled',
            message: {
                contentType: 'PlainText',
                content: message
            }
        }
    }
}
httpResponse.setBody(intentResponse);})}(ctx, operationRequest, operationResponse, httpResponse);
CTIOperationResponse - getInteractionSysId()

Returns the sys_id of the interaction record associated with the call session that was set on the CTIOperationResponse object.

The CTIOperationResponse - setInteractionSysId(String Id) method must have been called prior to calling this method.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_id of the interaction record set on the CTIOperationResponse object. Interaction records are located in the Interaction [interaction] table. If the interaction record sys_id is not set, the method returns null.</td>
</tr>
</tbody>
</table>

var request = new sn_cti_core.CTIOperationResponse();
var interactionId = request.getInteractionSysId();

CTIOperationResponse - getMajorVersion()

Returns the major version of the computer telephony integrator (CTI) software set on the current CTIOperationResponse object.

Use this method if the CTI providers connected to your ServiceNow instance are running multiple versions of their software, as different software versions may require different payloads. If the payload is only slightly different between the software versions, it may make sense to have only a single message transformer. You can then handle the payload differences by just checking the version of software making the request within your message transformer. If the required payload is significantly different between versions, it may be more effective to use multiple message transformers.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Major version number of the CTI software.</td>
</tr>
<tr>
<td></td>
<td>If the major version was never set, returns 1.</td>
</tr>
</tbody>
</table>

```javascript
var request = new sn_cti_core.CTIOperationResponse();
var majorVersion = response.getMajorVersion();
```

### CTIOperationResponse - getMessage()

Returns the message string set on the current CTIOperationResponse object.

Typically this is a message that is spoken back to the user once an operation is complete. You can set this value using the `CTIOperationResponse - setMessage(String message)` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Message text. If no message exists, returns null.</td>
</tr>
</tbody>
</table>

```javascript
(function /*CTIOperationResponse*/operationResponse, /*HTTPResponse*/ httpResponse) {
  // AWS Connect Lambda expects name value pairs only
  var out = {};
  for(var k in operationResponse.getSessionAttributes()) {
    out[k] = operationResponse.getSessionAttributes()[k];
  }
  out.statusCode = operationResponse.getStatusCode();
  out.message = operationResponse.getMessage();
  httpResponse.setBody(out);
})(operationResponse, httpResponse);
```
CTIOperationResponse - getMinorVersion()

Returns the minor version of the computer telephony integrator (CTI) software set on the current CTIOperationResponse object.

Use this method if the CTI providers connected to your ServiceNow instance are running multiple versions of their software, as different software versions may require different payloads. If the payload is only slightly different between the software versions, it may make sense to have only a single message transformer. You can then handle the payload differences by just checking the version of software making the request within your message transformer. If the required payload is significantly different between versions, it may be more effective to use multiple message transformers.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Minor version number of the payload associated with this interaction. If the minor version was never set, returns 0.</td>
</tr>
</tbody>
</table>

```javascript
var request = new sn_cti_core.CTIOperationResponse();
var minorVersion = request.getMinorVersion();
```

CTIOperationResponse - getParameter(String key)

Returns the value of a specified key previously set on the current CTIOperationResponse object.

Using the get/set parameter methods enables the passing of virtually any string or number value between an operation handler and a message transformer. The parameter object can contain zero or more key/value pairs of data that directly correlate to the current operation response. The operation handler determines what key/value pairs are required as it produces these values. Use the CTIOperationResponse - setParameter(String key, Object value) method to set parameters on the current CTIOperationResponse object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>The name of the key value to return.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The value of the passed in key. If no such key exists, returns null.</td>
</tr>
</tbody>
</table>

```javascript
var request = new sn_cti_core.CTIOperationResponse();
var parm = request.getParameter('contact.phone');
```

**CTIOperationResponse - getParameters()**

Returns the key-value pairs for all parameters that were previously set on the current CTIOperationResponse object.

Using the get/set parameter methods enables the passing of virtually any string or number value between an operation handler and a message transformer. The parameter object can contain zero or more key/value pairs of data that directly correlate to the current operation response. The operation handler determines what key/value pairs are required as it produces these values. Use the `CTIOperationResponse - setParameter(String key, Object value)` method to set parameters on the current CTIOperationResponse object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>A map of key-value pairs that were previously set on the CTIOperationResponse object. This key-value pairs are free-form and defined by the needs of the operation handler. The returned values are either Strings or Numbers.</td>
</tr>
</tbody>
</table>
function( /*CTIProviderResponseExecutionContext*/ ctx, /*CTIOperationRequest*/ operationRequest, /*CTIOperationResponse*/ operationResponse, /*HTTPResponse*/ httpResponse) {
  var sessionAttributes = {},
      intentResponse;
  var statusCode = -1;
  var message = 'Unprocessed';
  if (operationResponse) {
    statusCode = operationResponse.getStatusCode();
    sessionAttributes = operationResponse.getSessionAttributes();
    message = operationResponse.getMessage();
    var error = operationResponse.getError();
    if (error) {
      sessionAttributes.error = error;
    }
  }
  if (sn_cti_core.CTIRequestDispatcher.Constants.HTTP.Status.AUTH_REQUIRED.code === statusCode) {
    //first call for a secure action without authentication token
    sessionAttributes.statusCode = 401;
    sessionAttributes.message = message;
    intentResponse = {
      sessionAttributes: sessionAttributes,
      dialogAction: {
        type: 'Close',
        fulfillmentState: 'Fulfilled',
        message: {
          contentType: 'PlainText',
          content: 'This operation requires authentication. Say authenticate for authentication.'
        }
      }
    };
  } else if ('DialogCodeHook' === operationRequest.getParameter('$$invocationSource')) {
    //pass back authToken and interactionId
    sessionAttributes.authToken = operationRequest.getAuthToken();
    sessionAttributes.interactionId = operationRequest.getInteractionSysId();
    var originalSlots = operationRequest.getParameter('$$slots');
    var responseParameters = operationResponse ? operationResponse.getParameters() : {};
    responseParameters = responseParameters ? responseParameters : {};
    // See if processing happened and we got a dialogAction
    var responseDialogAction = responseParameters['dialogAction'];
    // Default dialog action
var dialogAction = {
    type: 'Delegate',
    slots: originalSlots
};
// If the response doesn't contain an override replay the inputs
if (responseDialogAction) {
    // We bypassed dialoghook so delegate and pass through inputs
dialogAction = responseDialogAction;
}
intentResponse = {
    sessionAttributes: sessionAttributes,
    dialogAction: dialogAction
};
} else {
// Override required session attributes
    sessionAttributes.statusCode = statusCode;
    sessionAttributes.message = message;
// Build response template
intentResponse = {
    sessionAttributes: sessionAttributes,
    dialogAction: {
        type: 'Close',
        fulfillmentState: 'Fulfilled',
        message: {
            contentType: 'PlainText',
            content: message
        }
    }
};
}
httpResponse.setBody(intentResponse);
})(ctx, operationRequest, operationResponse, httpResponse);

**CTIOperationResponse - getSessionAttribute(String key)**

Returns the value for a specified session attribute key set on the current CTIOperationResponse object.

Using the get/set session attribute methods enables the passing of virtually any string or number value between an operation handler and a message transformer. Session attributes can also be accessed within a contact flow. The sessionAttribute object can contain zero or more key/value pairs of data, such as the contact's phone number, that are valid for the duration of a computer telephony integration provider defined session. The operation handler
determines what session attribute key/value pairs are needed as it utilizes these values during processing. Typically the operation handler sets these attributes on the CTIOperationResponse object by calling the `CTIOperationResponse - setSessionAttribute(String key, Object value)` method and the message transformer consumes the attributes using this method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>The name of the key value to return.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the passed in key. If no such key exists, returns null.</td>
</tr>
</tbody>
</table>

```javascript
var response = new sn_cti_core.CTIOperationResponse();
var phoneAttribute = response.getSessionAttribute('contact.phone');
```

### CTIOperationResponse - getSessionAttributes()

Returns the key-value pairs for all session attributes that were previously set on the current CTIOperationResponse object.

Using the get/set session attribute methods enables the passing of virtually any string or number value between an operation handler and a message transformer. Session attributes can also be accessed within a contact flow. The sessionAttribute object can contain zero or more key/value pairs of data, such as the contact's phone number, that are valid for the duration of a computer telephony integration provider defined session. The operation handler determines what session attribute key/value pairs are needed as it utilizes these values during processing. Typically the operation handler sets these attributes on the CTIOperationResponse object by calling the `CTIOperationResponse - setSessionAttribute(String key, Object value)` method and the message transformer consumes the attributes using this method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>A map of key-value pairs that were previously set on the CTIOperationResponse object. This key-value pairs are free-form and defined by the needs of the operation handler. The returned values are either Strings or Numbers.</td>
</tr>
</tbody>
</table>

```javascript
(function(/*CTIOperationResponse*/operationResponse, /*HTTPResponse*/ httpResponse) {
    // AWS Connect Lambda expects name value pairs only
    var out = {};
    for(var k in operationResponse.getSessionAttributes()) {
        out[k] = operationResponse.getSessionAttributes()[k];
    }
    out.statusCode = operationResponse.getStatusCode();
    out.message = operationResponse.getMessage();
    httpResponse.setBody(out);
})(operationResponse, httpResponse);
```

**CTIOperationResponse - getStatusCode()**

Returns the status code set on the current CTIOperationResponse object.

These status codes should correspond with standard HTTP status codes, such as 200 for success, 401 for authorization required. Typically, the resultant status codes are determined by the operation handler and are set on the CTIOperationResponse object using the `CTIOperationResponse - setStatusCode(Number statusCode)` method. The message transformer then uses this method to include the status code in the payload returned to the computer telephony integrator provider.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>(207,667,217,681)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Status code of the current operation. If a status code has not been set, returns -1.</td>
</tr>
</tbody>
</table>
(function /*CTIOperationResponse*/operationResponse, /*HTTPResponse*/ httpResponse) {
    // AWS Connect Lambda expects name value pairs only
    var out = {};
    for(var k in operationResponse.getSessionAttributes()) {
        out[k] = operationResponse.getSessionAttributes()[k];
    }
    out.statusCode = operationResponse.getStatusCode();
    out.message = operationResponse.getMessage();
    httpResponse.setBody(out);
})(operationResponse, httpResponse);

CTIOperationResponse - setAuthToken(String token)
Sets an authentication token on the current CTIOperationResponse object.

This token is used by the Cloud Call Center framework to authenticate the current Cloud Call Center user before executing the requested operation handler if the handler's auth_required flag is set to true. The auth_required flag is a field in the Operation Handler [sn_cti_operation_handler] table. The life of an authentication token should be for the life of the call session, but is determined by the CTI provider.

You can define whatever authentication/authorization handling required by your implementation by creating your own authentication operation handler. Regardless of how the authentication token is generated, the message translator must pass the token back in the CTI payload. Additionally, the CTI provider must store this authentication token locally and pass it in each operation request that requires authentication.

If using the instance provided authenticate operation handler, the handler initiates the creation of the authentication token based on a four-digit user-entered pin. It then sets the authentication token in the sessionAttributes object of the CTIOperationResponse object. The associated message transformer translates the sessionAttributes object into the CTI-specific payload and then sends it to the CTI provider.

⚠️ Note: If the authentication token is not passed, all requests to execute operation handlers with the auth_required flag set will fail. If you do not use authentication, you do not need to maintain authentication tokens.

The ServiceNow base system provides working operation handlers and message transformers that enable connection to Amazon Connect. When building contact flows within Amazon Connect, there are two integration points between Amazon services and a ServiceNow instance:
- Amazon Web Services (AWS) Lambda Proxy (Invoke AWS Lambda function)
- AWS Lex Bot (Get Customer Input)

You can find the available operation handlers and message transformers for these integration points in the Operation Handlers [sn_cti_operation_handler] and Provider Message Transformer [sn_cti_provider_msg_transormer] tables.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The authentication token to use to authenticate operations that have the <code>auth_required</code> flag set.</td>
</tr>
</tbody>
</table>

```javascript
var response = new sn_cti_core.CTIOperationResponse();
var authenticate = '<Your authentication code here>'; // Authenticate the user
if (authenticated) {
  var claims = {
    "interactionId":interactionGr.sys_id.toString()
  };
  var jwt = new sn_cti_core.AuthHelpers().generateJWT(claims);
  response.setAuthToken(jwt);
}
```

#### CTIOperationResponse - setError(Object error)

Sets the error object in the current CTIOperationResponse object.

You can use the error object in the component to formulate the payload to send to the computer telephony integrator (CTI) provider.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Object</td>
<td>Information about the error that was detected while processing the requested operation. The format of this object is determined by the CTI provider.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var response = new sn_cti_core.CTIOperationResponse();
response.setError(new Error("You did not supply a field"));
```

`CTIOperationResponse - setInteractionSysId(String Id)`

Sets the sys_id of the interaction record associated with the operation request on the current CTIOperationResponse object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The sys_id to set for the current interaction.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var response = new sn_cti_core.CTIOperationResponse();
response.setInteractionSysId('98529cc55380001048e5ddeff7b120b');
```

`CTIOperationResponse - setMajorVersion(Number majorVersion)`

Sets the major version of the computer telephony integrator (CTI) provider software making the request on the associated CTIOperationResponse object.

Use this method if the CTI providers connected to your ServiceNow instance are running multiple versions of their software, as different software versions may require different payloads. If the payload is only slightly different between the software versions, it may make sense to have only a single message transformer. You can then handle the payload differences by just checking the version of software making the request within your message transformer. If the required payload is significantly different between versions, it may be more effective to use multiple message transformers.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>majorVersion</td>
<td>Number</td>
<td>Major version of the CTI provider software making the operation request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this value was not previously set, returns 1.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var request = new sn_cti_core.CTIOperationResponse();
request.setMajorVersion(2);
```

`CTIOperationResponse - setMessage(String message)`

Sets a text message on the current `CTIOperationResponse` object.

Typically this is a message that is spoken back to the user once an operation is complete and is set by the associated operation handler. The message transformer then uses the `CTIOperationResponse - getMessage()` method to obtain the message and pass it back to the computer telephony integration provider.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Message text</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
(function(/*CTIOperationRequest*/ request, /*CTIOperationResponse*/ response, /*Context*/ ctx) {
    var notes = '', lang = request.getLanguage();
    try {
```
var interactionGr = request.getInteractionRecord();

var now_GR = new GlideRecord('sys_user');
if(interactionGr && now_GR.get(interactionGr.opened_for.toString())) {
    if(now_GR.locked_out == true) {
        notes += gs.getMessageLang("User record found locked. Unlocking the account.\n", lang);
        now_GR.locked_out=false;
        now_GR.update();
        response.setStatusCode(200);
        response.setMessage(gs.getMessageLang("Your account has been unlocked.", lang));
    } else {
        notes += gs.getMessageLang("User record not locked.\n", lang);
        response.setStatusCode(200);
        response.setMessage(gs.getMessageLang("Your account doesn't seem to be locked.\", lang));
    }
} else {
    notes += gs.getMessageLang("User record not found. Transfer to agent\n", lang);
    response.setStatusCode(302);
    response.setMessage(gs.getMessageLang("I am unable to find your record. Let me transfer to someone who can help.\", lang));
}
if(interactionGr) {
    interactionGr.work_notes = notes;
    interactionGr.update();
}
} catch(e) {
    ctx.setError(e);
}
})(request, response, ctx);

CTIOperationResponse - setMinorVersion(Number minorVersion)

Sets the minor version of the computer telephony integrator (CTI) provider software making the request on the associated CTIOperationResponse object.

Use this method if the CTI providers connected to your ServiceNow instance are running multiple versions of their software, as different software versions may require different payloads. If the payload is only slightly different between the software versions, it may make sense to have only a single message transformer. You can then handle the payload differences by just checking the version of software making the request within your message transformer. If the required
payload is significantly different between versions, it may be more effective to use multiple message transformers.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>minorVersion</td>
<td>Number</td>
<td>Minor version of the CTI provider software making the operation request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this value was not previously set, returns 0.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var response = new sn_cti_core.CTIOperationResponse();
response.setMinorVersion(3);
```

**CTIOperationResponse - setParameter(String key, Object value)**

Sets the specified key-value pair on the associated CTIOperationResponse object.

Using the get/set parameter methods enables the passing of virtually any string or number value between an operation handler and a message transformer. The parameter object can contain zero or more key/value pairs of data that directly correlate to the current operation request. The operation handler determines what key/value pairs are required as it consumes these values. Typically the operation handler sets these parameters on the CTIOperationResponse object by calling this method and the message transformer consumes them using the `CTIOperationResponse - getParameter(String key)` or `CTIOperationResponse - getParameters()` method.

ℹ Note: You should only store objects that can survive a `JSON.parse(JSON.stringify(object))` operation. Objects not meeting this criteria may not propagate correctly through the entire operation processing chain.

For example:

```javascript
var x = {
    "string": 'abc',
    "int": 123,
};
```
```javascript
"float": 1.234,
"number": new Number(3),
"boolean": true,
"date": new Date(2006, 0, 2, 15, 4, 5),
"object": {    "string": 'abc',
    "int": 123,
    "float": 1.234,
    "number": new Number(3),
    "boolean": true,
    "date": new Date(2006, 0, 2, 15, 4, 5)
},
"function": function(abc) {

}
}

var stringify = JSON.stringify(x);
gs.info(stringify);
var hydrate = JSON.parse(stringify);
var stringify2 = JSON.stringify(hydrate);
gs.info(stringify2);

Produces:

```json
{
"string": "abc",
"int": 123,
"float": 1.234,
"number": 3,
"boolean": true,
"date": "2006-01-02T23:04:05.000Z",
"object": {
"string": "abc",
"int": 123,
"float": 1.234,
"number": 3,
"boolean": true,
"date": "2006-01-02T23:04:05.000Z"
}
}
```

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the key under which to store the associated value.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>Value to store for the key. Valid data types:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• string</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• int</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• float</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• number</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>boolean</td>
<td></td>
</tr>
<tr>
<td></td>
<td>date</td>
<td></td>
</tr>
<tr>
<td></td>
<td>object</td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var response = new sn_cti_core.CTIOperationResponse();
response.setParameter('contact.phone', '+15552222345');
response.setParameter('contact.okToCall', true);
response.setParameter('contact.address', {
    street: '1234 Main St'
    city: 'API City',
    zip: 91335
});
```

**CTIOperationResponse - setSessionAttribute(String key, Object value)**

Sets the specified session attribute key-value pair on the associated CTIOperationResponse object.

Using the get/set session attribute methods enables the passing of virtually any string or number value between an operation handler and a message transformer. The sessionAttribute object can contain zero or more key/value pairs of data that are valid for the duration of a computer telephony integration provider defined session. Session attributes can also be accessed within a contact flow. The operation handler determines what session attribute key/value pairs are needed as it consumes these values. Typically the operation handler sets these attributes on the CTIOperationResponse object by calling this method and the message transformer gets the attributes using the `CTIOperationResponse - getSessionAttribute(String key)` or `CTIOperationResponse - getSessionAttributes()` method.
Note: You should only store objects that can survive a JSON.parse(JSON.stringify(object)) operation. Objects not meeting this criteria may not propagate correctly through the entire operation processing chain.

For example:

```javascript
var x = {
    "string": 'abc',
    "int": 123,
    "float": 1.234,
    "number": new Number(3),
    "boolean": true,
    "date": new Date(2006, 0, 2, 15, 4, 5),
    "object": {
        "string": 'abc',
        "int": 123,
        "float": 1.234,
        "number": new Number(3),
        "boolean": true,
        "date": new Date(2006, 0, 2, 15, 4, 5)
    },
    "function": function(abc) {
    }
};

var stringify = JSON.stringify(x);
gs.info(stringify);
var hydrate = JSON.parse(stringify);
var stringify2 = JSON.stringify(hydrate);
gs.info(stringify2);
```

Produces:

```javascript
```
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the session attribute key under which to set the associated value.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>Value to set. Valid data types:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• string</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• int</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• float</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• boolean</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• date</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• object</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var response = new sn_cti_core.CTIOperationResponse();
response.setSessionAttribute('contact.phone', '+15552222345');
response.setSessionAttribute('contact.okToCall', true);
response.setSessionAttribute('contact.address', {
  street: '1234 Main St'
  city: 'API City',
  zip: 91335
});
```

**CTIOperationResponse - setStatusCode(Number statusCode)**

Sets the status code of the current operation request on the CTIOperationResponse object.

You can use any integer values to define the status, but it is recommended that you emulate the HTTP status codes, such as 200 = Ok/Success, 400 = Bad Request, 401 = Unauthorized.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>statusCode</td>
<td>Number</td>
<td>Integer that uniquely identifies the status of the requested operation.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
(function(/*CTIOperationRequest*/ request, /*CTIOperationResponse*/ response, /*Context*/ ctx) {
    var notes = '', lang = request.getLanguage();
    try {
        var interactionGr = request.getInteractionRecord();
        var now_GR = new GlideRecord('sys_user');
        if(interactionGr && now_GR.get(interactionGr.opened_for.toString())) {
            if(now_GR.locked_out == true) {
                notes += gs.getMessageLang("User record found locked. Unlocking the account.
                ", lang);
                now_GR.locked_out=false;
                now_GR.update();
                response.setStatusCode(200);
                response.setMessage(gs.getMessageLang("Your account has been unlocked.", lang));
            } else {
                notes += gs.getMessageLang("User record not locked.
                ", lang);
                response.setStatusCode(200);
                response.setMessage(gs.getMessageLang("Your account doesn't seem to be locked.",
                
                lang));
            }
        } else {
            notes += gs.getMessageLang("User record not found. Transfer to agent
            ", lang);
            response.setStatusCode(302);
            response.setMessage(gs.getMessageLang("I am unable to find your record. Let me
            
            transfer to someone who can help.", lang));
        }
        if(interactionGr) {
            interactionGr.work_notes = notes;
        }
    }
});
```
CustomEvent - Client

You can use CustomEvent API to show qualified embedded help in the right sidebar.

See Use embedded help qualifiers for more information.

CustomEvent - fireAll(String event, String qualifier)

Show the embedded-help content specified by the qualifier parameter in the right sidebar.

Before using the fireAll() method, you must have created the Embedded Help qualifier and help content.

⚠️ Note: To write to the debug log, make a call to the global function jslog().

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
<td>String</td>
<td>The event to send. Must be the string &quot;embedded_help:load_embedded_help&quot;</td>
</tr>
<tr>
<td>qualifier</td>
<td>String</td>
<td>The qualifier name created in the Embedded Help application.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

var qualifier = 'your-EH-qualifier';

CustomEvent.fireAll("embedded_help:load_embedded_help", qualifier);
Data - Scoped, Global
A Data object contains the results of transform performed by a
\texttt{sn\_clotho.Client.transform()} method.

Do not use a constructor to create an instance of this class, instead use the
object returned by the \texttt{sn\_clotho.Client.transform()} method.

The \texttt{Data} class can be used in scoped and global server scripts. When using the
\texttt{Data} class, use the \texttt{sn\_clotho} namespace identifier.

This class is part of the MetricBase application.

**Scoped Data - \texttt{getEnd()}**
Returns the end time for data in the Data object.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>GlideDateTime</td>
</tr>
</tbody>
</table>

**Scoped Data - \texttt{getLabel()}**
Returns the label assigned by the \texttt{sn\_clotho.ClothoTransform.label()} method.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>
Scoped Data - `getMetricName()`

Returns the name of the metric of the data series. Returns null when the data object is associated with multiple data series.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the metric field. Returns null when the data object is associated with multiple data series.</td>
</tr>
</tbody>
</table>

Scoped Data - `getPeriod()`

Returns the time period in milliseconds.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The elapsed time in seconds.</td>
</tr>
</tbody>
</table>

Scoped Data - `getStart()`

Returns the start time for data in the Data object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDateTime</td>
<td>The time for the first data point.</td>
</tr>
</tbody>
</table>

Scoped Data - getSubject()

Returns the subject of the data series. Returns null when the data object is associated with multiple data series.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The subject field value of the subject GlideRecord. This is generally the sys_id of the subject GlideRecord.</td>
</tr>
</tbody>
</table>

Scoped Data - getTableName()

Returns the name of the table assigned in the DataSelector class constructor. Returns null when the data object is associated with multiple data series.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Table name. Returns null when the data object is associated with multiple data series.</td>
</tr>
</tbody>
</table>
Scoped Data - getValues()
Returns an array of values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of numbers.</td>
</tr>
</tbody>
</table>

Scoped Data - size()
Returns the number of values in the Data object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of values in the object.</td>
</tr>
</tbody>
</table>

DataBuilder - Scoped, Global
Use the DataBuilder class to create a series of data points for a metric. Use the sn_clotho.Client.put() method to save the values.

The DataBuilder class can be used in scoped and global server scripts. When using the DataBuilder class, use the sn_clotho namespace identifier.

This class is part of the MetricBase application.

Scoped DataBuilder - add(GlideDateTime start, Array value)
Add a series of data points to the DataBuilder object. Each data point is a time stamp and a value.
Uses the start parameter and the retention policy collection period to calculate the time stamp for each value in the array. The first value has the start parameter as the time stamp. This method does not save the data in the MetricBase database. Use the `sn_clotho.Client.put()` method to save the values.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>GlideDateTime</td>
<td>The time stamp for the first data point. Subsequent time stamps are calculated using the retention policy collection period.</td>
</tr>
<tr>
<td>value</td>
<td>Array</td>
<td>An array of numbers.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataBuilder</td>
<td>The same DataBuilder object.</td>
</tr>
</tbody>
</table>

```javascript
var points = [7, 0.5, 273];
var dataBuilder = new sn_clotho.DataBuilder(now_GR, 'cpu_percentage');
// this creates a GlideDateTime object set to the current date and time
var time = new GlideDateTime();
dataBuilder.add(time, points);
```

### Scoped DataBuilder - add(GlideDateTime start, Number value)

Add a data point to the DataBuilder object. Each data point is a time stamp and a value. This method does not save the data point in the metric. Use the `sn_clotho.Client.put()` method to save the values.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>GlideDateTime</td>
<td>The time stamp for the data point.</td>
</tr>
<tr>
<td>value</td>
<td>Number</td>
<td>The value of the data point.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataBuilder</td>
<td>The DataBuilder object.</td>
</tr>
</tbody>
</table>

```javascript
var dataBuilder = new sn_clotho.DataBuilder(now_GR, 'cpu_percentage');
// this creates a GlideDateTime object set to the current date and time
var time = new GlideDateTime();
dataBuilder.add(time, 0.6);
```

**Scoped DataBuilder - DataBuilder(Object glideRecord, String subject, String metric)**

Creates an instance of the DataBuilder class.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>glideRecord</td>
<td>Object</td>
<td>GlideRecord from which to obtain the domain.</td>
</tr>
<tr>
<td>subject</td>
<td>String</td>
<td>The sys_id of the GlideRecord associated with this series.</td>
</tr>
<tr>
<td>metric</td>
<td>String</td>
<td>The field name of the metric.</td>
</tr>
</tbody>
</table>

```javascript
// Where cpu_percentage is the name of the metric
var dataBuilder = new sn_clotho.DataBuilder(now_GR, 'cpu_percentage');
```

**DCManager - Global**

The DCManager API enables you to group data by type.

With the DCManager API, you can:

- Assign data classifications to existing dictionary entries.
- Look up the data classifications for specific dictionary entries.
- Remove all data classifications associated with specific dictionary entries.
- Retrieve a list of all data classifications available in the current domain.

This API requires the Data Classification [com.glide.data_classification] plugin.

For more information, see Data Classification.
DCManager - classify(String dictEntries, String dataClasses)

Assigns pre-defined or user-defined data classifications to existing dictionary entries.

Requires the admin or data_classification_admin role.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>dictEntries</td>
</tr>
<tr>
<td>dataClasses</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

This example finds records containing social security numbers and classifies the records as confidential.

```javascript
var dcm = new SNC.DCManager();
var confidentialClass = {};
var ssnFields = [];
var dataClasses = JSON.parse(dcm.getAllDataClasses());

// Get the Confidential data class record
dataClasses.forEach(function (dataClass) {
    if (dataClass.name == "Confidential")
        confidentialClass = dataClass;
});

// Find fields that seem to be strong social security numbers
var dictionaryGR = new GlideRecord("sys_dictionary");
dictionaryGR.addQuery("element", "ssn").addOrCondition("element", "social_security_number");
dictionaryGR.query();
```
while (dictionaryGR.next())
    ssnFields.push(dictionaryGR.getUniqueValue());

// Classify any found entries as Confidential
if (ssnFields.length > 0)
    dcm.classify(ssnFields.join(), confidentialClass.sys_id);

Output:
"Successfully stored the data classification configurations"

DCManager - clearClassification(String dictEntries)
Removes all data classifications for the specified dictionary entries.
Requires the admin or data_classification_admin role.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dictEntries</td>
<td>String</td>
<td>The sys_ids of the records you want to remove classifications from. The sys_ids are from the Dictionary [sys_dictionary] table. Entered as a comma-separated list enclosed in a string.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Message describing the result of the operation.</td>
</tr>
</tbody>
</table>

This example removes the data classification for a dictionary entry.

```javascript
var dcm = new SNC.DCManager();
gs.info(dcm.clearClassification("445de0a6dba30300efc57416bf9619b0"));
```

Output:
"Classifications removed for the specified dictionary entries"

DCManager - getAllDataClasses()
Returns a list of all data classifications available in the current domain.
Requires the admin, data_classification_admin, or data_classification_auditor role.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Array&gt;</td>
<td>Result of the request. Returns the sys.id and name for each available data classification. If there are no data classifications, it returns an empty array.</td>
</tr>
<tr>
<td></td>
<td>Data classifications can be organized into parent-child relationships. If there are parent data classifications, they are identified in the result.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>[</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;parent&quot;: (Object),</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>&lt;Array&gt;.parent</td>
<td>Entry for a parent data classification.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&quot;parent&quot;: (</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>)</td>
</tr>
<tr>
<td>&lt;Array&gt;.parent.sys_id</td>
<td>Sys_id of the parent data classification from the Data Classification [data_classification] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Array&gt;.parent.name</td>
<td>Name of the parent data classification.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Array&gt;.sys_id</td>
<td>Sys_id of the data classification from the Data Classification [data_classification] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Array&gt;.name</code></td>
<td>Name of the data classification.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

This example retrieves a list of all available data classifications.

```javascript
var dcm = new SNC.DCManager();
gs.info(dcm.getAllDataClasses());
```

Output:

```
[
  {
    "parent": {
      "sys_id": "a9670fc773fc1010ae8dd21eef6a735",
      "name": "Confidential"
    },
    "sys_id": "348107b951d71010f877f3f178e7dd0d",
    "name": "Personally identifiable information"
  },
  {
    "sys_id": "a9670fc773fc1010ae8dd21eef6a735",
    "name": "Confidential"
  },
  {
    "sys_id": "59b7070b73fc1010ae8dd21eef6a764",
    "name": "Restricted"
  },
  {
    "sys_id": "11d60fc773fc1010ae8dd21eef6a744",
    "name": "Internal"
  },
  {
    "sys_id": "f5b4cf4773fc1010ae8dd21eef6a766",
    "name": "Public"
  }
]
```

**DCManager - getClassification(String dictEntries)**

Retrieves all data classifications for the specified dictionary entries.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dictEntries</td>
<td>String</td>
<td>The sys_ids of the records you want to retrieve classifications for. The sys_ids are from the Dictionary [sys_dictionary] table. Entered as a comma-separated list enclosed in a string.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;</td>
<td>JSON object containing each dictionary entry's sys_id with an array of its associated data classes. If there are no associated data classifications, it returns a message describing the result of the operation. Data classifications can be organized into parent-child relationships. If there are parent data classifications, they are identified in the result.</td>
</tr>
</tbody>
</table>

Data type: Object

```
{
  <sys_dictionary_sys_id>: [
    {
      "parent": {Object},
      "sys_id": "String",
      "name": "String"
    }
  ]
}
```

<Object>.parent Entry for a parent data classification. Data type: Object

```
"parent": {
  "sys_id": "String",
  "name": "String"
}
```

<Object>.parent.sys_id_sys_id of the parent data classification from the Data Classification [data_classification] table. Data type: String
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.parent.name</td>
<td>Name of the parent data classification.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.sys_id</td>
<td>Sys_id of the data classification from the Data Classification [data_classification] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.name</td>
<td>Name of the data classification.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

This example retrieves the data classifications for a dictionary entry.

```javascript
var dcm = new SNC.DCManager();
gs.info(dcm.getClassification("445de0a6dba30300efc57416bf9619b0"));
```

**Output:**

```javascript
{
  "445de0a6dba30300efc57416bf9619b0": [
    {
      "parent": {
        "sys_id": "a9670fc773fc1010ae8dd21efaf6a735",
        "name": "Confidential"
      },
      "sys_id": "348107b951d71010f877f3f178e7dd0d",
      "name": "Personally identifiable information"
    }
  ]
}
```

**ScopedDCManager - Scoped**

The `ScopedDCManager` API enables you to group data by type.

With the `ScopedDCManager` API, you can:

- Assign data classifications to existing dictionary entries.
- Look up the data classifications for specific dictionary entries.
- Remove all data classifications associated with specific dictionary entries.
- Retrieve a list of all data classifications available in the current domain.
This API requires the Data Classification [com.glide.data_classification] plugin. For more information, see Data Classification.

ScopedDCManager - classify(String dictEntries, String dataClasses)
Assigns pre-defined or user-defined data classifications to existing dictionary entries.

Requires the admin or data_classification_admin role.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dictEntries</td>
<td>String</td>
<td>The sys_ids of the records you want to classify. The sys_ids are from the Dictionary [sys_dictionary] table. Entered as a comma-separated list enclosed in a string.</td>
</tr>
<tr>
<td>dataClasses</td>
<td>String</td>
<td>The sys_ids of the data classifications you want to assign. The sys_ids are from the Data Classification [data_classification] table. Entered as a comma-separated list enclosed in a string.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Message describing the result of the operation.</td>
</tr>
</tbody>
</table>

This example finds records containing social security numbers and classifies the records as confidential.

```javascript
var dcm = new global.ScopedDCManager();
var confidentialClass = {};
var ssnFields = [];
var dataClasses = JSON.parse(dcm.getAllDataClasses());

// Get the Confidential data class record
dataClasses.forEach(function (dataClass) {
  if (dataClass.name == "Confidential")
    confidentialClass = dataClass;
});

// Find the fields that seem to be strong social security numbers
var dictionaryGR = new GlideRecord("sys_dictionary");
```
dictionaryGR.addQuery("element", "ssn").addOrCondition("element", "social_security_number");
dictionaryGR.query();
while (dictionaryGR.next())
    ssnFields.push(dictionaryGR.getUniqueValue());

// Classify any found entries as confidential
if (ssnFields.length > 0)
    dcm.classify(ssnFields.join(), confidentialClass.sys_id);

Output:
"Successfully stored the data classification configurations"

ScopedDCManager - clearClassification(String dictEntries)
Removes all data classifications for the specified dictionary entries.
Requires the admin or data_classification_admin role.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dictEntries</td>
<td>String</td>
<td>The sys_ids of the records you want to remove classifications from. The sys_ids are from the Dictionary [sys_dictionary] table. Entered as a comma-separated list enclosed in a string.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Message describing the result of the operation.</td>
</tr>
</tbody>
</table>

This example removes the data classification for a dictionary entry.

```javascript
var dcm = new global.ScopedDCManager();
gs.info(dcm.clearClassification("445de0a6dba30300efc57416bf9619b0"));
```

Output:
"Classifications removed for the specified dictionary entries"

ScopedDCManager - getAllDataClasses()

Returns a list of all data classifications available in the current domain.
Requires the admin, data_classification_admin, or data_classification_auditor role.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Array&gt;</td>
<td>Result of the request. Returns the sys_id and name for each available data classification. If there are no data classifications, it returns an empty array. Data classifications can be organized into parent-child relationships. If there are parent data classifications, they are identified in the result. Data type: Array</td>
</tr>
<tr>
<td>&lt;Array&gt;.parent</td>
<td>Entry for a parent data classification.</td>
</tr>
<tr>
<td>&lt;Array&gt;.parent.sys_id</td>
<td>Sys_id of the parent data classification from the Data Classification [data_classification] table.</td>
</tr>
<tr>
<td>&lt;Array&gt;.parent.name</td>
<td>Name of the parent data classification.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Array&gt;.sys_id</td>
<td>Sys_id of the data classification from the Data Classification [data_classification] table. Data type: String</td>
</tr>
<tr>
<td>&lt;Array&gt;.name</td>
<td>Name of the data classification. Data type: String</td>
</tr>
</tbody>
</table>

This example retrieves a list of all available data classifications.

```javascript
var dcm = new global.ScopedDCManager();
gs.info(dcm.getAllDataClasses());
```

Output:

```json
[
{
"parent": {
  "sys_id": "a9670fc773fc1010ae8dd21efaf6a735",
  "name": "Confidential"
},
  "sys_id": "348107b951d71010f877f3f178e7dd0d",
  "name": "Personally identifiable information"
},
{
  "sys_id": "a9670fc773fc1010ae8dd21efaf6a735",
  "name": "Confidential"
},
{
  "sys_id": "59b7070b73fc1010ae8dd21efaf6a764",
  "name": "Restricted"
},
{
  "sys_id": "11d60fc773fc1010ae8dd21efaf6a744",
  "name": "Internal"
},
{
  "sys_id": "f5b4cf4773fc1010ae8dd21efaf6a766",
  "name": "Public"
}
]
ScopedDCManager - getClassification(String dictEntries)
Retrieves all data classifications for the specified dictionary entries.
Requires the admin, data_classification_admin, or data_classification_auditor role.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dictEntries</td>
<td>String</td>
<td>The sys_ids of the records you want to retrieve classifications for. The sys_ids are from the Dictionary [sys_dictionary] table. Entered as a comma-separated list enclosed in a string.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;</td>
<td>JSON object containing each dictionary entry’s sys_id with an array of its associated data classes. If there are no associated data classifications, it returns a message describing the result of the operation. Data classifications can be organized into parent-child relationships. If there are parent data classifications, they are identified in the result. Data type: Object</td>
</tr>
<tr>
<td>&lt;Object&gt;.parent</td>
<td>Entry for a parent data classification. Data type: Object</td>
</tr>
</tbody>
</table>

```json
{
  "sys_dictionary_sys_id": [ 
    {
      "parent": {Object},
      "sys_id": "String",
      "name": "String"
    }
  ]
}
```

```json
"parent": {
  "sys_id": "String",
```
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
</tbody>
</table>

<Object>.parent.sys_id of the parent data classification from the Data Classification [data_classification] table.
Data type: String

<Object>.parent.name name of the parent data classification.
Data type: String

<Object>.sys_id Sys_id of the data classification from the Data Classification [data_classification] table.
Data type: String

<Object>.name Name of the data classification.
Data type: String

This example retrieves the data classifications for a dictionary entry.

```javascript
var dcm = new global.ScopedDCManager();
gs.info(dcm.getClassification("445de0a6dba30300efc57416bf9619b0"));
```

Output:

```json
{
    "445de0a6dba30300efc57416bf9619b0": [
    {
        "parent": {
            "sys_id": "a9670fc773fc1010ae8dd21e6af6a735",
            "name": "Confidential"
        },
        "sys_id": "348107b951d71010f877f3f178e7dd0d",
        "name": "Personally identifiable information"
    }
    ]
}
```

**DatasetDefinition - Global**

Identifies a set of records including a table name, columns, and row selection criteria to use as input for ML training algorithms. Datasets do not contain the actual data.
The DatasetDefinition API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the sn_ml namespace. For information, see Predictive Intelligence.

Use the dataset to estimate mutual information PredictabilityEstimate or train data specified by an Encoder. You can also use the dataset to train data specified by one of the following solution types:

- ClassificationSolution
- ClusteringSolution
- RegressionSolution
- SimilaritySolution

For usage guidelines, refer to Using ML APIs.

**DatasetDefinition - DatasetDefinition(Object)**

Creates an instance of the DatasetDefinition class, enabling you to define a dataset by table name, fields, and query.

Create your dataset definition by passing a table and a list of fields. You can also pass a query to restrict datasets to include rows with specific characteristics. Once created, a DatasetDefinition object cannot be modified.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>Object</td>
<td>JavaScript object containing the dataset definition properties.</td>
</tr>
<tr>
<td>config.tableName</td>
<td>String</td>
<td>Name of the table for the dataset. For example, &quot;tableName&quot; : &quot;Incident&quot;.</td>
</tr>
<tr>
<td>config.fieldNames</td>
<td>Array</td>
<td>Optional. List of field names from the specified table as strings. For example, &quot;fieldNames&quot; : [&quot;short_description&quot;, &quot;priority&quot;]. Default: All fields</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>config.fieldDetails</td>
<td>Array</td>
<td>Optional. List of JavaScript objects that specify field properties. Use this property to force machine learning algorithms to interpret fields as being a specific type. You do not need to get field details for every field listed in the <code>fieldNames</code> property. All details must correspond with a field listed in the <code>fieldNames</code> array.</td>
</tr>
<tr>
<td>config.fieldDetails.name</td>
<td>String</td>
<td>Name of the field defining the type of information to restrict this dataset to. If used, this field name must match the corresponding name listed in the <code>fieldNames</code> property.</td>
</tr>
<tr>
<td>config.fieldDetails.type</td>
<td>String</td>
<td>Machine-learning field type. Specifying the data type forces the ML trainer to interpret a field as having that type. If no data type is specified, the system determines the type. Supported types:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>nominal</strong>: ML interprets this field as containing classes or categories.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>numeric</strong>: ML interprets this field as containing numbers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>text</strong>: ML interprets this field as containing text.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>These types identify data types from a machine learning perspective. The ML type might differ from the type listed in the source table. A field can be a string type, but its purpose can be to encode a nominal value. For example, t-shirt sizes such as “XL”, “L”, or…</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config.encodedQuery</td>
<td>String</td>
<td>Optional. Encoded query string in standard Glide format. See Encoded query strings. You can construct the query to be absolute or relative. For example, your query can return rows for the previous 3 months (relative), or for the May through July period (absolute). Whether using an absolute or relative pattern, the data a definition identifies can change if the rows in the underlying table change.</td>
</tr>
</tbody>
</table>

The following example shows how to create a dataset definition.

```javascript
var myData = new sn_ml.DatasetDefinition(
  {
    'tableName' : 'incident',
    'fieldNames' : ['category', 'short_description', 'priority', 'assignment_group.name'],
    'fieldDetails' : [
      {
        'name' : 'category',
        'type' : 'nominal'
      },
      {
        'name' : 'short_description',
        'type' : 'text'
      }
    ],
    'encodedQuery' : 'sys_created_onONLast%20%20quarters@javascript:gs.beginningOfLast2Quarters()@javascript:gs.endOfLast2Quarters()^state=3'
  });
```

**DatasetDefinition - getEligibleFields(String capability)**

Returns a list of fields that are eligible as either input fields (features) or predicted fields regarding a solution of a given capability, for example, a classification solution. Eligibility is determined based on the fields having the appropriate glide data types.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>capability</td>
<td>String</td>
<td>Capability for which to retrieve fields eligible for training. This method currently only supports classification solutions, any other value for the capability throws a &quot;capability not supported&quot; exception. Valid values: &quot;classification&quot;</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing eligible input field names and eligible output field names.</td>
</tr>
<tr>
<td></td>
<td>{ &quot;eligibleInputFieldNames&quot; : [Array], &quot;eligibleOutputFieldNames&quot; : [Array] }</td>
</tr>
<tr>
<td>&lt;Object&gt;.eligibleInputFieldNames</td>
<td>List of strings indicating input fields eligible for training. Data type: Array</td>
</tr>
<tr>
<td>&lt;Object&gt;.eligibleOutputFieldNames</td>
<td>List of strings indicating output fields eligible for training. Data type: Array</td>
</tr>
</tbody>
</table>

The following example shows how to display eligible fields for a classification solution.

```javascript
var myIncidentData = new sn_ml.DatasetDefinition({
  'tableName' : 'incident',
  'encodedQuery' : 'activeANYTHING'
});

var eligibleFields = JSON.parse(myIncidentData.getEligibleFields('classification'));

gs.print(JSON.stringify(eligibleFields, null, 2));
```

Output:
.DateTimeUtils - Global

DateTimeUtils class is a collection of date/time functions.
The DateTimeUtils class is provided via the Script Include DateTimeUtils.
The DateTimeUtils class can be used in any server-side script, and is available through GlideAjax.

DateTimeUtils - int8ToGlideDateTime(Number int64)

Converts Microsoft AD integer8 DateTime format into GlideDateTime format.

Integer8 is also known as Microsoft Filetime format. This method is commonly used when importing AD user's date fields, such as Expiration Date.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int64</td>
<td>Number</td>
<td>A 64-bit value representing the number of 100-nanosecond intervals since January 1, 1601 (UTC).</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDateTime</td>
<td>A GlideDateTime object set to the Integer8 date and time.</td>
</tr>
</tbody>
</table>
//convert and set account expiration date from AD
//this is an example that could be used in an LDAP import transform map to import the LDAP account
//expires attribute to a customer created u_account_expires GlideDateTime field
var dtUtil = new DateTimeUtils();
target.u_account_expires = dtUtil.int8ToGlideDateTime(source.u_accountexpires);

DateTimeUtils - msToGlideDateTime(Number milliseconds)
Convert milliseconds to a GlideDateTime object

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>milliseconds</td>
<td>Number</td>
<td>The number of milliseconds</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDateTime</td>
<td>A GlideDateTime object</td>
</tr>
</tbody>
</table>

//example script to call the method from a client
Replace MILLISECONDSVALUE with your variable
var ga = new GlideAjax('DateTimeUtils');
ga.addParam('sysparm_name','msToGlideDateTime');
ga.addParam('sysparm_value', MILLISECONDSVALUE);
ga/XMLWait();
var newGDT = ga.getAnswer();

DecisionTableAPI - Scoped, Global
Use the DecisionTableAPI class to access Decision Tables data in scoped and global server-side scripts.
Access DecisionTableAPI methods using the sn_dt namespace. To learn more about Decision Tables, see Decision Tables.

DecisionTableAPI - DecisionTableAPI()
Instantiates a DecisionTableAPI object.
var dt = new sn_dt.DecisionTableAPI();

**DecisionTableAPI - getAll()**

Returns all decision tables from the Decision Tables [sys_decision] table.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Object  | All decision tables and their fields from the Decision Tables [sys_decision] table. Each decision table contains these key-value pairs:  
  - distinctAnswers: Array. Answer records associated with the decision table. Each record contains these key-value pairs:  
    - valid: Boolean. True if the record exists in the system; otherwise false.
    - label: String. Answer record label.
    - value: String. Sys ID of the answer record.
    - table: String. Table containing the answer record.
  - referenceQualifier: String. Used for internal purposes.
  - inputs: Array. Decision inputs from the Decision Inputs [sys_decision_input] table associated with the decision table. Each record contains these key-value pairs:  
    - searchField: String. Used for internal purposes.
    - defaultValue: String. Default value for the input.
    - show_ref_finder: Boolean. Used for internal purposes.
    - use_dependent: Boolean. Used for internal purposes. |
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>◦</td>
<td>type: String. Input data type.</td>
</tr>
<tr>
<td>◦</td>
<td>mandatory: Boolean. True if the input is mandatory; otherwise false.</td>
</tr>
<tr>
<td>◦</td>
<td>extended: Boolean. True if the input extends another field; otherwise false.</td>
</tr>
<tr>
<td>◦</td>
<td>local: Boolean. Used for internal purposes.</td>
</tr>
<tr>
<td>◦</td>
<td>sys_class_name: String. Used for internal purposes.</td>
</tr>
<tr>
<td>◦</td>
<td>reference: String. Reference table used if the input type is reference.</td>
</tr>
<tr>
<td>◦</td>
<td>dependent_on: String. Field that document ID and choice inputs depend on.</td>
</tr>
<tr>
<td>◦</td>
<td>data_structure: String. Used for internal purposes.</td>
</tr>
<tr>
<td>◦</td>
<td>readonly: Boolean. True if the input is read-only; otherwise false.</td>
</tr>
<tr>
<td>◦</td>
<td>id: String. Sys ID of the input from the Decision Inputs [sys_decision_input] table.</td>
</tr>
<tr>
<td>◦</td>
<td>type_label: String. Input data type label.</td>
</tr>
<tr>
<td>◦</td>
<td>table: String. Reference field table if the input type is reference.</td>
</tr>
<tr>
<td>◦</td>
<td>order: Number. Order in which the system evaluates inputs.</td>
</tr>
<tr>
<td>◦</td>
<td>ref_qual: String. Condition applied to the reference table.</td>
</tr>
<tr>
<td>◦</td>
<td>reference_display: String. Used for internal purposes.</td>
</tr>
<tr>
<td>◦</td>
<td>choiceOption: String. Method for users to see a list of suggested values.</td>
</tr>
<tr>
<td>◦</td>
<td>label: String. Label of the input record.</td>
</tr>
<tr>
<td>◦</td>
<td>hint: String. Hint text used to help users understand the input required.</td>
</tr>
<tr>
<td>◦</td>
<td>name: String. Internal name of the input record.</td>
</tr>
<tr>
<td>◦</td>
<td>attributes: Array. Field attributes that apply to the input record.</td>
</tr>
<tr>
<td>◦</td>
<td>maxsize: Number. Maximum number of characters allowed in the field.</td>
</tr>
<tr>
<td>◦</td>
<td>columnName: String. Name of the input.</td>
</tr>
</tbody>
</table>

• domain: String. Domain in which the decision table is visible.
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• answerType: String. Used for internal purposes.</td>
<td></td>
</tr>
<tr>
<td>• questions: Array. Decision records from the Decision [sys_decision_question] table associated with the decision table. Decision records contain these key-value pairs.</td>
<td></td>
</tr>
<tr>
<td>◦ defaultAnswer: Boolean. True if this decision is the default for the decision table; otherwise false.</td>
<td></td>
</tr>
<tr>
<td>◦ condition: String. Conditions required to reach this decision. The fields available for the condition are the decision inputs that are associated with this decision table.</td>
<td></td>
</tr>
<tr>
<td>◦ answer: Object. Answer record associated with the decision. Answer objects contain these key-value pairs.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ valid: Boolean. True if the record exists in the system; otherwise false.</td>
</tr>
<tr>
<td></td>
<td>◦ label: String. Answer record label.</td>
</tr>
<tr>
<td></td>
<td>◦ value: String. Sys ID of the answer record.</td>
</tr>
<tr>
<td></td>
<td>◦ table: String. Table containing the answer record.</td>
</tr>
<tr>
<td></td>
<td>◦ decisionTable: String. Sys ID of the decision table from the Decision Tables [sys_decision] table associated with the decision.</td>
</tr>
<tr>
<td></td>
<td>◦ domain: String. Domain in which the decision is visible.</td>
</tr>
<tr>
<td></td>
<td>◦ active: Boolean. True if the decision record is active; otherwise false.</td>
</tr>
<tr>
<td></td>
<td>◦ id: String. Sys ID of the decision record from the Decision [sys_decision_question] table.</td>
</tr>
<tr>
<td></td>
<td>◦ label: String. Label for the decision record.</td>
</tr>
<tr>
<td></td>
<td>◦ order: Number. Order in which the system evaluates decisions.</td>
</tr>
<tr>
<td>• accessibleFrom: String. Scopes that have access to the record.</td>
<td></td>
</tr>
<tr>
<td>• name: String. Internal name of the decision table record.</td>
<td></td>
</tr>
<tr>
<td>• active: Boolean. True if the decision table record is active; otherwise false.</td>
<td></td>
</tr>
<tr>
<td>• id: String. Sys ID of the decision table record.</td>
<td></td>
</tr>
<tr>
<td>• label: String. Label of the decision table record.</td>
<td></td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>answerTable: Object</td>
<td>Answer table associated with the decision table.</td>
</tr>
<tr>
<td>displayValue: String</td>
<td>Display name of the table that contains answer records.</td>
</tr>
<tr>
<td>value: String</td>
<td>Internal name of the table that contains answer records.</td>
</tr>
</tbody>
</table>

Format: JSON

```javascript
var dt = new sn_dt.DecisionTableAPI();
var response = dt.getAll();
gs.info(JSON.stringify(response));
```

Output:

```json
{"result": [
{
"distinctAnswers": [{
"valid": true,
"label": "Assignment Group Manager Approval",
"value": "5684821db701230034d1da23ee11a94d",
"table": "chg_approval_def"
}],
"referenceQualifier": ",",
"inputs": [
{
"searchField": null,
"defaultValue": "0",
"show_ref_finder": false,
"use_dependent": false,
"type": "integer",
"mandatory": false,
"extended": false,
"local": false,
"sys_class_name": ",",
"reference": ",",
"dependent_on": ",",
"data_structure": ",",
"readonly": false,
"id": "eca52ca7b701230034d1da23ee11a95a",
}]
}]
```
"type_label": "Integer",
"table": "",
"order": 100,
"ref_qual": "",
"reference_display": "",
"choiceOption": "",
"label": "CI Count",
"hint": "",
"name": "ci_count",
"attributes": {"element_mapping_provider": "com.glide.decision_table.data.DecisionTableVariableMapper"},
"maxsize": 40,
"columnName": ""
},
{
"searchField": "number",
"defaultValue": "",
"show_ref_finder": false,
"use_dependent": false,
"type": "reference",
"mandatory": false,
"extended": false,
"local": false,
"sys_class_name": "",
"reference": "change_request",
"dependent_on": "",
"data_structure": "",
"readonly": false,
"id": "1395e8a7b701230034d1da23ee11a9b6",
"type_label": "Reference",
"table": "",
"order": 100,
"ref_qual": "",
"reference_display": "Change Request",
"choiceOption": "",
"label": "Change request",
"hint": "",
"name": "change_request",
"attributes": {"element_mapping_provider": "com.glide.decision_table.data.DecisionTableVariableMapper"},
"maxsize": 32,
"columnName": ""
}
"domain": "global",
"answerType": "reference",
"questions": [
  {
    "defaultAnswer": false,
    "condition": "ci_count>=1000^EQ",
    "answer": {
      "valid": true,
      "label": "Assignment Group Manager Approval",
      "value": "5684821db701230034d1da23ee11a94d",
      "table": "chg_approval_def"
    },
    "decisionTable": "e49568a7b701230034d1da23ee11a913",
    "domain": "global",
    "active": true,
    "state": null,
    "id": "2a36eca7b701230034d1da23ee11a961",
    "label": "Mandatory Approval if 1000 CIs or more",
    "order": 0
  }
],
"accessibleFrom": "public",
"name": "Example Standard Change Policy",
"active": true,
"id": "e49568a7b701230034d1da23ee11a913",
"label": "",
"answerTable": {
  "displayValue": "Change Approval Definition",
  "value": "chg_approval_def"
}
]

"distinctAnswers": [
  {
    "valid": true,
    "label": "CAB Approval",
    "value": "45358a5db701230034d1da23ee11a938",
    "table": "chg_approval_def"
  },
  {
    "valid": true,
    "label": "Assignment Group Manager Approval",
    "value": "5684821db701230034d1da23ee11a94d",
    "table": "chg_approval_def"
  }
]
"valid": true,
"label": "Assignment Group Approval",
"value": "dc95ca5db701230034d1da23ee11a9e9",
"table": "chg_approval_def"
}
",
"referenceQualifier": "",
"inputs": [
{
   "searchField": null,
   "defaultValue": "false",
   "show_ref_finder": false,
   "use_dependent": false,
   "type": "boolean",
   "mandatory": false,
   "extended": false,
   "local": false,
   "sys_class_name": "",
   "reference": "",
   "dependent_on": "",
   "data_structure": "",
   "readonly": false,
   "id": "c0a7869db701230034d1da23ee11a9e7",
   "type_label": "True/False",
   "table": "",
   "order": 100,
   "ref_qual": "",
   "reference_display": "",
   "choiceOption": "",
   "label": "Manager approved",
   "hint": "",
   "name": "manager_approved",
   "attributes": {"element_mapping_provider": "com.glide.decision_table.data.DecisionTableVariableMapper"},
   "maxsize": 40,
   "columnName":"
}
]
"extended": false,
"local": false,
"sys_class_name": "",
"reference": "change_request",
"dependent_on": "",
"data_structure": "",
"readonly": false,
"id": "23164e5db701230034d1da23ee11a9c5",
"type_label": "Reference",
"table": "",
"order": 100,
"ref_qual": "",
"reference_display": "Change Request",
"choiceOption": "",
"label": "Change request",
"hint": "",
"name": "change_request",
"attributes": {"element_mapping_provider": "com.glide.decision_table.data.DecisionTableVariableMapper"},
"maxsize": 32,
"columnName": ""
}]
"domain": "global",
"answerType": "reference",
"questions": [
{
"defaultAnswer": false,
"condition": "change_request.state=-3^change_request.riskIN2,3^NQchange_request.state=-3^change_request.assignment_group.managerISEMPTY"EQ",
"answer": {
"valid": true,
"label": "CAB Approval",
"value": "45358a5db701230034d1da23ee11a938",
"table": "chg_approval_def"
},
"decisionTable": "5b06ce5db701230034d1da23ee11a965",
"domain": "global",
"active": true,
"state": null,
"id": "10284a9db701230034d1da23ee11a9c1",
"label": "Requires CAB Approval",
"order": 0


{  
  "defaultAnswer": false,
  "condition": "change_request.state=-3^change_request.risk=4^manager_approved=false^EQ",
  "answer": {  
    "valid": true,
    "label": "Assignment Group Manager Approval",
    "value": "5684821db701230034dida23ee11a94d",
    "table": "chg_approval_def"
  },
  "decisionTable": "5b06ce5db701230034dida23ee11a965",
  "domain": "global",
  "active": true,
  "state": null,
  "id": "88e6829db701230034dida23ee11a9a9",
  "label": "Low Risk Manager approval",
  "order": 0
},

{  
  "defaultAnswer": false,
  "condition": "change_request.state=-4^EQ",
  "answer": {  
    "valid": true,
    "label": "Assignment Group Approval",
    "value": "dc95ca5db701230034dida23ee11a9e9",
    "table": "chg_approval_def"
  },
  "decisionTable": "5b06ce5db701230034dida23ee11a965",
  "domain": "global",
  "active": true,
  "state": null,
  "id": "a436029db701230034dida23ee11a989",
  "label": "Assess Technical Approvals",
  "order": 0
}

"accessibleFrom": "public",
"name": "Normal Change Policy",
"active": true,
"id": "5b06ce5db701230034dida23ee11a965",
"label": "",
"answerTable": {  
  "displayValue": "Change Approval Definition",

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
DecisionTableAPI - getAnswers(String decisionID)

Returns the answers associated with the specified decision table. An answer is a record on any table associated with a Decision [sys_decision_question] record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>decisionID</td>
<td>String</td>
<td>Sys ID of the Decision Table record from the Decision Tables [sys_decision] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Answer records associated with the decision table. Each record contains these key-value pairs:</td>
</tr>
<tr>
<td></td>
<td>• valid: Boolean. True if the record exists in the system; otherwise false.</td>
</tr>
<tr>
<td></td>
<td>• label: String. Answer record label.</td>
</tr>
<tr>
<td></td>
<td>• value: String. Sys ID of the answer record.</td>
</tr>
<tr>
<td></td>
<td>• table: String. Table containing the answer record.</td>
</tr>
</tbody>
</table>

Format: JSON

```javascript
var dt = new sn_dt.DecisionTableAPI();
var response = dt.getAnswers("3c4464a7b701230034d1da23ee11a993");
gs.info(JSON.stringify(response));
```

Output:

```json
"result": [{
  "valid": true,
  "label": "CAB Approval",
  "value": "45358a5db701230034d1da23ee11a938",
  "table": "chg_approval_def"
}]
```
**DecisionTableAPI - getDecision(String decisionID, Map inputs)**

Evaluates a decision table based on the provided inputs and returns an answer. If no inputs are provided, returns the first default answer found.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>decisionID</td>
<td>String</td>
<td>Sys ID of the Decision Table record from the Decision Tables [sys_decision] table.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Optional. Input values for the Decision Inputs [sys_decision_input] table associated with the provided decision table. Use the value of the Column name field as the key. The data type of the value must match the Decision Input Type field. If no inputs are provided, returns the first default answer found.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The correctly evaluated Answer record associated with the decision table. An answer record is a record from the table defined in the Decision [sys_decision_question] record Answer table field. If no inputs are provided, returns the first default answer found. If no default or correctly evaluated answers are found, returns an error.</td>
</tr>
</tbody>
</table>

```javascript
var dt = new sn_dt.DecisionTableAPI();
var inputs = new Object();
inputs['age'] = 25;
var response = dt.getDecision('26eea7a9dba07300efc65404ce961961', inputs);
```

**DecisionTableAPI - getDecisions(String decisionID, Map inputs)**

Evaluates a decision table based on the provided inputs and returns all correctly evaluated answers. If no inputs are provided, returns all default answers.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>decisionID</td>
<td>String</td>
<td>Sys ID of the Decision Table record from the Decision Tables [sys_decision] table.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Optional. Input values for the Decision Inputs [sys_decision_input] table associated with the provided decision table. If no inputs are provided, returns all default answers.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Answer records in GlideRecord format. An answer record is a record on any table associated with a Decision [sys_decision_question] record. If no inputs are provided, returns all default answers. If no default or correctly evaluated answers are found, returns an error.</td>
</tr>
</tbody>
</table>

```javascript
var dt = new sn_dt.DecisionTableAPI();
var input = new Object();
input['age'] = 25;
var response = dt.getDecisions('5b06ce5db701230034d1da23ee11a965', input);
```

**DecisionTableAPI - getDecisionTable(String decisionID)**

Returns a single decision table from the Decision Tables [sys_decision] table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>decisionID</td>
<td>String</td>
<td>Sys ID of the Decision Table record from the Decision Tables [sys_decision] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Decision table from the Decision Tables [sys_decision] table. Each decision table contains these key-value pairs.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• distinctAnswers: Array. Answer records associated with the decision table. Each record contains these key-value pairs.</td>
<td></td>
</tr>
<tr>
<td>◦ valid: Boolean. True if the record exists in the system; otherwise false.</td>
<td></td>
</tr>
<tr>
<td>◦ label: String. Answer record label.</td>
<td></td>
</tr>
<tr>
<td>◦ value: String. Sys ID of the answer record.</td>
<td></td>
</tr>
<tr>
<td>◦ table: String. Table containing the answer record.</td>
<td></td>
</tr>
<tr>
<td>• referenceQualifier: String. Used for internal purposes.</td>
<td></td>
</tr>
<tr>
<td>• inputs: Array. Decision inputs from the Decision Inputs [sys_decision_input] table associated with the decision table. Each record contains these key-value pairs.</td>
<td></td>
</tr>
<tr>
<td>◦ searchField: String. Used for internal purposes.</td>
<td></td>
</tr>
<tr>
<td>◦ defaultValue: String. Default value for the input.</td>
<td></td>
</tr>
<tr>
<td>◦ show_ref_finder: Boolean. Used for internal purposes.</td>
<td></td>
</tr>
<tr>
<td>◦ use_dependent: Boolean. Used for internal purposes.</td>
<td></td>
</tr>
<tr>
<td>◦ type: String. Input data type.</td>
<td></td>
</tr>
<tr>
<td>◦ mandatory: Boolean. True if the input is mandatory; otherwise false.</td>
<td></td>
</tr>
<tr>
<td>◦ extended: Boolean. True if the input extends another field; otherwise false.</td>
<td></td>
</tr>
<tr>
<td>◦ local: Boolean. Used for internal purposes.</td>
<td></td>
</tr>
<tr>
<td>◦ sys_class_name: String. Used for internal purposes.</td>
<td></td>
</tr>
<tr>
<td>◦ reference: String. Reference table used if the input type is reference.</td>
<td></td>
</tr>
<tr>
<td>◦ dependent_on: String. Field that document ID and choice inputs depend on.</td>
<td></td>
</tr>
<tr>
<td>◦ data_structure: String. Used for internal purposes.</td>
<td></td>
</tr>
<tr>
<td>◦ readonly: Boolean. True if the input is read-only; otherwise false.</td>
<td></td>
</tr>
<tr>
<td>◦ id: String. Sys ID of the input from the Decision Inputs [sys_decision_input] table.</td>
<td></td>
</tr>
</tbody>
</table>
| ◦ type_label: String. Input data type label. }
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>◦ table: String.</td>
<td>Reference field table if the input type is reference.</td>
</tr>
<tr>
<td>◦ order: Number.</td>
<td>Order in which the system evaluates inputs.</td>
</tr>
<tr>
<td>◦ ref_qual: String.</td>
<td>Condition applied to the reference table.</td>
</tr>
<tr>
<td>◦ reference_display: String.</td>
<td>Used for internal purposes.</td>
</tr>
<tr>
<td>◦ choiceOption: String.</td>
<td>Method for users to see a list of suggested values.</td>
</tr>
<tr>
<td>◦ label: String.</td>
<td>Label of the input record.</td>
</tr>
<tr>
<td>◦ hint: String.</td>
<td>Hint text used to help users understand the input required.</td>
</tr>
<tr>
<td>◦ name: String.</td>
<td>Internal name of the input record.</td>
</tr>
<tr>
<td>◦ attributes: Array.</td>
<td>Field attributes that apply to the input record.</td>
</tr>
<tr>
<td>◦ maxsize: Number.</td>
<td>Maximum number of characters allowed in the field.</td>
</tr>
<tr>
<td>◦ columnName: String.</td>
<td>Name of the input.</td>
</tr>
<tr>
<td>◦ domain: String.</td>
<td>Domain in which the decision table is visible.</td>
</tr>
<tr>
<td>◦ answerType: String.</td>
<td>Used for internal purposes.</td>
</tr>
<tr>
<td>◦ questions: Array.</td>
<td>Decision records from the Decision [sys_decision_question] table associated with the decision table. Decision records contain these key-value pairs.</td>
</tr>
<tr>
<td>◦ defaultAnswer: Boolean.</td>
<td>True if this decision is the default for the decision table; otherwise false.</td>
</tr>
<tr>
<td>◦ condition: String.</td>
<td>Conditions required to reach this decision. The fields available for the condition are the decision inputs that are associated with this decision table.</td>
</tr>
<tr>
<td>◦ answer: Object.</td>
<td>Answer record associated with the decision. Answer objects contain these key-value pairs.</td>
</tr>
<tr>
<td>■ valid: Boolean.</td>
<td>True if the record exists in the system; otherwise false.</td>
</tr>
<tr>
<td>■ label: String.</td>
<td>Answer record label.</td>
</tr>
<tr>
<td>■ value: String.</td>
<td>Sys ID of the answer record.</td>
</tr>
<tr>
<td>■ table: String.</td>
<td>Table containing the answer record.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>decisionTable: String.</td>
<td>Sys ID of the decision table from the Decision Tables [sys_decision] table associated with the decision.</td>
</tr>
<tr>
<td>domain: String.</td>
<td>Domain in which the decision is visible.</td>
</tr>
<tr>
<td>active: Boolean.</td>
<td>True if the decision record is active; otherwise false.</td>
</tr>
<tr>
<td>id: String.</td>
<td>Sys ID of the decision record from the Decision [sys_decision_question] table.</td>
</tr>
<tr>
<td>label: String.</td>
<td>Label for the decision record.</td>
</tr>
<tr>
<td>order: Number.</td>
<td>Order in which the system evaluates decisions.</td>
</tr>
<tr>
<td>accessibleFrom: String.</td>
<td>Scopes that have access to the record.</td>
</tr>
<tr>
<td>name: String.</td>
<td>Internal name of the decision table record.</td>
</tr>
<tr>
<td>active: Boolean.</td>
<td>True if the decision table record is active; otherwise false.</td>
</tr>
<tr>
<td>id: String.</td>
<td>Sys ID of the decision table record.</td>
</tr>
<tr>
<td>label: String.</td>
<td>Label of the decision table record.</td>
</tr>
<tr>
<td>answerTable: Object.</td>
<td>Answer table associated with the decision table.</td>
</tr>
<tr>
<td>displayValue: String.</td>
<td>Display name of the table that contains answer records.</td>
</tr>
<tr>
<td>value: String.</td>
<td>Internal name of the table that contains answer records.</td>
</tr>
</tbody>
</table>

Format: JSON

```javascript
var dt = new sn_dt.DecisionTableAPI();
var response = dt.getDecisionTable("3c4464a7b701230034d1da23ee11a993");
gs.info(JSON.stringify(response));
```

Output:

```json
"distinctAnswers": [{
  "valid": true,
  "label": "CAB Approval",
  "value": "45358a5db701230034d1da23ee11a938",
  "table": "chg_approval_def"
}
```
"referenceQualifier": "",
"inputs": [{
  "searchField": "number",
  "defaultValue": "",
  "show_ref_finder": false,
  "use_dependent": false,
  "type": "reference",
  "mandatory": false,
  "extended": false,
  "local": false,
  "sys_class_name": "",
  "reference": "change_request",
  "dependent_on": "",
  "data_structure": "",
  "readonly": false,
  "id": "634420a7b701230034d1da23ee11a94f",
  "type_label": "Reference",
  "table": "",
  "order": 100,
  "ref_qual": "",
  "reference_display": "Change Request",
  "choiceOption": "",
  "label": "Change request",
  "hint": "",
  "name": "change_request",
  "attributes": {
    "element_mapping_provider": "com.glide.decision_table.data.DecisionTableVariableMapper"
  },
  "maxsize": 32,
  "columnName": ""
},
"domain": "global",
"answerType": "reference",
"questions": [{
  "defaultAnswer": true,
  "condition": "change_request.state=-3^EQ",
  "answer": {
    "valid": true,
    "label": "CAB Approval",
    "value": "45358a5db701230034d1da23ee11a938",
    "table": "chg_approval_def"
  },
  "decisionTable": "3c4464a7b701230034d1da23ee11a993",
  "domain": "global"}]}
"active": true,
"state": null,
"id": "2764a4a7b0123034d1da23ee11a989",
"label": "Requires CAB Approval",
"order": 0
}]
"accessibleFrom": "public",
"name": "Emergency Change Policy",
"active": true,
"id": "3c4464a7b0123034d1da23ee11a993",
"label": "",
"answerTable": {
  "displayValue": "Change Approval Definition",
  "value": "chg_approval_def"
}

DecisionTableAPI - getInputs(String decisionID)

Returns the decision inputs from the Decision Inputs [sys_decision_input] table associated with the passed in decision table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>decisionID</td>
<td>String</td>
<td>Sys ID of the Decision Table record from the Decision Tables [sys_decision] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Object | Decision inputs from the Decision Inputs [sys_decision_input] table associated with the decision table. You can find the inputs associated with a decision table using the Decision Inputs related list in the Decision Tables [sys_decision] table. Inputs contain these key-value pairs.  
  • searchField: String. Used for internal purposes.  
  • defaultValue: String. Default value for the input.  
  • show_ref_finder: Boolean. Used for internal purposes.  
  • use_dependent: Boolean. Used for internal purposes. |
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type: String</td>
<td>Input data type.</td>
</tr>
<tr>
<td>mandatory: Boolean</td>
<td>True if the input is mandatory; otherwise false.</td>
</tr>
<tr>
<td>extended: Boolean</td>
<td>True if the input extends another field; otherwise false.</td>
</tr>
<tr>
<td>local: Boolean</td>
<td>Used for internal purposes.</td>
</tr>
<tr>
<td>sys_class_name: String</td>
<td>Used for internal purposes.</td>
</tr>
<tr>
<td>reference: String</td>
<td>Reference table used if the input type is reference.</td>
</tr>
<tr>
<td>dependent_on: String</td>
<td>Field that document ID and choice inputs depend on.</td>
</tr>
<tr>
<td>data_structure: String</td>
<td>Used for internal purposes.</td>
</tr>
<tr>
<td>readonly: Boolean</td>
<td>True if the input is read-only; otherwise false.</td>
</tr>
<tr>
<td>id: String</td>
<td>Sys ID of the input from the Decision Inputs [sys_decision_input] table.</td>
</tr>
<tr>
<td>type_label: String</td>
<td>Input data type label.</td>
</tr>
<tr>
<td>table: String</td>
<td>Reference field table if the input type is reference.</td>
</tr>
<tr>
<td>order: Number</td>
<td>Order in which the system evaluates inputs.</td>
</tr>
<tr>
<td>ref_qual: String</td>
<td>Condition applied to the reference table.</td>
</tr>
<tr>
<td>reference_display: String</td>
<td>Used for internal purposes.</td>
</tr>
<tr>
<td>choiceOption: String</td>
<td>Method for users to see a list of suggested values.</td>
</tr>
<tr>
<td>label: String</td>
<td>Label of the input record.</td>
</tr>
<tr>
<td>hint: String</td>
<td>Hint text used to help users understand the input required.</td>
</tr>
<tr>
<td>name: String</td>
<td>Internal name of the input record.</td>
</tr>
<tr>
<td>attributes: Array</td>
<td>Field attributes that apply to the input record.</td>
</tr>
<tr>
<td>maxsize: Number</td>
<td>Maximum number of characters allowed in the field.</td>
</tr>
<tr>
<td>columnName: String</td>
<td>Name of the input.</td>
</tr>
</tbody>
</table>

Format: JSON
var dt = new sn_dt.DecisionTableAPI();
var response = dt.getInputs("3c4464a7b701230034d1da23ee11a993");
gs.info(JSON.stringify(response));

Output:

```json
{"result": [{
  "searchField": "number",
  "defaultValue": "",
  "show_ref_finder": false,
  "use_dependent": false,
  "type": "reference",
  "mandatory": false,
  "extended": false,
  "local": false,
  "sys_class_name": "",
  "reference": "change_request",
  "dependent_on": "",
  "data_structure": "",
  "readonly": false,
  "id": "634420a7b701230034d1da23ee11a94f",
  "type_label": "Reference",
  "table": "",
  "order": 100,
  "ref_qual": "",
  "reference_display": "Change Request",
  "choiceOption": "",
  "label": "Change request",
  "hint": "",
  "name": "change_request",
  "attributes": {
    "element_mapping_provider": "com.glide.decision_table.data.DecisionTableVariableMapper",
    "maxsize": 32,
    "columnName": ""
  }
}]
```

**DecisionTableAPI - getQuestions(String decisionID)**

Returns the decisions from the Decision [sys_decision_question] table associated with the passed in decision table.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>decisionID</td>
<td>String</td>
<td>Sys ID of the Decision Table record from the Decision Tables [sys_decision] table.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Decisions from the Decision [sys_decision_question] table associated with the decision table. You can find the decisions associated with a decision table using the Decisions related list in the Decision Tables [sys_decision] table. Decision records contain these key-value pairs.</td>
</tr>
<tr>
<td></td>
<td>• defaultAnswer: Boolean. True if this decision is the default for the decision table; otherwise false.</td>
</tr>
<tr>
<td></td>
<td>• condition: String. Conditions required to reach this decision. The fields available for the condition are the decision inputs that are associated with this decision table.</td>
</tr>
<tr>
<td></td>
<td>• answer: Object. Answer record associated with the decision. Answer objects contain these key-value pairs.</td>
</tr>
<tr>
<td></td>
<td>◦ valid: Boolean. True if the record exists in the system; otherwise false.</td>
</tr>
<tr>
<td></td>
<td>◦ label: String. Answer record label.</td>
</tr>
<tr>
<td></td>
<td>◦ value: String. Sys ID of the answer record.</td>
</tr>
<tr>
<td></td>
<td>◦ table: String. Table containing the answer record.</td>
</tr>
<tr>
<td></td>
<td>• decisionTable: String. Sys ID of the decision table from the Decision Tables [sys_decision] table associated with the decision.</td>
</tr>
<tr>
<td></td>
<td>• domain: String. Domain in which the decision is visible.</td>
</tr>
<tr>
<td></td>
<td>• active: Boolean. True if the decision record is active; otherwise false.</td>
</tr>
<tr>
<td></td>
<td>• id: String. Sys ID of the decision record from the Decision [sys_decision_question] table.</td>
</tr>
<tr>
<td></td>
<td>• label: String. Label for the decision record.</td>
</tr>
<tr>
<td></td>
<td>• order: Number. Order in which the system evaluates decisions.</td>
</tr>
</tbody>
</table>

Format: JSON
var dt = new sn_dt.DecisionTableAPI();
var response = dt.getQuestions("3c4464a7b701230034d1da23ee11a993");
gs.info(JSON.stringify(response));

Output:

```javascript
{"result": [{
    "defaultAnswer": true,
    "condition": "change_request.state=-3^EQ",
    "answer": {
        "valid": true,
        "label": "CAB Approval",
        "value": "45358a5db701230034d1da23ee11a938",
        "table": "chg_approval_def"
    },
    "decisionTable": "3c4464a7b701230034d1da23ee11a993",
    "domain": "global",
    "active": true,
    "state": null,
    "id": "2764a4a7b701230034d1da23ee11a989",
    "label": "Requires CAB Approval",
    "order": 0
}]
}
```

**Document - Scoped, Global**

The Document API provides methods to initialize a PDF, add content, and close the PDF. After adding content, the document can be attached to a target record.

This API is part of the ServiceNow PDF Generation Utilities plugin (com.snc.apppdfgenerator) and is provided within the sn_pdfgeneratorutils namespace. The plugin is activated by default.

This API depends on the a suite of classes to build various elements comprising a PDF.

- **Cell** – Creates a Cell object as a cell in a table. You can use this API to format the cell and include additional blocks, such as paragraphs and images.

- **Color** – Creates a Color object used to define color attributes that you can apply to elements in a PDF: such as cells, tables, and lines.

- **Image** – Creates an Image object representing an image and its layout insert in a PDF. Enables defining attributes such as scale, alignment, and border color.

- **Line** – Creates a Line object using methods to draw a line in a PDF.
• **Paragraph** – Creates a Paragraph object representing a block of text in a PDF.

• **PdfPage** – Creates a PdfPage object representing a PDF page and its attributes; such as size, width, and color.

• **Style** – Creates a style for defining properties such as font size, border, and alignment. You can apply the same style to multiple objects simultaneously.

• **Table** – Creates a Table object to add to a PDF document. Defines the data to use in each cell and sets styles, margins, and alignment.

The following example shows how to create a basic PDF using the Document API and several components, such as a table, cell, and paragraph. The result is a list of Incidents from the Incident [incident] table listed in a PDF. You can test this example in your instance if you replace `<sys_id>` with the sys_id of an incident record.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var whiteColor = sn_pdfgeneratorutils.Color([1,1,1]);
var greyColor = sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
var headerBgColor = new sn_pdfgeneratorutils.Color([0.4,0.6,0.8]);

// Query the Incident table
var gr = new GlideRecord("incident");
gr.query();

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [70,200], false);

var headerStyle = new sn_pdfgeneratorutils.Style;
headerStyle setBackgroundColor(headerBgColor);
headerStyle.setTextAlignment("text-center");
headerStyle.setBold();
headerStyle.setFontColor(whiteColor);

var nParagraph = new sn_pdfgeneratorutils.Paragraph("Number");
var sParagraph = new sn_pdfgeneratorutils.Paragraph("Short Description");

var hdrCell1 = new sn_pdfgeneratorutils.Cell;
var hdrCell2 = new sn_pdfgeneratorutils.Cell;
```
The PDF attachment is listed in the Attachments [sys_attachment] table.
Document - Document(PdfPage pageSize)

Instantiates a Document object and generates a PDF document.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pageSize</td>
<td>PdfPage</td>
<td>PDF page size.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>PDF document.</td>
</tr>
</tbody>
</table>

The following example shows how to create a Document object and return a PDF.

```javascript
var pageSize = new sn_pdfs generatorutils.PdfPage("A4");
var document = new sn_pdfs generatorutils.Document(pageSize);
```
Document – addAndStartNewPage()

Adds a page to the document by terminating the current page and creating a new one.

Additional methods for adding a new page in a document:

- addNewPage() – Adds a new blank page to the document. Use to force a page break to start a new chapter or section in your document.
- addNewPageAtIndex() – Adds a new page at the specified index of the document. For example, setting the index to 6 inserts a page six or inserts the page at the position of the existing page six in a document. The original page six becomes page seven.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add a new page to a document. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var para1 = new sn_pdfgeneratorutils.Paragraph("This text lands on the first page.");
var para2 = new sn_pdfgeneratorutils.Paragraph("This text lands on the new page.");
document.addParagraph(para1);
document.addAndStartNewPage();
document.addParagraph(para2);

// save pdf as attachment to target record in the Incident table
document.saveAsAttachment("incident", "<record_sys_id>", "newPage.pdf");
```
**Document – addAuthor(String author)**

Adds a name to the author field in PDF document properties.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>author</td>
<td>String</td>
<td>Name of the document’s author.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add a name to the author field in PDF document properties. For a document usage example, see `Document` API.

```javascript
var author = "John Do";
document.addAuthor(author);
```

**Document – addImage(Image image)**

Adds an image to a document.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>image</td>
<td>Image</td>
<td>Image to add to a document.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add an image to a document. For a document usage example, see `Document` API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare image using sys attachment
var image = new sn_pdfgeneratorutils.Image("<imgAttachment_sys_id>");
```
// add the image to the doc
document.addImage(image);

document.saveAsAttachment("incident", "<record_sys_id>", "docWithImage.pdf");

**Document – addNewLine()**

Adds a new empty line to the document.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

The following example shows how to add a new line to a document. For a document usage example, see **Document** API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");

var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

document.addNewLine();
```

**Document – addNewPage()**

Adds a new blank page to the document. Use to force a page break to start a new chapter or section in your document.

Additional methods for adding a new page in a document:

- **addAndStartNewPage()** – Adds a page to the document by terminating the current page and creating a new one.

- **addNewPageAtIndex()** – Adds a new page at the specified index of the document. For example, setting the index to 6 inserts a page six or inserts the page at the position of the existing page six in a document. The original page six becomes page seven.
The following example shows how to add a new blank page to a document. For a document usage example, see Document API.

Usage:
```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

document.addNewPage();
```

**Document – addNewPageAtIndex( Number index )**

Adds a new page at the specified index of the document. For example, setting the index to 6 inserts a page six or inserts the page at the position of the existing page six in a document. The original page six becomes page seven.

Additional methods for adding a new page in a document:
- ```addAndStartNewPage()``` – Adds a page to the document by terminating the current page and creating a new one.
- ```addNewPage()``` – Adds a new blank page to the document. Use to force a page break to start a new chapter or section in your document.
The following example shows how to add a new PDF page to position 6 of a document. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");

var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var index = 6;

document.addNewPageAtIndex(index);
```

### Document – addParagraph(Paragraph paragraph)

Adds a paragraph to a document.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paragraph</td>
<td>Paragraph</td>
<td>Block of text provided as a paragraph object.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add a paragraph to a document. For a document usage example, see Document API.

```javascript
var para = "Lorem ipsum dolor sit amet.";

document.addParagraph(para);
```

### Document – addTable(Table table)

Adds a table to a document.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>Table</td>
<td>Table to be inserted into the document.</td>
</tr>
</tbody>
</table>
The following example shows how to add a table to a document. See the Table API for more details on how to define a table. For a document usage example, see Document API.

```javascript
var table = new sn_pdffgeneratorutils.Table([70,200], false);
document.addTable(table);
```

**Document – close()**

Closes a document.

The following example shows how to close a document. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdffgeneratorutils.PdfPage("A4");
var document = new sn_pdffgeneratorutils.Document.createDocument(pageSize);
document.close();
```


Creates a document with the specified page size.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The following example shows how to create a document. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);
```

**Document – getPageCount()**

Gets the number of pages in the document.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of pages in the document.</td>
</tr>
</tbody>
</table>

The following example shows how to get the page count of a nine-page document. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);
var count = document.getPageCount();
gs.info("The number of pages is "+ count);
```

Output:

```
The number of pages is 9
```

**Document – getPageSize()**

Gets the default page size of the document.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String   | Value of the default page size set using the PdfPage API. Possible values:  
- A4 – 595 x 842 points  
- EXECUTIVE – 522 x 756 points  
- LETTER – 612 x 792 points  
- LEDGER – 792 x 1224 points |

The following example shows how to get the page size of a document. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);
var pagesize = document.getPageSize();
```

Document – isClosed()

Indicates whether a document is closed or open.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Flag that indicates whether a document is open or closed. Valid values:</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>• true: Document is closed.</td>
<td></td>
</tr>
<tr>
<td>• false: Document is open.</td>
<td></td>
</tr>
<tr>
<td>Default: true</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to get the page size of a document. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");

var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var closed = document.isClosed();
```

**Document – saveAsAttachment(String tableName, String tableSysId, String fileName)**

Attaches the document file to the specified target table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table on which to attach the document.</td>
</tr>
<tr>
<td>tableSysId</td>
<td>String</td>
<td>Sys_id of the record on which to attach the document.</td>
</tr>
<tr>
<td>fileName</td>
<td>String</td>
<td>Name of the document to attach.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

The following example shows how to attach a document to an incident record. For a document usage example, see Document API.
Document – setBaseDirection(String direction)

Sets the base text flow direction to reorder from based on character recognition

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>direction</td>
<td>String</td>
<td>Text flow direction. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LEFT_TO_RIGHT: Order text flow left to right. The text direction is only</td>
</tr>
<tr>
<td></td>
<td></td>
<td>reordered if left-to-right language characters are detected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RIGHT_TO_LEFT: Order text flow right to left. The text direction is only</td>
</tr>
<tr>
<td></td>
<td></td>
<td>reordered if right-to-left language characters are detected.</td>
</tr>
</tbody>
</table>

Default: LEFT_TO_RIGHT

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

The following example shows how to set text flow left to right. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);
document.setBaseDirection("RIGHT_TO_LEFT");
```

Document – setMargins(Number topMargin, Number rightMargin, Number bottomMargin, Number leftMargin)

Sets the page margin sizes in the document.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>topMargin</td>
<td>Number</td>
<td>Height of the top margin in points.</td>
</tr>
<tr>
<td>rightMargin</td>
<td>Number</td>
<td>Width of the right margin in points.</td>
</tr>
<tr>
<td>bottomMargin</td>
<td>Number</td>
<td>Height of the bottom margin in points.</td>
</tr>
<tr>
<td>leftMargin</td>
<td>Number</td>
<td>Width of the left margin in points.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set page margins in a document. For a document usage example, see [Document API](#).

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);
document.setMargins(72,36,36,36);
```

### DocumentListEntryService - Scoped, Global

Provides methods for maintaining document templates in a document list.

This API requires the Document Management plugin (com.snc.platform_document_management) and is provided within the `sn_doc_services` namespace. For information, see [Document Services](#).

Before using the methods in this API, add a document template with its version and create a document list. Use the `SystemDocumentListEntry` API to add or remove documents from a document list.

Before using the methods in this API, you must add a document with its default version and create a document list.

- **SystemDocument** – Define a document object.
- **DocumentService** – Add, update, or delete a document.
- **DocumentVersionService** – Add, update, or delete a document version.
• **SystemDocumentList** – Define a document list.

• **DocumentListService** – Add, update, or delete a document list.

To define a document as a template:

• Define the document object setting the **SystemDocument – template()** method to true. Create or update the document record with the methods in the **DocumentService API**.

• Select the Template check box of a document in the Documents [ds_document] table.

**DocumentListEntryService** - **DocumentListEntryService()**

InstANTIATES a DocumentListEntryService object.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

The following example shows how to instantiate a **DocumentListEntryService** object.

```javascript
var s = new sn_doc_services.DocumentListEntryService();
```

**DocumentListEntryService** - **createDocumentListEntry(SystemDocumentListEntry entry)**

Adds a document template entry to a document list.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>entry</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;request_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>&lt;Object&gt;.message</td>
<td>Message confirming success or error.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.status</td>
<td>Status indicating whether the operation is successful. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• success - The operation was successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – The operation was not successful. The message provides details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

The following example shows how to define a document entry and add it to a document list. See also SystemDocumentListEntry.

```javascript
var docListEntry = new
  sn_doc_services.SystemDocumentListEntry('21afdde2460fc10f877a6fed1c2b0dd');
docListEntry.document("9acd7efa24587410f877a6fed1c2b060");
docListEntry.description('birth certificate');

var dlEntrySvc = new sn_doc_services.DocumentListEntryService();
gs.info(JSON.stringify(dlEntrySvc.createDocumentListEntry(docListEntry), null, 2));
```

Output:

```json
{
   "message": "Create List Entry , List Entry sysId : d3f0b13624e8fc10f877a6fed1c2b0d9 is successful. ",
   "request_id": "d3f0b13624e8fc10f877a6fed1c2b0d9",
   "status": "success"
}
```
DocumentListEntryService - deleteDocumentListEntry(String listEntryId)


### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Success or error message.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>&lt;Object&gt;.message</td>
<td>Message confirming success or error.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.status</td>
<td>Status indicating whether the operation is successful. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• success - The operation was successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – The operation was not successful. The message provides details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

The following example shows how to delete a document list record. See also SystemDocumentList.

```javascript
var listEntryId = 'd3f0b13e24e8fc10f877a6fed1c2b0d9';

var dlEntrySvc = new sn_doc_services.DocumentListEntryService();
gs.info(JSON.stringify(dlEntrySvc.deleteDocumentListEntry(listEntryId), null, 2));
```

Output:
{  
  "message": "Deleting an entry for the given documentListEntrySysId : d3f0b13624e8fc10f877a6fed1c2b0d9 is successful.",
  "status": "success"
}

**DocumentListEntryService- updateDocumentListEntry(String listEntryId, SystemDocumentListEntry entry)**

Updates the field values of an existing document list entry.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entry</td>
<td>SystemDocumentListEntry</td>
<td>One or more properties representing fields of a document list entry object.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Success or error message.</td>
</tr>
<tr>
<td>&lt;Object&gt;.message</td>
<td>Message confirming success or error. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.status</td>
<td>Status indicating whether the operation is successful. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• success - The operation was successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – The operation was not successful. The message provides details.</td>
</tr>
</tbody>
</table>
The following example shows how to update a document list entry description. See also `SystemDocumentListEntry`.

```javascript
var listEntryId = 'd3f0b13624e8fc10f877a6fed1c2b0d9';

var entry = new sn_doc_services.SystemDocumentListEntry();
entry.description('description update');

var dlEntrySvc = new sn_doc_services.DocumentListEntryService();
gs.info(JSON.stringify(dlEntrySvc.updateDocumentListEntry(listEntryId, entry), null, 2));
```

Output:

```json
{
  "message": "Updating an entry for the given documentListEntrySysId: d3f0b13624e8fc10f877a6fed1c2b0d9 is successful.",
  "request_id": "d3f0b13624e8fc10f877a6fed1c2b0d9",
  "status": "success"
}
```

### DocumentListService - Scoped, Global

Provides methods for creating, deleting, and updating a document list.

This API requires the Document Management plugin (com.snc.platform_document_management) and is provided within the `sn_doc_services` namespace. For information, see Document Services.

The Document Management plugin also supports creating lists of document templates to associate with your document. For example, a job application requiring multiple documents such as a diploma, ID, or passport. After you add your document list, you can add document templates:

- `SystemDocumentListEntry` – Define a document template list entry.
- `DocumentListEntryService` – Add or remove a document template list entry.

To define a document as a template:
• Define the document object setting the `SystemDocument - template()` method to true. Create or update the document record with the methods in the `DocumentService` API.

• Select the Template check box of a document in the Documents [ds_document] table.

The following APIs enable you to define and manage documents:

• `SystemDocument` – Define a document object.

• `DocumentService` – Add, update, or delete a document.

To define a document as a template:

• Define the document object setting the `SystemDocument - template()` method to true. Create or update the document record with the methods in the `DocumentService` API.

• Select the Template check box of a document in the Documents [ds_document] table.

See also `SystemDocumentList`.

**DocumentListService - createDocumentList(SystemDocumentList doc)**


The Document Management plugin also supports creating lists of document templates to associate with your document. For example, a job application requiring multiple documents such as a diploma, ID, or passport. After you add your document list, you can add document templates:

• `SystemDocumentListEntry` – Define a document template list entry.

• `DocumentListEntryService` – Add or remove a document template list entry.

Use the `createDocumentsFromList()` method to create documents from the document template list.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc</td>
<td>SystemDocumentList</td>
<td>One or more properties representing fields of a new record. The name property is required and can be set using the <code>SystemDocumentList constructor or name()</code> method.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
<td></td>
</tr>
</tbody>
</table>
|            |   "message": "String",
|            |   "request_id": "String",
|            |   "status": "String"
|            | }                                                                                                                                                                                                       |
| <Object>.message | Message confirming success or error.                                                                                                         |
|            | Data type: String                                                                                                                           |
|            | Data type: String                                                                                                                           |
| <Object>.status | Status indicating whether the operation is successful. Possible values:                                                                   |
|            | • success - The operation was successful.                                                                                                   |
|            | • failure – The operation was not successful. The message provides details.                                                                                                                               |
|            | Data type: String                                                                                                                           |

The following example shows how to create a document list. See also SystemDocumentList.

```javascript
var dL = new sn_doc_services.SystemDocumentList('My document list');

// Define the document list field
dL.description('description');

var docList = new sn_doc_services.DocumentListService();
gs.info(JSON.stringify(docList.createDocumentList(dL), null, 2));
```

Output:

```json
{
   "message": "Create List for the given name : My document list, List sysId : b2c021a9246b3c10f877a6fed1c2b0b1 is successful."
}
DocumentListService - createDocumentsFromList(String listSysId, String tableName, String tableSysId)

Creates documents from a list of document templates.

⚠️ **Note:** Before using this method, create a document template list using the `createDocumentList()` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table containing the record on which add a list of document template references.</td>
</tr>
<tr>
<td>tableSysId</td>
<td>String</td>
<td>Sys_id of the record containing the content from which to add a list of document template references.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Success or error message.</td>
</tr>
<tr>
<td>&lt;Object&gt;.message</td>
<td>Message containing the number of document templates in the document list. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.status</td>
<td>Status indicating whether the operation is successful.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• success - The operation was successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – The operation was not successful. The <strong>message</strong> provides details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

The following example shows how add a list of document template references to a record. The documentation is added to a record in the Incidents [incident] table.

```javascript
var result = new sn_doc_services.DocumentListService().createDocumentsFromList('21afddea2460fc10f877a6fed1c2b0dd', 'incident', 'd7158da0c0a8016700eef46c8d1f3661');
gs.info(JSON.stringify(result, null, 2));
```

**Output:**

```javascript
{
    "message": "Created 1 document references for the given table name: incident, table sysId : d7158da0c0a8016700eef46c8d1f3661 with list SysId 21afddea2460fc10f877a6fed1c2b0dd",
    "request_id": "21afddea2460fc10f877a6fed1c2b0dd",
    "status": "success"
}
```

**DocumentListService - deleteDocumentList(String listSysId)**


**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Success or error message.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| {}         | {  "message": "String",
    "request_id": "String",
    "status": "String"
        }                                                                                                                                 |
| <Object>.message | Message confirming success or error.                                                                                                          |
|        | Data type: String                                                                                                                              |
|        | Data type: String                                                                                                                              |
| <Object>.status | Status indicating whether the operation is successful. Possible values:                                                                      |
|        | • success - The operation was successful.                                                                                                     |
|        | • failure – The operation was not successful. The message provides details.                                                                      |
|        | Data type: String                                                                                                                              |

The following example shows how to delete a document list record. See also SystemDocumentList.

```javascript
var docListID = 'b2c021a924683c10f877a6fed1c2b0b1';

var docListSvc = new sn_doc_services.DocumentListService();
gs.info(JSON.stringify(docListSvc.deleteDocumentList(docListID), null, 2));
```

Output:

```javascript
{
    "message": "Delete List for the given sysId : b2c021a924683c10f877a6fed1c2b0b1 is successful.",
    "request_id": "b2c021a924683c10f877a6fed1c2b0b1",
    "status": "success"
}
```

**DocumentListService - DocumentListService()**

Instantiates a DocumentListService object.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to instantiate a `DocumentListService` object. See also `SystemDocumentList`.

```javascript
var s = new sn_doc_services.DocumentListService();
```

**DocumentListService - updateDocumentList(String docSysId, SystemDocumentList doc)**

Updates the field values of an existing document list.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc</td>
<td>SystemDocument</td>
<td>One or more properties representing document list fields to be updated.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Success or error message.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;request_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>&lt;Object&gt;.message</td>
<td>Message confirming success or error.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.status</td>
<td>Status indicating whether the operation is successful. Possible values:</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>success</td>
<td>The operation was successful.</td>
</tr>
<tr>
<td>failure</td>
<td>The operation was not successful. The message provides details.</td>
</tr>
</tbody>
</table>

Data type: String

The following example shows how to update a document list name and description. See also SystemDocumentList.

```javascript
var docListID = 'b2c021a924683c10f877a6fed1c2b0b1';

var docListUpdate = new sn_doc_services.SystemDocumentList();
docListUpdate.name('name change');
docListUpdate.description('description update');

var docListSvc = new sn_doc_services.DocumentListService();
gs.info(JSON.stringify(docListSvc.updateDocumentList(docListID , docListUpdate), null, 2));
```

Output:

```json
{
  "message": "Update List for the given sysId : b2c021a924683c10f877a6fed1c2b0b1 is successful. ",
  "request_id": "b2c021a924683c10f877a6fed1c2b0b1",
  "status": "success"
}
```

**DocumentReferenceService - Scoped, Global**

Provides methods for managing document references within a target table, such as the Incidents [incident] or Knowledge [kb_knowledge] table. You can use document references to associate information with a record.

This API requires the Document Management plugin (com.snc.platform_document_management) and is provided within the sn_doc_services namespace. For information, see Document Services.

The following APIs enable you to define and manage documents:

- **SystemDocument** – Define a document object.
- **DocumentService** – Add, update, or delete a document.
DocumentReferenceService - DocumentReferenceService()

Instantiates a DocumentReferenceService object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to instantiate a DocumentReferenceService object.

```javascript
var result = new sn_doc_services.DocumentReferenceService();
```

DocumentReferenceService - addDocumentReference(String docSysId, String targetTable, String tableSysId)

Adds a document reference to a target table record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>targetTable</td>
<td>String</td>
<td>Name of the target table on which to provide a document reference.</td>
</tr>
<tr>
<td>tableSysId</td>
<td>String</td>
<td>Sys_id of the target table record to which the document reference is to be added.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

```javascript
{
  "message": "String",
  "status": "String"
}
```
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;.message</code></td>
<td>Message confirming success or error. If successful, includes the sys_id of the record the document reference was added to. Data type: String</td>
</tr>
</tbody>
</table>
| `<Object>.status` | Status indicating whether the operation is successful. Possible values:  
  • success - The operation was successful.  
  • failure – The operation was not successful. The `message` provides details. Data type: String                                                   |

The following example shows how to add a document reference to a record in the Incidents `[incident]` table.

```javascript
var docReference = new sn_doc_services.DocumentReferenceService();

var result = docReference.addDocumentReference("<docSysId>", "incident", ",<tableSysId>");

gs.info(JSON.stringify(result, null, 2));
```

Output:

```javascript
{
  "message": "Adding an entry to DocumentReference for the given docId to target table: incident, sysId : <tableSysId> is successful.",
  "status": "success"
}
```

**DocumentReferenceService - removeDocumentReference(String docSysId, String targetTable, String tableSysId)**

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>targetTable</td>
<td>String</td>
<td>Name of the table containing the document reference. This information is listed with the document in the Document References [ds_document_reference] table.</td>
</tr>
<tr>
<td>tableSysId</td>
<td>String</td>
<td>Sys_id of the record from which the document is referenced. You can find related information listed with the document in the Document References [ds_document_reference] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Success message. Error message otherwise.</td>
</tr>
<tr>
<td>&lt;Object&gt;.message</td>
<td>Message confirming success or error. If successful, includes the sys_id of the record the document reference was removed from.</td>
</tr>
<tr>
<td>&lt;Object&gt;.status</td>
<td>Status indicating whether the operation is successful. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• success - The operation was successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – The operation was not successful. The message provides details.</td>
</tr>
</tbody>
</table>

The following example shows how to remove a document reference from a record in the Incidents [incident] table.
var docReference = new sn_doc_services.DocumentReferenceService();

var result = docReference.removeDocumentReference("<docSysId>", "incident",
  "<tableSysId>");

gs.info(JSON.stringify(result, null, 2));

Output:

{
  "message": "Deleting an entry to DocumentReference for the given docId to target table : incident, sysId : <tableSysId> is successful.",
  "status": "success"
}

DocumentService - Scoped, Global

Provides methods for creating, deleting, and updating a document.

This API requires the Document Management plugin (com.snc.platform_document_management) and is provided within the sn_doc_services namespace. For information, see Document Services.

A document is a collection of information about a document record. The methods used to create or update a record modify fields using the SystemDocument object.

Document content is managed using versions. The following APIs enable you to define and manage document versions:

- SystemDocumentVersion – Define a document version as the source of the document content. Each version is an element containing the document content and is provided using a single URL or attachment. An attachment can only be added in the Document Versions [ds_document_version] table UI and not with the API.

- DocumentVersionService – Document content is managed using versions.

Use the DocumentReferenceService API to manage documents referenced in a target table, such as the Incidents [incident] or Knowledge [kb_knowledge] table.

The Document Management plugin also supports creating lists of document templates to associate with your document. For example, a job application requiring multiple documents such as a diploma, ID, or passport.


- DocumentListService – Add or remove a document template list.
• **SystemDocumentListEntry** – Define a document template list entry.
• **DocumentListEntryService** – Add or remove a document template list entry.

**DocumentService - DocumentService()**
Instantiates a `DocumentService` object.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to instantiate a `DocumentService` object.

```csharp
var s = new sn_doc_services.DocumentService();
```

**DocumentService - createDocument(SystemDocument doc)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc</td>
<td><code>SystemDocument</code></td>
<td>One or more properties representing fields of a new record. The name property is required and can be set using the <code>SystemDocument</code> constructor or <code>name()</code> method.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;.message</code></td>
<td>Message confirming success or error. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|      | • Create document sysId : `<sys_id>` is successful.  
|      | • Document Name is mandatory and should be valid.  
|      | • Document Name is mandatory and it should not be null or empty.  
|      | • `<SystemDocument` property requiring sys_id input> withSysId: "<sys_id>" does not exist  
|      | Data type: String |

Data type: String

<Object>.statusStatus indicating whether the operation is successful. Possible values:  
• success - The operation was successful.  
• failure – The operation was not successful. The message provides details.  
Data type: String

The following example shows how to populate `SystemDocument` object properties and create a new document record.

```javascript
var d = new sn_doc_services.SystemDocument('My document');

// Define the document fields
var reviewers = '62826bf03710200044e0bfc8bcbe5df1,a8f98bb0eb32010045e1a5115206fe3a';
d.description('description');
d.classification('restricted');
d.state('review');
d.department('93b25282c0a8000b0b55c8ab34e2f1e6');
d.template(false);
d.type('policy');
d.reviewers(reviewers);
d.audience('external');
```
var s = new sn_doc_services.DocumentService();
gs.info(JSON.stringify(s.createDocument(d), null, 2));

Output:
{
  "message": "Create document sysId : 1040420224503410f877a6fed1c2b031 is successful.",
  "request_id": "1040420224503410f877a6fed1c2b031",
  "status": "success"
}

DocumentService - deleteDocument(String docSysId)

Removes a document record from the Documents [ds_document] table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Success or error message.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;request_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>&lt;Object&gt;.message</td>
<td>Message confirming success or error.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.status</td>
<td>Status indicating whether the operation is successful. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• success - The operation was successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – The operation was not successful. The message provides details.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to delete an existing document record.

```javascript
var docid = "<sys_id>";
var svc = new sn_doc_services.DocumentService();
gs.info(JSON.stringify(svc.deleteDocument(docid), null, 2));
```

Output:

```json
{
  "message": "Delete document sysId : <sys_id> is successful.",
  "request_id": "<sys_id>",
  "status": "success"
}
```

DocumentService - updateDocument(String docSysId, SystemDocument doc)

Updates the field values of an existing document record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc</td>
<td>SystemDocument</td>
<td>One or more properties representing document fields to be updated.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Success or error message.</td>
</tr>
</tbody>
</table>

```json
{
  "message": "String",
  "request_id": "String",
  "status": "String"
}
```
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.status</code></td>
<td>Status indicating whether the operation is successful.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• success - The operation was successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – The operation was not successful. The message provides details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

The following example shows how to change a document name. See also SystemDocument.

```javascript
var dId = "19aab54e24103410f877a6fed1c2b03d";
var d = new sn_doc_services.SystemDocument();
d.name("c22.txt");
var s = new sn_doc_services.DocumentService();
gs.info(JSON.stringify(s.updateDocument(dId, d), null, 2));
```

Output:

```javascript
{
  "message": "Update document sysId : 19aab54e24103410f877a6fed1c2b03d is successful.",
  "request_id": "19aab54e24103410f877a6fed1c2b03d",
  "status": "success"
}
```

### DocumentVersionService - Scoped, Global

Provides a service for creating and deleting document versions. Each version is an element containing the document content and is provided using a single URL or attachment. An attachment can only be added in the Document Versions [ds_document_version] table UI and not with the API.

This API requires the Document Management plugin (com.snc.platform_document_management) and is provided within the sn_doc_services namespace. For information, see Document Services.
Before creating a document version, you must first define and add a document record.

- **SystemDocument** – Define a document object.
- **DocumentService** – Add, update, or delete a document.

**DocumentVersionService - DocumentVersionService()**

Instantiates a `DocumentVersionService` object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to instantiate a `DocumentVersionService` object.

```javascript
var versionSvc = new sn_doc_services.DocumentVersionService();
```


**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

```json
{
    "message": "String",
    "request_id": "String",
    "status": "String"
}
```
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;.message</code></td>
<td>Message confirming success or error. If successful, includes the sys_id of the document record and the sys_id of its new document version. Data type: String</td>
</tr>
</tbody>
</table>
| `<Object>.status`         | Status indicating whether the operation is successful. Possible values:  
  • success - The operation was successful.  
  • failure – The operation was not successful. The message provides details. Data type: String                                                                 |

The following example shows how to populate SystemDocumentVersion object properties and create a new document version. See also SystemDocumentVersion.

```javascript
var versionDefinition = new sn_doc_services.SystemDocumentVersion('491efbee24187410f877a6fed1c2b0a9');
versionDefinition.type('url');
versionDefinition.url('http://one/1.0');
var versionSvc = new sn_doc_services.DocumentVersionService();
gs.info(JSON.stringify(versionSvc.createDocumentVersion(versionDefinition), null, 2));

Output:
```

```javascript
{
  "message": "Adding an entry to DocumentVersion for the given docId : 491efbee24187410f877a6fed1c2b0a9, sysId : d5a340d9242cf810f877a6fed1c2b069 is successful.",
  "request_id": "d5a340d9242cf810f877a6fed1c2b069",
  "status": "success"
}
```
DocumentVersionService - deleteDocumentVersion(String docVersionSysId)

Deletes a document version.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Success message. Error message otherwise.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>&lt;Object&gt;.message</td>
<td>Message confirming success or error. If successful, includes the sys_id of the deleted document version.</td>
</tr>
<tr>
<td>&lt;Object&gt;.status</td>
<td>Status indicating whether the operation is successful. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• success - The operation was successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – The operation was not successful. The message provides details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

The following example shows how to delete an existing document version.

```javascript
var vId = "<versionSysId>";
var versionSvc = new sn_doc_services.DocumentVersionService();

gs.info(JSON.stringify(versionSvc.deleteDocumentVersion(vId), null, 2));
```

Output:

```javascript
{
   "message": "Deleting an entry to DocumentVersion for the given documentVersionSysId : , sysId : <versionSysId> is successful.",
```
DSCScriptableAPI - Global

Provides methods for checking the health of your domain configuration for the instance.

The DSCScriptableAPI is part of the Domain Separation Center dashboard and is available in the SNC global namespace.

The com.glide.domain.msp_extensions.installer plugin must be activated for Domain Separation Center dashboard availability.

DSCScriptableAPI - chkDataInvalidDomainForTables(String tableNames)

Checks a list of comma-separated tables for invalid domains.

View records that reference invalid domains in the Domain Log [syslog_domain] table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableNames</td>
<td>String</td>
<td>Comma-separated list of table names.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GUID of any invalid references; OK otherwise</td>
</tr>
</tbody>
</table>

```
var dsc = new SNC.DSCScriptableAPI();
dsc.chkDataInvalidDomainForTables('task, sys_user');
```

DSCScriptableAPI - chkDataInvalidPathForTables(String tableNames)

Checks a list of comma-separated tables for invalid domain paths.

Records that have an invalid domain path are logged in the syslog_domain table.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Comma-separated list of table names.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GUID if there are invalid paths; OK otherwise</td>
</tr>
</tbody>
</table>

```javascript
var dsc = new SNC.DSCScriptableAPI();
dsc.chkDataInvalidPathForTables('task, sys_user');
```

DSCScriptableAPI - DSCScriptableAPI()

Instantiates a new DSCScriptable object.

DurationCalculator - Global

Provides methods for calculating durations and due dates.

DurationCalculator - calcDuration(Number seconds)

Calculates an end date and time based on current start date and time and passed in duration (seconds).

This method also sets the `this.endDateDateTime`, `this.seconds`, and `this.totalSeconds` properties with the updated values. Prior to calling this method, you must call `setStartDateTime()` with the start time to use in the computation.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>seconds</td>
<td>Number</td>
<td>Number of seconds to add to the start date and time to compute the end date and time, seconds, and total seconds values.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>False if the input value is not a number.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2020-05-01 00:00:00");
var dc = new DurationCalculator();
dc.setStartDateTime(gdt);

if(!dc.calcDuration(2*24*3600)){ // 2 days
  gs.log("*** Error calculating duration");
  return;
}
var secs = dc.getSeconds();
var totalSecs = dc.getTotalSeconds();

gs.print("***SCHEDULE DURATION: SECS=" + secs + " TOTALSECS=" + totalSecs + " ENDTIME = " + endDateTime);
```

Output

```
***SCHEDULE DURATION: SECS=172800 TOTALSECS=970534 ENDTIME = 2020-05-03 00:00:00
```

DurationCalculator - calcRelativeDueDate(GlideDateTime start, Number days, String endTime)

Calculates the due date starting at the passed in start time and adding the number of days using the current schedule and time zone.

Called from relative duration definitions, initiated by calcRelativeDuration(), as calculator.calcRelativeDueDate(calculator.startDateTime, days). Once the day that the work is due is determined, the method sets the time to the passed in endTime of that day. If there are not enough days left in the schedule, uses the last day in the schedule.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>GlideDateTime</td>
<td>GlideDateTime object that contains the start date for the computation.</td>
</tr>
<tr>
<td>days</td>
<td>Number</td>
<td>Number of days to add to the start date.</td>
</tr>
<tr>
<td>endTime</td>
<td>String</td>
<td>Time of day that the work is due on the computed due date. If blank,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>defaults to the end of the work day. Format: HH:mm:ss</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates if the completion date is within the schedule. Possible</td>
</tr>
<tr>
<td></td>
<td>values:</td>
</tr>
<tr>
<td></td>
<td>• true: Completion date is within the schedule.</td>
</tr>
<tr>
<td></td>
<td>• false: Completion date falls outside the schedule. Undefined if no</td>
</tr>
<tr>
<td></td>
<td>schedule was set prior to calling this method.</td>
</tr>
</tbody>
</table>

```javascript
var dc = new DurationCalculator();
var startTime = new GlideDateTime();

// Settings for calculations
// Optional: Specify the schedule to use for the following calculations
dc.setSchedule('08fcd0830a0a0b2600079f56b1adb9ae');
// Optional: Specify a different timezone to use
dc.setTimeZone("Los Angeles");
// Optional: Set a start date and time, otherwise the current time is assumed
dc.setStartDateTime("2020-04-10 08:00:00")
// Calculate end time, from number of seconds required in the schedule
dc.calcDuration(3*24*3600); // 3 days
dc.calcRelativeDueDate(startTime, "3", "07:00:00");
```

**DurationCalculator - calcRelativeDuration(String relativeDurationID)**
Calculates the duration using the specified relative duration script.
Upon completion, the `this.endDateTime` and `this.seconds` properties are set to indicate the results of the calculation.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relativeDurationID</td>
<td>String</td>
<td>sys_id of relative duration schedule (table cmn_relative_duration).</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>The result of the duration script.</td>
</tr>
</tbody>
</table>

```javascript
var dc = new DurationCalculator();
dc.calcRelativeDuration('08fcd0830a0a1b2600074f56b1ad7cb');
```

### DurationCalculator - calcScheduleDuration(String startTime, String endTime)

Returns the duration between the `startTime` and the `endTime` within the already-specified schedule and optionally overridden timezone.

This method also sets `this.endDateTime`, `this.seconds`, and `this.totalSeconds` in the current schedule object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startTime</td>
<td>String</td>
<td>Optional. Display value for the end time.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Uses the current date and time (set using <code>setStartDateTime()</code>).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You must pass a placeholder if this parameter it not passed, such as</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>dur.calcScheduleDuration('', endDateTime)</code>;</td>
</tr>
<tr>
<td>endTime</td>
<td>String</td>
<td>Optional. Display value for the start time.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Uses the current date and time (set using <code>setEndDateTime()</code>).</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Schedule duration. Returns 0 if endTime is before startTime. Unit: Seconds</td>
</tr>
</tbody>
</table>

```javascript
var dur = new DurationCalculator();

// Set 9-5 weekday schedule. This is the schedule in which endDateTime, seconds, and totalseconds is set
dur.setSchedule('08fcd0830a0a0b2600079f56b1ad9ae');
dur.calcScheduleDuration("2019-01-02 11:00:00", "2019-01-06 09:00:00");
var secs = dur.getSeconds();
var totalSecs = dur.getTotalSeconds();
var endDateTime = dur.getEndDateTime() + "";

gs.print("***SCHEDULE DURATION: SECS=\" + secs + " TOTALSECS=\" + totalSecs + " ENDTIME = \" + endDateTime);```

Output

```markdown
***SCHEDULE DURATION: SECS=97200 TOTALSECS=338400 ENDTIME = 2019-01-06 09:00:00
```

**DurationCalculator - calcScheduleDuration(GlideDateTime startTime, GlideDateTime endTime)**

Returns the duration between the startTime and the endTime within the already-specified schedule and optionally overridden timezone.

This method also sets this.endDateTime, this.seconds, and this.totalSeconds in the current schedule object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startTime</td>
<td>GlideDateTime</td>
<td>Optional. GlideDateTime object that contains the start time to use.  Default: Uses the current date and time (set using setStartDateTime()). You must pass a placeholder if this parameter it not passed, such as dur.calcScheduleDuration(&quot;&quot;, endDateTime);.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>endTime</td>
<td>GlideDateTime</td>
<td>Optional. GlideDateTime object that contains the end time to use. Default: Uses the current date and time (set using setEndDateTime()).</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Schedule duration. Returns 0 if endTime is before startTime. Unit: Seconds</td>
</tr>
</tbody>
</table>

```javascript
var startDateTime = new GlideDateTime("2020-11-02 11:00:00");
// Instantiate a new GlideDateTime object which has the end date as the current date and time
var endDateTime = new GlideDateTime();
var dur = new DurationCalculator();

// Set 9-5 weekday schedule. This is the schedule in which endTime, seconds, and totalSeconds is set
dur.setSchedule('08fcd0830a0a0b2600079f56b1adb9ae');
dur.calcScheduleDuration(startDateTime, endDateTime);
var secs = dur.getSeconds();
var totalSecs = dur.getTotalSeconds();

gs.print("***SCHEDULE DURATION: SECS=" + secs + " TOTALSECS=" + totalSecs + " ENDTIME = " + endDateTime);
```

### Output

```text
***SCHEDULE DURATION: SECS=293734.24 TOTALSECS=970534 ENDTIME = 2020-11-13 16:35:34
```

**DurationCalculator** - Constructor for **DurationCalculator** class.
DurationCalculator - getEndDateTime()

Gets the `endDateTime` property that was set by `calcDuration/calcRelativeDuration`, indicating the end date and time for the duration.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDateTime</td>
<td>The end datetime.</td>
</tr>
</tbody>
</table>

```javascript
var dc = new DurationCalculator();
dc.calcDuration(52);
gs.print(dc.getEndDateTime());
```

2012-04-17 20:57:27

DurationCalculator - getSeconds()

Returns the `this.seconds` property that was set by `calcDuration/calcRelativeDuration`, indicating the total number of seconds of work to be performed for the duration.

This is the total work time, not the total time between start and end times and may be used to determine percentages of the work time.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The total work time, in seconds.</td>
</tr>
</tbody>
</table>

```javascript
var startDate = new GlideDateTime("2020-11-02 11:00:00");
// Instantiate a new GlideDateTime object which has the end date as the current date and time
var endDate = new GlideDateTime();
var dur = new DurationCalculator();

// Set 9-5 weekday schedule. This is the schedule in which endDate, seconds, and totalSeconds is set
dur.setSchedule('08fcd0830a0a0b2600079f56b1adb9ae');
dur.calcScheduleDuration(startDate, endDate);
var secs = dur.getSeconds();
var totalSecs = dur.getTotalSeconds();

gs.print("***SCHEDULE DURATION: SECS=" + secs + " TOTALSECS=" + totalSecs + " ENDTIME = " + endDate);
```

Output

```
***SCHEDULE DURATION: SECS=293734.24 TOTALSECS=970534 ENDTIME = 2020-11-13 16:35:34
```

**DurationCalculator - getTotalSeconds()**

Returns the `totalSeconds` value that was set by a call to `calculate(record)`.
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The total number of seconds.</td>
</tr>
</tbody>
</table>

```javascript
var startDateTime = new GlideDateTime("2020-11-02 11:00:00");
// Instantiate a new GlideDateTime object which has the end date as the current date and
time
var endDateTime = new GlideDateTime();
var dur = new DurationCalculator();

// Set 9-5 weekday schedule. This is the schedule in which endDateTime, seconds, and
totalseconds is set
dur.setSchedule('08fcd0830a0a0b2600079f56b1adb9ae');
dur.calcScheduleDuration(startDateTime, endDateTime);
var secs = dur.getSeconds();
var totalSecs = dur.getTotalSeconds();

gs.print("***SCHEDULE DURATION: SECS=\" + secs + " TOTALSECS=\" + totalSecs + " ENDTIME = \" +
endDateTime);```

Output

```
***SCHEDULE DURATION: SECS=293734.24 TOTALSECS=970534 ENDTIME = 2020-11-13 16:35:34
```

**DurationCalculator - isAfter(GlideDateTime dt, String tm)**

Compares the passed in time to the time value in the passed in GlideDateTime object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dt</td>
<td>GlideDateTime or String</td>
<td>Either a GlideDateTime object or a display value date and time to compare to the passed in tm value. If you pass a display value date and time, the method converts it to a GlideDateTime object.</td>
</tr>
<tr>
<td>tm</td>
<td>String</td>
<td>Display value for the time to compare against the time value in the GlideDateTime object. Assumes same date. Format: HH:mm:ss.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates if the passed in time value (tm) is after the time in</td>
</tr>
<tr>
<td></td>
<td>the GlideDateTime object (dt). Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: tm is after dt.</td>
</tr>
<tr>
<td></td>
<td>• false tm is not after dt.</td>
</tr>
</tbody>
</table>

```javascript
var dc = new DurationCalculator();
gs.print(dc.isAfter("2020-04-10 08:00:00", "09:00:00"));

*** Script: true

DurationCalculator - setSchedule(String schedId, String timezone)
Sets the schedule and time zone to use for calculating the due date.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>schedId</td>
<td>String</td>
<td>Sys_id of the schedule to set. Located in the Schedule [cmn_schedule] table.</td>
</tr>
<tr>
<td>timezone</td>
<td>String</td>
<td>Optional. Time zone to set. Default: User’s time zone.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var startDateTime = new GlideDateTime("2020-11-02 11:00:00");
// Instantiate a new GlideDateTime object which has the end date as the current date and
time
var endDateTime = new GlideDateTime();
var dur = new DurationCalculator();
```
// Set 9-5 weekday schedule. This is the schedule in which endDateTime, seconds, and
totalseconds is set
dur.setSchedule('08fcd0830a0a0b2600079f56b1adb9ae');
dur.calcScheduleDuration(startDateTime, endDateTime);
var secs = dur.getSeconds();
var totalSecs = dur.getTotalSeconds();

gs.print("***SCHEDULE DURATION: SECS=\" + secs + " TOTALSECS=\" + totalSecs + " ENDTIME = " +
endDateTime);

Output

***SCHEDULE DURATION: SECS=293734.24 TOTALSECS=970534 ENDTIME = 2020-11-13 16:35:34

DurationCalculator - setStartDateTime(String start)
Sets the start date and time for the duration calculations.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>String</td>
<td>Display value for the start time in GMT for subsequent calculations.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var dc = new DurationCalculator();
dc.setStartDateTime("2020-04-10 08:00:00")

DurationCalculator - setStartDateTime(GlideDateTime description)
Sets the start date and time for the duration calculations.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>GlideDateTime</td>
<td>GlideDateTime object that contains the start time in GMT for subsequent calculations.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var startDateTime = new GlideDateTime("2020-11-02 11:00:00");
var dur = new DurationCalculator();
// Set 9-5 weekday schedule.
dur.setSchedule('08fcd0830a0a0b260079f56b1ad9ae');
dur.setStartDateTime(startDateTime);
```

### DurationCalculator - setTimeZone(String timezone)

Sets the time zone to use for calculating the due date.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timezone</td>
<td>String</td>
<td>Value of the time zone.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var dc = new DurationCalculator();
dc.setTimeZone("Los Angeles");
```

### DynamicTranslation - Client

The **DynamicTranslation** API provides methods that translate text, in real time, into multiple languages using translation service providers. This API is available for both standard clients and Angular-based Service Portal clients.

In addition, you can use this API to detect the language of a specific string and check whether the **DynamicTranslation** methods are enabled for a translation service. Use this API to create a seamless localization experience for your user interface, enabling one interface to service multiple countries.
Currently this API supports three translation service providers: Microsoft Azure Translator Service, IBM Watson Translator Service, and Google Cloud Translator Service. You can also configure other translation services within your instance and then use the DynamicTranslation API to translate your text.

To use this API you must activate the Dynamic Translation plugin. For information on this plugin and additional information on Dynamic Translation, refer to Dynamic translation overview. Also, to use this API in a Service Portal widget, you must inject the dynamicTranslation service into the widget client script function.

⚠️ Note: The name of the class to use in Service Portal clients is dynamicTranslation, while the name of the class to use in standard clients is DynamicTranslation.

**DynamicTranslation - getDetectedLanguage(String text, Object parms)**

Detects the language of the passed in text.

If you pass in a translator, the method uses that translation service to detect the source language. Otherwise, the detection is performed by the default translation service. Ensure that the text strings that you provide contain enough verbiage to enable proper language detection.

In addition to the detected language, the response contains a confidence level of the detection, along with other possible language alternatives. If a translator is not passed in, the method also returns the default translation service used to detect the language.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>Text to use to detect the language.</td>
</tr>
<tr>
<td>parms</td>
<td>Object</td>
<td>Optional. JSON object that contains additional translation parameters.</td>
</tr>
<tr>
<td>parms.translator</td>
<td>String</td>
<td>Translation service to use to detect the language of a string. Translation services are configured under the Translator Configuration menu and located in the Translator Configuration [sn_dt_translator_configuration] table.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Possible values - not case-sensitive:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Google</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Microsoft</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- IBM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- &lt;custom&gt;</td>
</tr>
</tbody>
</table>

**Note:** To use custom translation services you must first configure the translation service in your instance. For details, see Integrate with a translation service provider.


## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alternatives</td>
<td>Array of objects that describe other languages that might also be a match. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"alternatives": [
  {
    "code": "String",
    "confidence": "String",
    "name": "String"
  }
]
```

<table>
<thead>
<tr>
<th>alternatives.code</th>
<th>Language code of the alternative language. Data type: String</th>
</tr>
</thead>
<tbody>
<tr>
<td>alternatives.confidence</td>
<td>Float value indicating the confidence level of the alternative language. Value is between zero and one. The lower the value, the lower the confidence level.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alternatives.name</td>
<td>Language code of the alternative language. Data type: String</td>
</tr>
<tr>
<td>detectedLanguage</td>
<td>Description of the detected language. Data type: Object</td>
</tr>
<tr>
<td>detectedLanguage.code</td>
<td>Language code of the detected language. Data type: String</td>
</tr>
<tr>
<td>detectedLanguage.confidence</td>
<td>Float value indicating the confidence level of the alternative language. Value is between zero and one. The lower the value, the lower the confidence level. Data type: String</td>
</tr>
<tr>
<td>detectedLanguage.name</td>
<td>Language code of the detected language. Data type: String</td>
</tr>
<tr>
<td>translator</td>
<td>Translation service used to detect the language. Data type: String</td>
</tr>
</tbody>
</table>

Error messages

The following are error messages that the method may return and indications as to the error's root cause.

- Text ("text" field) is missing or invalid. (40000): The text to detect the language is either missing or not a string.
- Dynamic Translation plugin is not installed. (40001): The Dynamic Translation API was invoked without activating the com.glide.dynamic_translation plugin. For
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>information on activating this plugin, see <strong>Dynamic translation overview</strong>.</td>
</tr>
<tr>
<td></td>
<td><strong>Translator</strong> (“translator” field) is invalid. (40003): The passed in <strong>translator</strong> parameter is not a string.</td>
</tr>
<tr>
<td></td>
<td><strong>&lt;translator&gt;</strong> translator is not configured. (40004): The specified translation service is not configured in the Translator Configuration. For information on creating/modifying a translator configuration, see <strong>Create a translator configuration</strong>.</td>
</tr>
<tr>
<td></td>
<td><strong>&lt;translator&gt;</strong> translator is inactive. (40005): The specified translation service is not set to <strong>Active</strong> in the Translator Configuration. For information on creating/modifying a translator configuration, see <strong>Create a translator configuration</strong>.</td>
</tr>
<tr>
<td></td>
<td>Additional parameters are invalid. (40006): The additional parameters that were passed are not an object.</td>
</tr>
<tr>
<td></td>
<td>Maximum time limit has been exceeded. (40009): The operation took longer than the defined timeout value specified in the Translation Configuration. Default: 40 seconds</td>
</tr>
<tr>
<td></td>
<td><strong>Default translator is not configured for detection.</strong> (40011): The default translation service has not been specified for language detection in the Translator Configuration. For information on creating/modifying a translator configuration, see <strong>Create a translator configuration</strong>.</td>
</tr>
</tbody>
</table>
|      | **<translator>** translator is not configured for detection. (40013): The specified translation service is not configured for language detection in the Translator Configuration. For information on creating/
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>modifying a translator configuration, see Create a translator configuration.</td>
</tr>
<tr>
<td></td>
<td>• Unknown error occurred. (40051): Default error thrown when the error doesn't fall in to any other category.</td>
</tr>
<tr>
<td></td>
<td>• Text (&quot;text&quot; field) has exceeded its maximum length. (40052): The text that was passed in for language detections exceeds the maximum length supported by the corresponding translation service.</td>
</tr>
<tr>
<td></td>
<td>• Request is not authorized because credentials are missing or invalid (40055): The credentials configured for the translation service in Connections &amp; Credentials are not valid. For information on connections and credentials, see Dynamic translation overview.</td>
</tr>
</tbody>
</table>

This example shows code that detects a string in English using IBM's translation service in a standard client script.

```
var detectedResponse = DynamicTranslation.getDetectedLanguage('Please detect the language of this text', {'translator':'IBM'}).then(function(res) {console.log(res); }, function(res) {console.log(res); } );
```

Output:

```
detectedResponse {
    detectedLanguage: {
        "code": "en", "confidence": "1", "name": "en" }
alternatives:
    [ {
        "code": "vi", "confidence": "0.86", "name": "vi" },
        { "code": "id", "confidence": "0.86", "name": "id" }
    ]
}
```

This example shows a client script that throws an error when an invalid translation service is passed in.
var detectedResponse = DynamicTranslation.getDetectedLanguage('Please detect the language of this text',{"translator":123}).then(function(res) {console.log(res); }, function(res) {console.log(res); });

Output:

{"code":40003,"message":"Translator ("translator" field) is invalid"}

This example shows code that detects a string in English using IBM's translation service in a Service Portal widget client script. Note that the name of the class is dynamicTranslation not DynamicTranslation.

var detectedResponse = dynamicTranslation.getDetectedLanguage('Please detect the language of this text',{"translator':'IBM'}).then(function(res) {console.log(res); }, function(res) {console.log(res); });

Output:

detectedResponse {
  detectedLanguage:
    { "code": "en", "confidence": "1", "name": "en" }
  alternatives:
    [
      { "code": "vi", "confidence": "0.86", "name": "vi" },
      { "code": "id", "confidence": "0.86", "name": "id" }
    ]
}

This example shows a Service Portal widget client script that throws an error when an invalid translation service is passed in.

var detectedResponse = dynamicTranslation.getDetectedLanguage('Please detect the language of this text',{"translator":123}).then(function(res) {console.log(res); }, function(res) {console.log(res); });

Output:

{"code":40003,"message":"Translator ("translator" field) is invalid"}

DynamicTranslation - getDetectedLanguages(Array texts, Object parms)
Detects the languages of the passed in text strings.

If you pass in a translator, the method uses that translation service to detect the source language. Otherwise, the detection is performed by the default translation service. Ensure that the text strings that you provide contain enough verbiage to enable proper language detection.
In addition to the detected language, the response contains a confidence level of the detection, along with other possible language alternatives. If a translator is not passed in, the method also returns the default translation service used to detect the language.

When calling this method from a portal client script, use the class name `dynamicTranslation`; such as `dynamicTranslation.getTranslations()`. When calling it from a platform client script, use the class name `DynamicTranslation`; such as `DynamicTranslation.getTranslations()`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parms</td>
<td>Object</td>
<td>Optional. JSON object that contains additional translation parameters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;parms&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;translator&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td>texts</td>
<td>Array</td>
<td>List of text strings to use to detect the language(s).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| parms.translator | String | Translation service to use to detect the language of a string. Translation services are configured under the Translator Configuration menu and located in the Translator Configuration [sn_dt_translator_configuration] table. Possible values - not case-sensitive:  
  - Google  
  - Microsoft  
  - IBM  
  - <custom>  

**Note:** To use custom translation services you must first configure the translation service in your instance. For details, see [Integrate with a translation service provider](#).

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>detections</td>
<td>Language detection of text strings. Data type: Object</td>
</tr>
<tr>
<td>detections.alternatives</td>
<td>Array of objects that describe other languages that might also be a match. Data type: Array</td>
</tr>
<tr>
<td>detections.alternatives.code</td>
<td>Language code of the alternative language. Data type: String</td>
</tr>
<tr>
<td>detections.alternatives.confidence</td>
<td>Float value indicating the confidence level of the alternative language. Value is between zero and one. The lower the value, the lower the confidence level. Data type: String</td>
</tr>
<tr>
<td>detections.alternatives.name</td>
<td>Language code of the alternative language. Data type: String</td>
</tr>
<tr>
<td>detections.detectedLanguage</td>
<td>Description of the detected language. Data type: Object</td>
</tr>
<tr>
<td>detections.detectedLanguage.code</td>
<td>Language code of the detected language. Data type: String</td>
</tr>
<tr>
<td>detections.detectedLanguage.confidence</td>
<td>Float value indicating the confidence level of the detected language. Data type: String</td>
</tr>
<tr>
<td>detections.detectedLanguage.name</td>
<td>Language code of the detected language. Data type: String</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>detections.detectedLanguage.code</td>
<td>Language code of the detected language.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>detections.detectedLanguage.confidence</td>
<td>Float value indicating the confidence level of the alternative language. Value is between zero and one. The lower the value, the lower the confidence level.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>detections.detectedLanguage.name</td>
<td>Language code of the detected language.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>detections.isError</td>
<td>Flag that indicates whether the detection of language for the text resulted in an error.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Error encountered.</td>
</tr>
<tr>
<td></td>
<td>• false: Language detection was successful.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>status</td>
<td>Status of the response to the method call.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Error</td>
</tr>
<tr>
<td></td>
<td>• Partial</td>
</tr>
<tr>
<td></td>
<td>• Success</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>translator</td>
<td>Translation service used to detect the language.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Error messages</td>
<td>The following are error messages that the method may return and indications as to the error’s root cause.</td>
</tr>
<tr>
<td></td>
<td>• Text (&quot;text&quot; field) is missing or invalid. (40000): The text to detect the language is either missing or not a string.</td>
</tr>
<tr>
<td></td>
<td>• Dynamic Translation plugin is not installed. (40001): The Dynamic Translation API was invoked without activating the com.glide.dynamic_translation plugin.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>For information on activating this plugin, see Dynamic translation overview.</td>
</tr>
<tr>
<td></td>
<td>• Translator (&quot;translator&quot; field) is invalid. (40003): The passed in <code>translator</code> parameter is not a string.</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;translator&gt;</code> translator is not configured. (40004): The specified translation service is not configured in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;translator&gt;</code> translator is inactive. (40005): The specified translation service is not set to <strong>Active</strong> in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
</tr>
<tr>
<td></td>
<td>• Additional parameters are invalid. (40006): The additional parameters that were passed are not an object.</td>
</tr>
<tr>
<td></td>
<td>• Maximum time limit has been exceeded. (40009): The operation took longer than the defined timeout value specified in the Translation Configuration. Default: 40 seconds</td>
</tr>
<tr>
<td></td>
<td>• Request failed with multiple errors. (40010): Multiple errors occurred in the language detection call. For more information, refer to the response for each individual text string.</td>
</tr>
<tr>
<td></td>
<td>• Default translator is not configured for detection. (40011): The default translation service has not been specified for language detection in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;translator&gt;</code> translator is not configured for detection. (40013): The specified translation service is not configured for language detection in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Translator configuration version is invalid. Migrate to v3. (40014): The associated version of the Translator Configuration for the specified translation service does not support the specified text translation method. For more information, see Migrate to version v3 of a translator configuration.</td>
<td></td>
</tr>
<tr>
<td>• Unknown error occurred. (40051): Default error thrown when the error doesn’t fall in to any other category.</td>
<td></td>
</tr>
<tr>
<td>• Text (&quot;text&quot; field) has exceeded its maximum length. (40052): The text that was passed in for language detections exceeds the maximum length supported by the corresponding translation service.</td>
<td></td>
</tr>
<tr>
<td>• Request is not authorized because credentials are missing or invalid (40055): The credentials configured for the translation service in Connections &amp; Credentials are not valid. For information on connections and credentials, see Dynamic translation overview.</td>
<td></td>
</tr>
</tbody>
</table>

This example shows code from a portal client script that detects English as the language of the passed-in strings using the Microsoft translation service.

```javascript
var detectedResponse = dynamicTranslation.getDetectedLanguages(["First text string language to detect", "Second text string language to detect"], {"translator": "Microsoft"}).then(function(res) {console.log(res);}, function(res) {console.log(res);});
gs.info(JSON.stringify(detectedResponse));
```

Output

```javascript
{
 "translator":"Microsoft",
 "status":"Success",
 "detections": [
 { 
 "isError":false,
 "detectedLanguage":{"name":"en", "code":"en", "confidence":"1"},
 "alternatives":[
 {"name":"id", "code":"id", "confidence":"0.83"},
 {"name":"ms", "code":"ms", "confidence":"0.83"}
 ]
 },
}
```
This example shows code in a portal client script that returns a Partial status when two text strings are passed in and one of them is invalid. To use this code example in a platform client script, change `dynamicTranslation.getDetectedLanguages` to `DynamicTranslation.getDetectedLanguages`.

```javascript
var detectedResponse = dynamicTranslation.getDetectedLanguages(
    ["First text string language to detect", ""],
    {"translator": "Microsoft"}
).then(function(res) {
    console.log(res);
},
function(res) {
    console.log(res);
});
gs.info(JSON.stringify(detectedResponse));
```

Output

```javascript
{
    "translator": "Microsoft",
    "status": "Partial",
    "detections": [
        {
            "isError": false,
            "detectedLanguage": {"name": "en", "code": "en", "confidence": "1"},
            "alternatives": [
                {"name": "fr", "code": "fr", "confidence": "0.83"},
                {"name": "id", "code": "id", "confidence": "0.83"}
            ]
        },
        {
            "isError": true,
            "code": "40000",
            "message": "Text is missing or invalid"
        }
    ]
}
```
This example shows code from a portal client script that throws an error when an invalid translation service is passed in. To use this code example for a platform client script, change `dynamicTranslation.getDetectedLanguages` to `DynamicTranslation.getDetectedLanguages`.

```javascript
var detectedResponse = dynamicTranslation.getDetectedLanguages(["First text string language to detect", "Second text string language to detect"], {"translator": "123"}).then(function(res) {console.log(res); }, function(res) {console.log(res); });
gs.info(JSON.stringify(detectedResponse));
```

Output

```json
{"code":40003,"message":"Translator ("translator" field) is invalid","status":"Error"}
```

**DynamicTranslation - getTranslation(String textToTranslate, Object parms)**

Translates the passed in text to one or more languages.

The method uses translation services, such as Microsoft Azure Translator Service, IBM Watson Translator Service, and Google Cloud Translator Service, to perform the translation. If you do not pass in translation parameters, the method uses the system default.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>textToTranslate</td>
<td>String</td>
<td>Text to translate.</td>
</tr>
<tr>
<td>parms</td>
<td>Object</td>
<td>Optional. JSON object that contains additional translation parameters.</td>
</tr>
</tbody>
</table>

  "parms": {
  "additionalParameters":
  {Object},
  "sourceLanguage": "String",
  "targetLanguages": [Array],
  "translator": "String"
  }

| parms.additionalParameters | Object | Optional. Array of JSON objects. Each object contains key-value pairs that provide additional information for performing the translation. |
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>additionalParameters</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>parameterName</strong></td>
<td>String</td>
<td>Optional. Key name. Valid values:</td>
</tr>
<tr>
<td><strong>parameterValue</strong></td>
<td>String</td>
<td>Optional. Value of the associated key. Valid values:</td>
</tr>
<tr>
<td><strong>sourceLanguage</strong></td>
<td>String</td>
<td>Optional. Language code of the source text. Default: Translation service detects the source language.</td>
</tr>
<tr>
<td><strong>targetLanguages</strong></td>
<td>Array</td>
<td>Optional. List of language codes to use to translate the text. Default: User preferred language.</td>
</tr>
<tr>
<td><strong>translator</strong></td>
<td>String</td>
<td>Optional. Translation service to use to translate the text (not case-sensitive). Valid values:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>• Google</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Microsoft</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IBM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &lt;custom&gt;</td>
</tr>
</tbody>
</table>

**Note:** To use custom translation services you must first configure the translation service in your instance. For details, see Integrate with a translation service provider.


Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>detectedLanguage</td>
<td>Description of the detected language. Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;detectedLanguage&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;code&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>detectedLanguage.code</td>
<td>Language code of the detected language. Data type: String</td>
</tr>
<tr>
<td>detectedLanguage.name</td>
<td>Language code of the detected language. Data type: String</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>translations</td>
<td>Array of objects that describe the language translations.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Array</td>
</tr>
<tr>
<td></td>
<td><strong>translations</strong>: [</td>
</tr>
<tr>
<td></td>
<td>- <strong>targetLanguage</strong>: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>- <strong>translatedText</strong>: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>translations.targetLanguage</td>
<td>Language code to which the source text was translated.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> String</td>
</tr>
<tr>
<td>translations.translatedText</td>
<td>Translated text.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> String</td>
</tr>
<tr>
<td>translator</td>
<td>Translation service used to detect the language.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> String</td>
</tr>
<tr>
<td>Error messages</td>
<td>The following are error messages that the method may return and indications as to their root cause.</td>
</tr>
<tr>
<td></td>
<td>- Text (&quot;text&quot; field) is missing or invalid. (40000): The text to translate is either missing or not a string.</td>
</tr>
<tr>
<td></td>
<td>- Dynamic Translation plugin is not installed. (40001): The Dynamic Translation API was invoked without activating the</td>
</tr>
<tr>
<td></td>
<td>com.glide.dynamic_translation plugin. For information on activating this plugin, see Dynamic translation overview.</td>
</tr>
<tr>
<td></td>
<td>- Default translator is not configured for translation. (40002): No translation service is selected as the default translation service in the Translator Configurations. For information on creating/modifying a translator</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>configuration, see Create a translator configuration.</td>
</tr>
<tr>
<td></td>
<td>• Translator (&quot;translator&quot; field) is invalid. (40003): The passed in <code>translator</code> parameter is not a string.</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;translator&gt;</code> translator is not configured. (40004): The specified translation service is not configured in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;translator&gt;</code> translator is inactive. (40005): The specified translation service is not set to <code>Active</code> in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
</tr>
<tr>
<td></td>
<td>• Additional parameters are invalid. (40006): The additional parameters that were passed are not an object.</td>
</tr>
<tr>
<td></td>
<td>• Target languages (&quot;targetLanguages&quot; field) are invalid. (40007): The <code>targetLanguages</code> parameter is passed in the call but is not valid for one of the following reasons:  ◦ Value is not an array  ◦ Array is empty  ◦ One or multiple of the entries is not a string</td>
</tr>
<tr>
<td></td>
<td>• Source language (&quot;sourceLanguage&quot; field) is invalid. (40008): The <code>sourceLanguage</code> parameter is passed in the call but the value is not a String.</td>
</tr>
<tr>
<td></td>
<td>• Maximum time limit has been exceeded. (40009): The operation took longer than the defined timeout value specified in the Translation Configuration. Default: 40 seconds</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) <code>&lt;translator&gt;</code> translator is not configured for translation. (40012): The specified translation service is not configured for text translation in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
<td></td>
</tr>
<tr>
<td>2) Unknown error occurred. (40051): Default error thrown when the error doesn’t fall in to any other category.</td>
<td></td>
</tr>
<tr>
<td>3) Text (“text” field) has exceeded its maximum length. (40052): The text that was passed in to translate exceeds the maximum length supported by the corresponding translation service.</td>
<td></td>
</tr>
<tr>
<td>4) Source language is invalid. (40053): The passed in <code>sourceLanguage</code> parameter contains a language code that is not supported by the corresponding translation service.</td>
<td></td>
</tr>
<tr>
<td>5) Target language is invalid. (40054): One or more of the language codes passed in the <code>targetLanguages</code> parameter is not supported by the corresponding translation service.</td>
<td></td>
</tr>
<tr>
<td>6) Request is not authorized because credentials are missing or invalid (40055): The credentials configured for the translation service in Connections &amp; Credentials are not valid. For information on connections and credentials, see Dynamic translation overview.</td>
<td></td>
</tr>
<tr>
<td>7) Text cannot be translated to target languages. (40056): The specified translation service is not able to translate the passed in text into the specified target languages.</td>
<td></td>
</tr>
</tbody>
</table>

This example shows the translation of plain text content from English (detected) into French and Italian using Microsoft's translation service in a standard client script.
DynamicTranslation.getTranslation("Translate this text using platform from client", {  "targetLanguages": ["fr", "it"],  "additionalParameters": [{    "parameterName": "texttype",    "parameterValue": "plain"  }],  "translator": "Microsoft"}).then(function(res){console.log(res);}, function(res){console.log(res);});

Response:

{"translations":[
  {
    "targetLanguage":"it",
    "translatedText":"Tradurre questo testo utilizzando la piattaforma dal client"
  },
  {
    "targetLanguage":"fr",
    "translatedText":"Traduire ce texte en utilisant la plate-forme du client"
  }
],
"translator":"Microsoft",
"detectedLanguage":{"code":"en","name":"en"}
}

This example shows a client script that throws an error when an invalid target language is passed in.

DynamicTranslation.getTranslation("Translate this text using platform from client", {  "targetLanguages": ["123"],  "additionalParameters": [{    "parameterName": "texttype",    "parameterValue": "plain"  }],  "translator": "Microsoft"}).then(function(res){console.log(res);}, function(res){console.log(res);});

Response:

{"code":"40054","message":"Target language is invalid"}

This example shows the translation of plain text content from English (detected) into French and Italian using Microsoft's translation service in a Service Portal widget client script. Note that the name of the class is dynamicTranslation not DynamicTranslation.
dynamicTranslation.getTranslation("Translate this text using platform from client", {
    "targetLanguages": ["fr", "it"],
    "additionalParameters": [{
        "parameterName": "texttype",
        "parameterValue": "plain"
    }],
    "translator": "Microsoft"
}).then(function(res){console.log(res);}, function(res){console.log(res);});

Response:

{"translations": [  
   {  
        "targetLanguage": "it",
        "translatedText": "Tradurre questo testo utilizzando la piattaforma dal client"
   },
   {  
        "targetLanguage": "fr",
        "translatedText": "Traduire ce texte en utilisant la plate-forme du client"
   }
],
"translator": "Microsoft",
"detectedLanguage": {"code": "en", "name": "en"}
}

This example shows a Service Portal widget client script that throws an error when an invalid target language is passed in

dynamicTranslation.getTranslation("Translate this text using platform from client", {
    "targetLanguages": [123],
    "additionalParameters": [{
        "parameterName": "texttype",
        "parameterValue": "plain"
    }],
    "translator": "Microsoft"
}).then(function(res){console.log(res);}, function(res){console.log(res);});

Response:

{"code": "40054", "message": "Target language is invalid"}

DynamicTranslation - getTranslations(Array texts, Object parms)
Translates the passed in text strings to one or more languages.
The method uses translation services, such as Microsoft Azure Translator Service, IBM Watson Translator Service, and Google Cloud Translator Service, to perform the translation. If you do not pass in translation parameters, the method uses the system default.

When calling this method from a portal client script, use the class name `dynamicTranslation`; such as `dynamicTranslation.getTranslations()`. When calling it from a platform client script, use the class name `DynamicTranslation`; such as `DynamicTranslation.getTranslations()`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>texts</td>
<td>Array</td>
<td>List of text stings to translate.</td>
</tr>
<tr>
<td>parms</td>
<td>Object</td>
<td>Optional. JSON object that contains additional translation parameters.</td>
</tr>
<tr>
<td>parms.additionalParameters</td>
<td>Object</td>
<td>Optional. Array of JSON objects. Each object contains key-value pairs that provide additional information for performing the translation.</td>
</tr>
</tbody>
</table>
| parms.additionalParameters.parameterName | String  | Optional. Key name. Valid values:  
- textype: Type of text to translate. For Microsoft Azure Translator Service only. |
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>parms.additionalParameters.parameterValue</code></td>
<td>String</td>
<td>Optional. Value of the associated key.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• plain: Standard text string</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• html: HTML text string</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: plain</td>
</tr>
<tr>
<td><code>parms.sourceLanguage</code></td>
<td>String</td>
<td>Optional. Language code of the source text.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Translation service detects the source language.</td>
</tr>
<tr>
<td><code>parms.targetLanguages</code></td>
<td>Array</td>
<td>Optional. List of language codes to use to translate the text. The method returns translated text for each language code.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: User preferred language.</td>
</tr>
<tr>
<td><code>parms.translator</code></td>
<td>String</td>
<td>Optional. Translation service to use to translate the text (not case-sensitive).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Google</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Microsoft</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IBM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &lt;custom&gt;</td>
</tr>
</tbody>
</table>
Parameters (continued)
Name

Type

Description

Note: To use custom
translation services
you must first configure
the translation service
in your instance. For
details, see Integrate
with a translation service
provider.
Default: Translation
service configured in the
Translator Configuration
[sn_dt_translator_configuration]
table.
Returns
Type

Description

status

Status of the response to the method
call.
Possible values:
• Error
• Partial
• Success
Data type: String

translations

Array of objects that describe the
language translations.
Data type: Array
translations": [
{
"isError": Boolean;
"detectedLanguage": {Object},
"textTranslations": [Array]
}
]

1743

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| translations.isError             | Flag that indicates whether the translation of the text resulted in an error. Valid values:  
• true: Error encountered.  
• false: Text translation was successful.  
Data type: Boolean                |
| translations.detectedLanguage    | Description of the detected language.                                       |
|                                  | Data type: Object                                                           |
|                                  | "detectedLanguage": {                                                      |
|                                  |   "code": "String",                                                       |
|                                  |   "name": "String"                                                       |
|                                  | }                                                                          |
| translations.detectedLanguage.code | Language code of the detected language.                                    |
|                                  | Data type: String                                                           |
| translations.detectedLanguage.name | Language code of the detected language.                                    |
|                                  | Data type: String                                                           |
| textTranslations                 | Array of objects. Description of the language used to translate the text string. |
|                                  | "textTranslation": {                                                      |
|                                  |   "targetLanguage": "String",                                             |
|                                  |   "translatedText": "String"                                              |
|                                  | }                                                                          |
| textTranslations.targetLanguage  | Language code to which the source text was translated.                     |
|                                  | Data type: String                                                           |
| textTranslations.translatedText  | Translated text.                                                            |
|                                  | Data type: String                                                           |
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>translator</td>
<td>Translation service used to translate the text. Data type: String</td>
</tr>
</tbody>
</table>

**Error messages**

The following are error messages that the method may return and indications as to their root cause.

- **Text ("text" field) is missing or invalid. (40000):** The text to translate is either missing or not a string.

- **Dynamic Translation plugin is not installed. (40001):** The Dynamic Translation API was invoked without activating the com.glide.dynamic_translation plugin. For information on activating this plugin, see [Dynamic translation overview](#).

- **Default translator is not configured for translation. (40002):** No translation service is selected as the default translation service in the Translator Configurations. For information on creating/modifying a translator configuration, see [Create a translator configuration](#).

- **Translator ("translator" field) is invalid. (40003):** The passed in `translator` parameter is not a string.

- **<translator> translator is not configured. (40004):** The specified translation service is not configured in the Translator Configuration. For information on creating/modifying a translator configuration, see [Create a translator configuration](#).

- **<translator> translator is inactive. (40005):** The specified translation
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>service is not set to <strong>Active</strong> in the Translator Configuration. For information on creating/modifying a translator configuration, see <a href="#">Create a translator configuration</a>.</td>
<td></td>
</tr>
</tbody>
</table>

- Additional parameters are invalid. (40006): The additional parameters that were passed are not an object.

- Target languages ("targetLanguages" field) are invalid. (40007): The `targetLanguages` parameter is passed in the call but is not valid for one of the following reasons:
  - Value is not an array
  - Array is empty
  - One or multiple of the entries is not a string

- Source language ("sourceLanguage" field) is invalid. (40008): The `sourceLanguage` parameter is passed in the call but the value is not a String.

- Maximum time limit has been exceeded. (40009): The operation took longer than the defined timeout value specified in the Translation Configuration. Default: 40 seconds

- Request failed with multiple errors. (40010): Multiple errors occurred in the language detection call. For more information, refer to the response for each individual text string.

- `<translator>` translator is not configured for translation. (40012): The specified translation service is...
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>not configured for text translation in the Translator Configuration. For information on creating/modifyng a translator configuration, see Create a translator configuration.</td>
</tr>
<tr>
<td></td>
<td>Translator configuration version is invalid. Migrate to v3. (40014): The associated version of the Translator Configuration for the specified translation service does not support the specified text translation method. For more information, see Migrate to version v3 of a translator configuration.</td>
</tr>
<tr>
<td></td>
<td>Unknown error occurred. (40051): Default error thrown when the error doesn't fall in to any other category.</td>
</tr>
<tr>
<td></td>
<td>Text (&quot;text&quot; field) has exceeded its maximum length. (40052): The text that was passed in to translate exceeds the maximum length supported by the corresponding translation service.</td>
</tr>
<tr>
<td></td>
<td>Source language is invalid. (40053): The passed in sourceLanguage parameter contains a language code that is not supported by the corresponding translation service.</td>
</tr>
<tr>
<td></td>
<td>Target language is invalid. (40054): One or more of the language codes passed in the targetLanguages parameter is not supported by the corresponding translation service.</td>
</tr>
<tr>
<td></td>
<td>Request is not authorized because credentials are missing or invalid (40055): The credentials configured for the translation service in Connections &amp; Credentials are not</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>valid. For information on connections and credentials, see Dynamic translation overview.</td>
</tr>
<tr>
<td></td>
<td>• Text cannot be translated to target languages. (40056): The specified translation service is not able to translate the passed in text into the specified target languages.</td>
</tr>
</tbody>
</table>

This example shows code in a portal client script that translates a list of text strings into French and Italian using the Microsoft translation service.

dynamicTranslation.getTranslations(
	"Translate first text using portal from client",
	"Translate second text using portal from client"), {
	"targetLanguages": ["fr", "it"],
	"additionalParameters": [
		"parameterName": "texttype",
		"parameterValue": "plain"
	],
	"translator": "Microsoft"
}).then(function(res){console.log(res);}, function(res){console.log(res);});

Response:

{
	"translations": [
	{
		"isError":false,
		"textTranslations": [
		{
			"targetLanguage": "it",
			"translatedText": "Tradurre il primo testo utilizzando il portale dal client"
		},
		{
			"targetLanguage": "fr",
			"translatedText": "Traduire le premier texte à l’aide du portail à partir du client"
		}
	],
	"detectedLanguage": {"name": "en", "code": "en"}
	}
}
This example shows a portal client script that returns a Partial status when one of the two passed in text strings is invalid. To use this code example for a platform client script, change `dynamicTranslation.getTranslations` to `DynamicTranslation.getTranslations`.

```javascript
dynamicTranslation.getTranslations(['Translate first text using portal from client', ''], {
  'targetLanguages': ['fr', 'it'],
  'additionalParameters': [{
    'parameterName': 'texttype',
    'ParameterValue': 'plain'
  }],
  'translator': 'Microsoft'
}).then(function(res){console.log(res);}, function(res){console.log(res);});
```

Response:

```json
{
  "translations":{
    {
      "isError":false,
      "textTranslations":{
        {
          "targetLanguage": "it",
          "translatedText": "Tradurre il primo testo utilizzando il portale dal client"
        }
      }
    }
  }
}
This example shows a portal client script that throws an error when an invalid translation service is passed in. To use this code example for a platform client script, change `dynamicTranslation.getTranslations` to `DynamicTranslation.getTranslations`.

```javascript
dynamicTranslation.getTranslations(
  [
    "Translate first text using portal from client",
    "Translate second text using portal from client"
  ],
  {
    "targetLanguages": ["fr", "it"],
    "additionalParameters": [{
      "parameterName": "texttype",
      "parameterValue": "plain"
    }],
    "translator": 123
  }).then(function(res){console.log(res);}, function(res){console.log(res);});
```

Response:

```json
{"code":40003,"message":"Translator ("translator" field) is invalid","status":"Error"}
```

**DynamicTranslation - isEnabled(String translator)**

Determines whether the various methods in the `DynamicTranslation API` are enabled for a translation service.

If you pass in a specific translation service, the method checks the method activation for that translation service; otherwise the method checks the default translation service.
When calling this method from a portal client script, use the class name `dynamicTranslation`; such as `dynamicTranslation.isEnabled()`. When calling it from a platform client script, use the class name `DynamicTranslation`; such as `DynamicTranslation.isEnabled()`.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>translator</td>
<td>String</td>
<td>Optional. Translation service to use to verify whether the methods are active. Translation services are configured under the Translator Configuration menu. Possible values - not case-sensitive:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Google</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Microsoft</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IBM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &lt;custom&gt;</td>
</tr>
</tbody>
</table>

**Note:** To use custom translation services you must first configure the translation service in your instance. For details, see [Integrate with a translation service provider](#).

Default: Default translation service.

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>batchDetection</td>
<td>Flag that indicates whether the <code>getDetectedLanguages()</code> method is enabled. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Enabled</td>
</tr>
<tr>
<td></td>
<td>• false: Disabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>batchTranslation</td>
<td>Flag that indicates whether the <code>getTranslations()</code> method is enabled. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Enabled</td>
</tr>
<tr>
<td></td>
<td>• false: Disabled</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>detection</td>
<td>Flag that indicates whether the <code>getDetectedLanguage()</code> method is enabled. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Enabled</td>
</tr>
<tr>
<td></td>
<td>• false: Disabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>translation</td>
<td>Flag that indicates whether the <code>getTranslation()</code> method is enabled. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Enabled</td>
</tr>
<tr>
<td></td>
<td>• false: Disabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>Error messages</td>
<td>The following are error messages that the API may return and indications as to their root cause.</td>
</tr>
<tr>
<td></td>
<td>• Translator (&quot;translator&quot; field) is invalid. (40003): The passed in translator parameter is</td>
</tr>
<tr>
<td></td>
<td>not a string.</td>
</tr>
</tbody>
</table>

This example shows a client script that checks whether the `DynamicTranslation` methods are enabled for the Microsoft translation service. To use this code example for a platform client script, change `DynamicTranslation.getTranslations` to `dynamicTranslation.getTranslations`.

```javascript
DynamicTranslation.isEnabled('Microsoft').then(function(res){console.log(res);},
    function(res){console.log(res);});
```

Output:

```
{"detection":true,"batchTranslation":true,"batchDetection":true,"translation":true}
```

This example shows a client script that throws an error when an invalid translation service is passed in. To use this code example for a platform client script, change `DynamicTranslation.getTranslations` to `dynamicTranslation.getTranslations`.

```javascript
DynamicTranslation.isEnabled(123).then(function(res){console.log(res);},
    function(res){console.log(res);});
```
**DynamicTranslation - Scoped**

The DynamicTranslation API provides methods that translate text, in real time, into multiple languages using translation service providers.

In addition, you can use this API to detect the language of a specific string and check whether the DynamicTranslation methods are enabled for a translation service. Use this API to create a seamless localization experience for your user interface, enabling one interface to service multiple countries.

Currently this API supports three translation service providers: Microsoft Azure Translator Service, IBM Watson Translator Service, and Google Cloud Translator Service. You can also configure other translation services within your instance and then use the DynamicTranslation API to translate your text.

When using these methods in a server-side script, use the `sn_dt_api` namespace identifier. Before you are able to use this API, you must activate the DynamicTranslation (com.glide.dynamic_translation) plugin. For information on this plugin and additional information on Dynamic Translation, refer to [Dynamic translation overview](#).

**DynamicTranslation - getDetectedLanguage(String text, Object parms)**

Detects the language of the passed in text.

If you pass in a translator, the method uses that translation service to detect the source language. Otherwise, the detection is performed by the default translation service. Ensure that the text strings that you provide contain enough verbiage to enable proper language detection.

In addition to the detected language, the response contains a confidence level of the detection, along with other possible language alternatives. If a translator is not passed in, the method also returns the default translation service used to detect the language.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>Text to use to detect the language.</td>
</tr>
<tr>
<td>parms</td>
<td>Object</td>
<td>Optional. JSON object that contains additional translation parameters.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parms.translator</td>
<td>String</td>
<td>Optional. Translation service to use to translate the text (not case-sensitive). Valid values: Google, Microsoft, IBM, &lt;custom&gt;</td>
</tr>
</tbody>
</table>

**Note:** To use custom translation services you must first configure the translation service in your instance. For details, see [Integrate with a translation service provider](#).


### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alternatives</td>
<td>Array of objects that describe other languages that also may be a match.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;alternatives&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;code&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;confidence&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>alternatives.code</td>
<td>Language code of the alternative language. Data type: String</td>
</tr>
<tr>
<td>alternatives.confidence</td>
<td>Float value indicating the confidence level of the alternative language. Value is</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alternatives.name</td>
<td>Language code of the alternative language. Data type: String</td>
</tr>
<tr>
<td>detectedLanguage</td>
<td>Description of the detected language. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;code&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;confidence&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>detectedLanguage.code</td>
<td>Language code of the detected language. Data type: String</td>
</tr>
<tr>
<td>detectedLanguage.confidence</td>
<td>Float value indicating the confidence level of the alternative language. Value is between zero and one. The lower the value, the lower the confidence level. Data type: String</td>
</tr>
<tr>
<td>detectedLanguage.name</td>
<td>Language code of the detected language. Data type: String</td>
</tr>
<tr>
<td>translator</td>
<td>Translation service used to detect the language. Data type: String</td>
</tr>
</tbody>
</table>

**Error messages**

The following are error messages that the method may return and indications as to the error's root cause.

- Text ("text" field) is missing or invalid. (40000): The text to translate is either missing or not a string.
- Dynamic Translation plugin is not installed. (40001): The Dynamic Translation API
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>was invoked without activating the com.glide.dynamic_translation plugin. For information on activating this plugin, see Dynamic translation overview.</td>
</tr>
</tbody>
</table>

- **Translator** ("translator" field) is invalid.  
  (40003): The passed in *translator* parameter is not a string.

- `<translator>` translator is not configured.  
  (40004): The specified translation service is not configured in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.

- `<translator>` translator is inactive.  
  (40005): The specified translation service is not set to **Active** in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.

- Additional parameters are invalid.  
  (40006): The additional parameters that were passed are not an object.

- Maximum time limit has been exceeded.  
  (40009): The operation took longer than the defined timeout value specified in the Translation Configuration. Default: 40 seconds

- Default translator is not configured for detection.  
  (40011): The default translation service has not been specified for language detection in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.

- `<translator>` translator is not configured for detection.  
  (40013): The specified translation service is not configured for detection.
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language detection</td>
<td>language detection in the Translator Configuration. For information on creating/</td>
</tr>
<tr>
<td></td>
<td>modifying a translator configuration, see Create a translator configuration.</td>
</tr>
<tr>
<td>Unknown error occurred</td>
<td>• Unknown error occurred. (40051): Default error thrown when the error doesn’t fall in to any other category.</td>
</tr>
<tr>
<td>Text (<strong>text</strong> field) has exceeded its maximum length.</td>
<td>• Text (<strong>text</strong> field) has exceeded its maximum length. (40052): The text that was passed in to translate exceeds the maximum length supported by the corresponding translation service.</td>
</tr>
<tr>
<td>Request is not authorized because credentials are missing or invalid</td>
<td>• Request is not authorized because credentials are missing or invalid (40055): The credentials configured for the translation service in Connections &amp; Credentials are not valid. For information on connections and credentials, see Dynamic translation overview.</td>
</tr>
</tbody>
</table>

This example shows code in a server-side script that detects a string in English using IBM’s translation service.

```javascript
var detectedResponse = sn_dt_api.DynamicTranslation.getDetectedLanguage('Please detect the language of this text', {"translator":"IBM");
gs.info(JSON.stringify(detectedResponse));
```

**Output:**

```json
{
  detectedLanguage:
    { "code": "en", "confidence": "1", "name": "en" },
  alternatives:
    {
      { "code": "vi", "confidence": "0.86", "name": "vi" },
      { "code": "id", "confidence": "0.86", "name": "id" }
    }
}
```
This example shows a server script that throws an error when an invalid translation service is passed in.

```
var detectedResponse = sn_dt_api.DynamicTranslation.getDetectedLanguage('Please detect the language of this text', {"translator":123});
gs.info(JSON.stringify(detectedResponse));
```

Output:

```
{"code":"40003","message":"Translator ("translator" field) is invalid"}
```

**DynamicTranslation - getDetectedLanguages(Array texts, Object parms)**

Detects the languages of the passed in text strings.

If you pass in a translator, the method uses that translation service to detect the source language. Otherwise, the detection is performed by the default translation service. Ensure that the text strings that you provide contain enough verbiage to enable proper language detection.

In addition to the detected language, the response contains a confidence level of the detection, along with other possible language alternatives. If a translator is not passed in, the method also returns the default translation service used to detect the language.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>texts</td>
<td>Array</td>
<td>List of text strings to use to detect the language(s).</td>
</tr>
<tr>
<td>parms</td>
<td>Object</td>
<td>Optional. JSON object that contains additional translation parameters.</td>
</tr>
</tbody>
</table>
| parms.translator   | String | Translation service to use to detect the language of a string. Translation services are configured under the Translator Configuration menu and located in the Translator Configuration [sn_dt_translator_configuration] table. Possible values - not case-sensitive:  
  - Google  
  - Microsoft |

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>- IBM&lt;br&gt;- &lt;custom&gt;</td>
</tr>
</tbody>
</table>

**Note:** To use custom translation services you must first configure the translation service in your instance. For details, see [Integrate with a translation service provider](#).


### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>detections</td>
<td>Language detection of text strings.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;detections&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;alternatives&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;detectedLanguage&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;isError&quot;: Boolean</td>
</tr>
<tr>
<td>detections.alternatives</td>
<td>Array of objects that describe other languages that might also be a match.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;alternatives&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;code&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;confidence&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>detections.alternatives.code</td>
<td>Language code of the alternative language.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>detections.alternatives.confidence</td>
<td>Float value indicating the confidence level of the alternative language. Value is between zero and one. The lower the value, the lower the confidence level. Data type: String</td>
</tr>
<tr>
<td>detections.alternatives.name</td>
<td>Language code of the alternative language. Data type: String</td>
</tr>
<tr>
<td>detections.detectedLanguage</td>
<td>Language code of the detected language. Data type: Object</td>
</tr>
<tr>
<td>detections.detectedLanguage.code</td>
<td>Language code of the detected language. Data type: String</td>
</tr>
<tr>
<td>detections.detectedLanguage.confidence</td>
<td>Float value indicating the confidence level of the alternative language. Value is between zero and one. The lower the value, the lower the confidence level. Data type: String</td>
</tr>
<tr>
<td>detections.detectedLanguage.name</td>
<td>Language code of the detected language. Data type: String</td>
</tr>
</tbody>
</table>
| detections.isError | Flag that indicates whether the detection of language for the text resulted in an error. Valid values:  
- true: Error encountered.  
- false: Language detection was successful. Data type: Boolean |
| status | Status of the response to the method call. Possible values:
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Error</td>
<td></td>
</tr>
<tr>
<td>• Partial</td>
<td></td>
</tr>
<tr>
<td>• Success</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

<table>
<thead>
<tr>
<th>translator</th>
<th>Translation service used to detect the language.</th>
</tr>
</thead>
</table>

Data type: String

<table>
<thead>
<tr>
<th>Error messages</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>The following are error messages that the method may return and indications as to the error's root cause.</td>
<td></td>
</tr>
<tr>
<td>• Text (&quot;text&quot; field) is missing or invalid. (40000): The text to translate is either missing or not a string.</td>
<td></td>
</tr>
<tr>
<td>• Dynamic Translation plugin is not installed. (40001): The Dynamic Translation API was invoked without activating the com.glide.dynamic_translation plugin. For information on activating this plugin, see Dynamic translation overview.</td>
<td></td>
</tr>
<tr>
<td>• Translator (&quot;translator&quot; field) is invalid. (40003): The passed in translator parameter is not a string.</td>
<td></td>
</tr>
<tr>
<td>• &lt;translator&gt; translator is not configured. (40004): The specified translation service is not configured in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
<td></td>
</tr>
<tr>
<td>• &lt;translator&gt; translator is inactive. (40005): The specified translation service is not set to Active in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
<td></td>
</tr>
<tr>
<td>• Additional parameters are invalid. (40006): The additional parameters that were passed are not an object.</td>
<td></td>
</tr>
<tr>
<td>• Maximum time limit has been exceeded. (40009): The operation took longer than the defined timeout value specified in the Translation Configuration. Default: 40 seconds</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>•</td>
<td>Request failed with multiple errors. (40010): Multiple errors occurred in the language detection call. For more information, refer to the response for each individual text string.</td>
</tr>
<tr>
<td>•</td>
<td>Default translator is not configured for detection. (40011): The default translation service has not been specified for language detection in the Translator Configuration. For information on creating/modifying a translator configuration, see <a href="#">Create a translator configuration</a>.</td>
</tr>
<tr>
<td>•</td>
<td>&lt;translator&gt; translator is not configured for detection. (40013): The specified translation service is not configured for language detection in the Translator Configuration. For information on creating/modifying a translator configuration, see <a href="#">Create a translator configuration</a>.</td>
</tr>
<tr>
<td>•</td>
<td>Translator configuration version is invalid. Migrate to v3. (40014): The associated version of the Translator Configuration for the specified translation service does not support the specified text translation method. For more information, see <a href="#">Migrate to version v3 of a translator configuration</a>.</td>
</tr>
<tr>
<td>•</td>
<td>Unknown error occurred. (40051): Default error thrown when the error doesn't fall in to any other category.</td>
</tr>
<tr>
<td>•</td>
<td>Text (&quot;text&quot; field) has exceeded its maximum length. (40052): The text that was passed in to translate exceeds the maximum length supported by the corresponding translation service.</td>
</tr>
<tr>
<td>•</td>
<td>Request is not authorized because credentials are missing or invalid (40055): The credentials configured for the translation service in Connections &amp; Credentials are not valid. For information on connections and credentials, see <a href="#">Dynamic translation overview</a>.</td>
</tr>
</tbody>
</table>

This example shows code in a server script that detects English as the language of the passed-in strings using the Microsoft translation service.
var detectedResponse = sn_dt_api.DynamicTranslation.getDetectedLanguages(["First text string language to detect", "Second text string language to detect"], {"translator": "Microsoft"});
gs.info(JSON.stringify(detectedResponse));

Output

```
{
   "translator": "Microsoft",
   "status": "Success",
   "detections": [
   {
      "isError": false,
      "detectedLanguage": {"name": "en", "code": "en", "confidence": "1"},
      "alternatives": [
         {"name": "id", "code": "id", "confidence": "0.83"},
         {"name": "ms", "code": "ms", "confidence": "0.83"}
      ]
   },
   {
      "isError": false,
      "detectedLanguage": {"name": "en", "code": "en", "confidence": "1"},
      "alternatives": [
         {"name": "fr", "code": "fr", "confidence": "0.83"},
         {"name": "id", "code": "id", "confidence": "0.83"}
      ]
   }
   ]
}
```

This example shows code in a server script that returns a Partial status when two text strings are passed in and one of them is invalid.

var detectedResponse = sn_dt_api.DynamicTranslation.getDetectedLanguages(["First text string language to detect", ""], {"translator": "Microsoft"});
gs.info(JSON.stringify(detectedResponse));

Output

```
{
   "translator": "Microsoft",
   "status": "Partial",
   "detections": [
   {
      "isError": false,
      "detectedLanguage": {"name": "en", "code": "en", "confidence": "1"},
```
This example shows code in a server script that throws an error when an invalid translation service is passed in.

```javascript
var detectedResponse = sn_dt_api.DynamicTranslation.getDetectedLanguages(["First text string language to detect", "Second text string language to detect"], {"translator": "123");
gs.info(JSON.stringify(detectedResponse));
```

Output

```
{"code":"40003","message":"Translator ("translator" field) is invalid","status":"Error"
```

**DynamicTranslation - getTranslation(String textToTranslate, Object parms)**

Translates the passed in text to one or more languages.

The method uses translation services, such as Microsoft Azure Translator Service, IBM Watson Translator Service, and Google Cloud Translator Service, to perform the translation. If you do not pass in translation parameters, the method uses the system default.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>textToTranslate</td>
<td>String</td>
<td>Text to translate.</td>
</tr>
<tr>
<td>parms</td>
<td>Object</td>
<td>Optional. JSON object that contains additional translation parameters.</td>
</tr>
</tbody>
</table>
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>parms.additionalParameters</code></td>
<td>Array</td>
<td>Optional. Array of JSON objects. Each object contains key-value pairs that provide additional information for performing the translation.</td>
</tr>
<tr>
<td><code>parms.additionalParameters.parameterName</code></td>
<td>String</td>
<td>Optional. Key name.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>textype</code>: Type of text to translate. For Microsoft Azure Translator Service only.</td>
</tr>
<tr>
<td><code>parms.additionalParameters.parameterValue</code></td>
<td>String</td>
<td>Optional. Value of the associated key.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>plain</code>: Standard text string</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>html</code>: HTML text string</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: <code>plain</code></td>
</tr>
<tr>
<td><code>parms.sourceLanguage</code></td>
<td>String</td>
<td>Optional. Language code of the source text.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Translation service detects the source language.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>parms.targetLanguages</code></td>
<td>Array</td>
<td>Optional. List of language codes to use to translate the text. The method returns translated text for each language code.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: User preferred language.</td>
</tr>
<tr>
<td><code>parms.translator</code></td>
<td>String</td>
<td>Optional. Translation service to use to translate the text (not case-sensitive).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values: Google, Microsoft, IBM, &lt;custom&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> To use custom translation services you must first configure the translation service in your instance. For details, see Integrate with a translation service provider.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>detectedLanguage</td>
<td>Description of the detected language.</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| detectedLanguage.code | Language code of the detected language.  
Data type: String                                   |
| detectedLanguage.name   | Language code of the detected language.  
Data type: String                                    |
| translations          | Array of objects that describe the language translations.  
Data type: Array                                      |
| translations.targetLanguage | Language code to which the source text was translated.  
Data type: String                                     |
| translations.translatedText | Translated text.  
Data type: String                                      |
| translator            | Translation service used to detect the language.  
Data type: String                                     |
<p>| Error messages        | The following are error messages that the method may return and indications as to their root cause. |</p>
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Text (&quot;text&quot; field) is missing or invalid. (40000): The text to translate is either missing or not a string.</td>
<td></td>
</tr>
<tr>
<td>• Dynamic Translation plugin is not installed. (40001): The Dynamic Translation API was invoked without activating the com.glide.dynamic_translation plugin. For information on activating this plugin, see Dynamic translation overview.</td>
<td></td>
</tr>
<tr>
<td>• Default translator is not configured for translation. (40002): No translation service is selected as the default translation service in the Translator Configurations. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
<td></td>
</tr>
<tr>
<td>• Translator (&quot;translator&quot; field) is invalid. (40003): The passed in translator parameter is not a string.</td>
<td></td>
</tr>
<tr>
<td>• &lt;translator&gt; translator is not configured. (40004): The specified translation service is not configured in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
<td></td>
</tr>
<tr>
<td>• &lt;translator&gt; translator is inactive. (40005): The specified translation service is not set to Active in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
<td></td>
</tr>
<tr>
<td>• Additional parameters are invalid. (40006): The additional parameters that were passed are not an object.</td>
<td></td>
</tr>
<tr>
<td>• Target languages (&quot;targetLanguages&quot; field) are invalid. (40007): The targetLanguages parameter</td>
<td></td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>is passed in the call but is not valid for one of the following reasons:</td>
</tr>
<tr>
<td></td>
<td>◦ Value is not an array</td>
</tr>
<tr>
<td></td>
<td>◦ Array is empty</td>
</tr>
<tr>
<td></td>
<td>◦ One or multiple of the entries is not a string</td>
</tr>
<tr>
<td></td>
<td>• Source language (&quot;sourceLanguage&quot; field) is invalid. (40008): The <code>sourceLanguage</code> parameter is passed in the call but the value is not a String.</td>
</tr>
<tr>
<td></td>
<td>• Maximum time limit has been exceeded. (40009): The operation took longer than the defined timeout value specified in the Translation Configuration. Default: 40 seconds</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;translator&gt;</code> translator is not configured for translation. (40012): The specified translation service is not configured for text translation in the Translator Configuration. For information on creating/modifying a translator configuration, see <a href="#">Create a translator configuration</a>.</td>
</tr>
<tr>
<td></td>
<td>• Unknown error occurred. (40051): Default error thrown when the error doesn’t fall in to any other category.</td>
</tr>
<tr>
<td></td>
<td>• Text (&quot;text&quot; field) has exceeded its maximum length. (40052): The text that was passed in to translate exceeds the maximum length supported by the corresponding translation service.</td>
</tr>
<tr>
<td></td>
<td>• Source language is invalid. (40053): The passed in <code>sourceLanguage</code> parameter contains a language code that is not supported by the corresponding translation service.</td>
</tr>
<tr>
<td></td>
<td>• Target language is invalid. (40054): One or more of the language codes passed in the <code>targetLanguages</code> parameter is not supported by the corresponding translation service.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Request is not authorized because credentials are missing or invalid (40055): The credentials configured for the translation service in Connections &amp; Credentials are not valid. For information on connections and credentials, see Dynamic translation overview.</td>
<td></td>
</tr>
<tr>
<td>• Text cannot be translated to target languages. (40056): The specified translation service is not able to translate the passed in text into the specified target languages.</td>
<td></td>
</tr>
</tbody>
</table>

This example shows a server script translating plain text from English (detected) into French and Italian using Microsoft’s translation service.

```javascript
try {
    gs.info(JSON.stringify(sn_dt_api.DynamicTranslation.getTranslation("Translate this text using platform from server", {
        "targetLanguages": ["fr", "it"],
        "additionalParameters": [{
            "parameterName": "texttype",
            "parameterValue": "plain"
        },
        "translator": "Microsoft"
    })));
} catch (error) {
    gs.info(error.message);
}
```

Response:

```javascript
[*translations*]:
{
    "targetLanguage": "it",
    "translatedText": "Tradurre questo testo utilizzando la piattaforma dal server"
},
{
    "targetLanguage": "fr",
    "translatedText": "Traduire ce texte en utilisant la plate-forme à partir du serveur"
},
"translator": "Microsoft",
```
This example shows a server script that throws an error when an invalid target language is passed in.

```javascript
try {
    gs.info(JSON.stringify(sn_dt_api.DynamicTranslation.getTranslation("Translate this text using platform from server", {
        "targetLanguages": ["123"],
        "additionalParameters": [{
            "parameterName": "texttype",
            "parameterValue": "plain"
        }],
        "translator": "Microsoft"
    }));
} catch (error) {
    gs.info(error.message);
}
```

Response:

```
{"code":"40054","message":"Target language is invalid"}
```

**DynamicTranslation - getTranslations(Array texts, Object parms)**

Translates the passed in text strings to one or more languages.

The method uses translation services, such as Microsoft Azure Translator Service, IBM Watson Translator Service, and Google Cloud Translator Service, to perform the translation. If you do not pass in translation parameters, the method uses the system default.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>texts</td>
<td>Array</td>
<td>List of text strings to translate.</td>
</tr>
<tr>
<td>parms</td>
<td>Object</td>
<td>Optional. JSON object that contains additional translation parameters.</td>
</tr>
</tbody>
</table>

"parms": {
    "additionalParameters": {
        "sourceLanguage": "String",
    },
...
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parms.additionalParameters</td>
<td>Object</td>
<td>Optional. Array of JSON objects. Each object contains key-value pairs that provide additional information for performing the translation.</td>
</tr>
<tr>
<td>parms.additionalParameters.parameterName</td>
<td>String</td>
<td>Optional. Key name. Valid values: • textype: Type of text to translate. For Microsoft Azure Translator Service only.</td>
</tr>
<tr>
<td>parms.additionalParameters.parameterValue</td>
<td>String</td>
<td>Optional. Value of the associated key. Valid values: • plain: Standard text string • html: HTML text string Default: plain</td>
</tr>
<tr>
<td>parms.sourceLanguage</td>
<td>String</td>
<td>Optional. Language code of the source text. Default: Translation service detects the source language.</td>
</tr>
<tr>
<td>parms.targetLanguages</td>
<td>Array</td>
<td>Optional. List of language codes to use to translate the text. The method returns translated text for each language code.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parms.translator</td>
<td>String</td>
<td>Optional. Translation service to use to translate the text (not case-sensitive).</td>
</tr>
</tbody>
</table>

Valid values:
- Google
- Microsoft
- IBM
- <custom>

**Note:** To use custom translation services you must first configure the translation service in your instance. For details, see [Integrate with a translation service provider](#).

Default: Translation service configured in the Translator Configuration table.

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>Status of the response to the method call. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Error</td>
</tr>
<tr>
<td></td>
<td>• Partial</td>
</tr>
<tr>
<td></td>
<td>• Success</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>translations</td>
<td>Array of objects that describe the language translations.</td>
</tr>
<tr>
<td>translations.isError</td>
<td>Flag that indicates whether the translation of the text resulted in an error.</td>
</tr>
<tr>
<td>translations.detectedLanguage</td>
<td>Description of the detected language.</td>
</tr>
<tr>
<td>translations.detectedLanguage.code</td>
<td>Language code of the detected language.</td>
</tr>
<tr>
<td>translations.detectedLanguage.name</td>
<td>Language code of the detected language.</td>
</tr>
</tbody>
</table>

- **Data type:** String
- **Data type:** Array
- **Data type:** Boolean
- **Data type:** Object
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>textTranslations</td>
<td>Array of objects. Description of the language used to translate the text string.</td>
</tr>
<tr>
<td>textTranslations.targetLanguage</td>
<td>Language code to which the source text was translated.</td>
</tr>
<tr>
<td>textTranslations.translatedText</td>
<td>Translated text.</td>
</tr>
<tr>
<td>translator</td>
<td>Translation service used to translate the text.</td>
</tr>
<tr>
<td>Error messages</td>
<td>The following are error messages that the method may return and indications as to their root cause.</td>
</tr>
<tr>
<td></td>
<td>• Text (&quot;text&quot; field) is missing or invalid. (40000): The text to translate is either missing or not a string.</td>
</tr>
<tr>
<td></td>
<td>• Dynamic Translation plugin is not installed. (40001): The Dynamic Translation API was invoked without activating the com.glide.dynamic_translation plugin. For information on activating this plugin, see Dynamic translation overview.</td>
</tr>
<tr>
<td></td>
<td>• Default translator is not configured for translation. (40002): No translation service is selected as the default translation service in the Translator Configurations. For information on creating/modifying a translator</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>configuration, see Create a translator configuration.</td>
</tr>
<tr>
<td></td>
<td>• Translator (&quot;translator&quot; field) is invalid. (40003): The passed in translator parameter is not a string.</td>
</tr>
<tr>
<td></td>
<td>• &lt;translator&gt; translator is not configured. (40004): The specified translation service is not configured in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
</tr>
<tr>
<td></td>
<td>• &lt;translator&gt; translator is inactive. (40005): The specified translation service is not set to Active in the Translator Configuration. For information on creating/modifying a translator configuration, see Create a translator configuration.</td>
</tr>
<tr>
<td></td>
<td>• Additional parameters are invalid. (40006): The additional parameters that were passed are not an object.</td>
</tr>
<tr>
<td></td>
<td>• Target languages (&quot;targetLanguages&quot; field) are invalid. (40007): The targetLanguages parameter is passed in the call but is not valid for one of the following reasons:  ◦ Value is not an array  ◦ Array is empty  ◦ One or multiple of the entries is not a string</td>
</tr>
<tr>
<td></td>
<td>• Source language (&quot;sourceLanguage&quot; field) is invalid. (40008): The sourceLanguage parameter is passed in the call but the value is not a String.</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Maximum time limit has been exceeded. (40009): The operation took longer than the defined timeout value specified in the Translation Configuration. Default: 40 seconds.</td>
<td></td>
</tr>
<tr>
<td>• Request failed with multiple errors. (40010): Multiple errors occurred in the language detection call. For more information, refer to the response for each individual text string.</td>
<td></td>
</tr>
<tr>
<td>• <code>&lt;translator&gt;</code> translator is not configured for translation. (40012): The specified translation service is not configured for text translation in the Translator Configuration. For information on creating/modifying a translator configuration, see <a href="#">Create a translator configuration</a>.</td>
<td></td>
</tr>
<tr>
<td>• Translator configuration version is invalid. Migrate to v3. (40014): The associated version of the Translator Configuration for the specified translation service does not support the specified text translation method. For more information, see <a href="#">Migrate to version v3 of a translator configuration</a>.</td>
<td></td>
</tr>
<tr>
<td>• Unknown error occurred. (40051): Default error thrown when the error doesn’t fall in to any other category.</td>
<td></td>
</tr>
<tr>
<td>• Text (&quot;text&quot; field) has exceeded its maximum length. (40052): The text that was passed in to translate exceeds the maximum length supported by the corresponding translation service.</td>
<td></td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Source language is invalid. (40053): The passed in <code>sourceLanguage</code> parameter contains a language code that is not supported by the corresponding translation service.</td>
</tr>
<tr>
<td></td>
<td>• Target language is invalid. (40054): One or more of the language codes passed in the <code>targetLanguages</code> parameter is not supported by the corresponding translation service.</td>
</tr>
<tr>
<td></td>
<td>• Request is not authorized because credentials are missing or invalid (40055): The credentials configured for the translation service in Connections &amp; Credentials are not valid. For information on connections and credentials, see Dynamic translation overview.</td>
</tr>
<tr>
<td></td>
<td>• Text cannot be translated to target languages. (40056): The specified translation service is not able to translate the passed in text into the specified target languages.</td>
</tr>
</tbody>
</table>

This example shows code in a server-side script that translates a list of text strings into French and Italian using the Microsoft translation service.

```javascript
try {
    gs.info(JSON.stringify(sn_dt_api.DynamicTranslation.getTranslations(["Translate first text using platform from server", "Translate second text using platform from server"], {
        "targetLanguages": ["fr", "it"],
        "additionalParameters": [{"parameterName": "texttype", "parameterValue": "plain"}],
        "translator": "Microsoft"
    }));
} catch (error) {
    gs.error(error.message);
}
```

Response:
This example shows a server script that returns a Partial status when one of the two passed in text strings is invalid.

```javascript
try {
  gs.info(JSON.stringify(sn_dt_api.DynamicTranslation.getTranslations(["Translate first text using platform from server", ""], {
    "targetLanguages": ["fr", "it"],
  }));
}
```
This example shows a server script that throws an error when an invalid translation service is passed in.

```js
try {
  gs.info(JSON.stringify(sn_dt_api.DynamicTranslation.getTranslations(["Translate first text using platform from server", "Translate second text using platform from server"], {
    "targetLanguages": ["fr", "it"],
    "additionalParameters": [{"parameterName": "texttype", "parameterValue": "plain"}],
    "translator": 123
  }));
} catch (error) {
  gs.error(error.message);
}
```
DynamicTranslation - isEnabled(String translator)

Determines whether the various methods in the DynamicTranslation API are enabled for a translation service.

If you pass in a specific translation service, the method checks the method activation for that translation service; otherwise the method checks the default translation service.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>translator</td>
<td>String</td>
<td>Optional. Translation service to use to verify whether the methods are active. Translation services are configured under the Translator Configuration menu. Possible values - not case-sensitive: • Google • Microsoft • IBM • &lt;custom&gt; Note: To use custom translation services you must first configure the translation service in your instance. For details, see Integrate with a translation service provider. Default: Default translation service.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>batchDetection</td>
<td>Flag that indicates whether the getDetectedLanguages() method is enabled.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Enabled</td>
</tr>
<tr>
<td></td>
<td>• false: Disabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>batchTranslation</td>
<td>Flag that indicates whether the <code>getTranslations()</code> method is enabled. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Enabled</td>
</tr>
<tr>
<td></td>
<td>• false: Disabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>detection</td>
<td>Flag that indicates whether the <code>getDetectedLanguage()</code> method is enabled. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Enabled</td>
</tr>
<tr>
<td></td>
<td>• false: Disabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>translation</td>
<td>Flag that indicates whether the <code>getTranslation()</code> method is enabled. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Enabled</td>
</tr>
<tr>
<td></td>
<td>• false: Disabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
</tbody>
</table>

Error messages

The following are error messages that the API may return and indications as to their root cause.

• Translator ("translator" field) is invalid. (40003): The passed in `translator` parameter is not a string.

This example shows a server script that checks whether the `DynamicTranslation` methods are active for the Microsoft translator.
try {
    var response = sn_dt_api.DynamicTranslation.isEnabled('Microsoft');
    gs.info(JSON.stringify(response));
} catch(error) {
    gs.info(error.message);
}

Output:

{"detection":true,"batchTranslation":true,"batchDetection":true,"translation":true}

This example shows a server script that throws an error when an invalid translation service is passed in.

try {
    var response = sn_dt_api.DynamicTranslation.isEnabled(123);
    gs.info(JSON.stringify(response));
} catch(error) {
    gs.info(error.message);
}

Output:

{"code":"40003","message":"Translator ("translator" field) is invalid"}

Encoder - Global

Scriptable object used in Predictive Intelligence stores. This object converts input data into vectors of numbers, based on encoder-specific goals and configurations. Encoders can be used independently to run encodings, or can be configured as part of solutions to encode text columns.

The Encoder API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the sn_ml namespace.

Encoders are text processing objects that are either pre-trained or trained based on the language datasets you provide. You can train encoders that determine how the system interprets and processes text fields. For ML solutions that include text, you can train an encoder to specify how to process text and use the trained encoder in a solution.

Encoders have configuration and versions, and can be trained independently with their own retraining frequency. API-defined encoders are different from UI-defined encoders, because the retraining of UI-defined encoders is controlled by the solutions using them.

The encoder setup-to-training flow is as follows:
1. Create one or more datasets using the `DatasetDefinition` API.
2. Use the `constructor` to create an encoder object.
3. Add the encoder object to the encoder store using the `EncoderStore.add()` method.
4. Train the encoder using the `submitTrainingJob()` method. This creates a version of the object that you can manage using the `EncoderVersion` API.

Once you have trained an encoder, you can use it in a solution object:

- `ClassificationSolution`
- `ClusteringSolution` (required unless using the Levenshtein distance algorithm)
- `RegressionSolution`
- `SimilaritySolution` (required)

**Note:** This API runs with full privileges. To restrict user access, include an access control mechanism in the script.

For usage guidelines, refer to Using ML APIs.

**Encoder - Encoder(Object config)**

Creates an encoder.

To get an encoder for one or more datasets, use this constructor to create a new encoder object with a unique name.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>Object</td>
<td>JavaScript object containing configuration properties of the encoder.</td>
</tr>
</tbody>
</table>

```json
{
  "algorithmConfig": {Object},
  "datasets": [Array],
  "domainName": "String",
  "label": "String",
  "minRowCount": "String",
  "processingLanguage": "String",
  "stopwords": [Array],
}
```
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config.algorithmConfig</td>
<td>Object</td>
<td>Optional. JavaScript object containing algorithm configuration properties.</td>
</tr>
<tr>
<td>config.algorithmConfig.algo</td>
<td>String</td>
<td>Name of the algorithm for training this encoder. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• paravec: Paragraph vector word embedding.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tf-idf: Term Frequency–Inverse Document Frequency (TF-IDF)-based text.</td>
</tr>
<tr>
<td>config.datasets</td>
<td>Array</td>
<td>List of DatasetDefinition object names.</td>
</tr>
<tr>
<td>config.domainName</td>
<td>String</td>
<td>Optional. Domain name associated with this dataset. See Domain separation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>and Predictive Intelligence.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Current domain, for example, &quot;global&quot;.</td>
</tr>
<tr>
<td>config.label</td>
<td>String</td>
<td>Identifies the prediction task.</td>
</tr>
<tr>
<td>config.minRowCount</td>
<td>String</td>
<td>Optional. Minimum number of records required in the dataset for training.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: 10000</td>
</tr>
<tr>
<td>config.processingLanguage</td>
<td>String</td>
<td>Optional. Processing language in two-letter ISO 639-1 language code format.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config.stopwords</td>
<td>Array</td>
<td>Optional. Preset list of strings that the system automatically generates based on the language property setting. For details, see Create a custom stopwords list. Default: English Stopwords</td>
</tr>
<tr>
<td>config.trainingFrequency</td>
<td>String</td>
<td>The frequency to retrain the model. Possible values: every_30_days, every_60_days, every_90_days, every_120_days, every_180_days, run_once Default: run_once</td>
</tr>
</tbody>
</table>

The following example shows how to create an encoder job and add it to the encoder store.

```javascript
var myPrbData = new sn_ml.DatasetDefinition({
    'tableName': 'problem',
    'fieldNames': ['short_description'],
    'encodedQuery': 'activeANYTHING'
});

var myIncidentData = new sn_ml.DatasetDefinition({
    'tableName': 'incident',
    'fieldNames': ['short_description', 'description'],
    'encodedQuery': 'activeANYTHING'
});

var myEncoder = new sn_ml.Encoder({
```
Encoder - cancelTrainingJob()

Cancels a job for a encoder object that has been submitted for training.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to cancel an existing training job.

```javascript
var myEncoder = sn_ml.EncoderStore.get('ml_sn_global_global_encoder');
myEncoder.cancelTrainingJob();
```

Encoder - getActiveVersion()

Gets the active EncoderVersion object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Active EncoderVersion object.</td>
</tr>
</tbody>
</table>
The following example shows how to get an active Encoder version from the store and return its training status.

```javascript
var mlEncoder = sn_ml.EncoderStore.get('ml_x_snc_global_global_encoder');

gs.print(JSON.stringify(JSON.parse(mlEncoder.getActiveVersion().getStatus()), null, 2));
```

Output:

```javascript
{
  "state": "encoder_complete",
  "percentComplete": "100",
  "hasJobEnded": "true"
}
```

**Encoder - getAllVersions()**

Gets all versions of an encoder.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Existing versions of an encoder object. See also EncoderVersion API.</td>
</tr>
</tbody>
</table>

The following example shows how to get all Encoder version objects and call the getVersionNumber() and getStatus() encoder version methods on them.

```javascript
var mlEncoder = sn_ml.EncoderStore.get('ml_x_snc_global_global_encoder');

var mlEncoderVersions = mlEncoder.getAllVersions();

for (i = 0; i < mlEncoderVersions.length; i++) {
  gs.print("Version "+ mlEncoderVersions[i].getVersionNumber() + " Status: " +
            mlEncoderVersions[i].getStatus() +"\n");
}
```

Output:

```javascript
Version 3 Status: {"state":"encoder_complete","percentComplete":"100","hasJobEnded":"true"}
```
Encoder - getLatestVersion()

Gets the latest version of an encoder.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>EncoderVersion object corresponding to the latest version of an Encoder().</td>
</tr>
</tbody>
</table>

The following example shows how to get the latest version of an encoder and return its training status.

```javascript
var mlEncoder = sn_ml.EncoderStore.get('ml_x_snc_global_global_encoder');

gs.print(JSON.stringify(JSON.parse(mlEncoder.getLatestVersion().getStatus()), null, 2));
```

Output:

```json
{
    "state": "encoder_complete",
    "percentComplete": "100",
    "hasJobEnded": "true"
}
```

### Encoder - getName()

Gets the name of the object to use for interaction with the store.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The following example shows how to update Encoder dataset information and print the name of the object.

```javascript
// Update encoder
var myIncidentData = new sn_ml.DatasetDefinition({
    'tableName': 'incident',
    'fieldNames': ['category', 'short_description', 'priority'],
    'encodedQuery': 'activeANYTHING'
});

var eligibleFields = JSON.parse(myIncidentData.getEligibleFields(encoder));

var myEncoder = new sn_ml.Encoder({
    'label': 'my encoder',
    'datasets': [myIncidentData],
    'inputFieldNames': eligibleFields['eligibleInputFieldNames'],
    'predictedFieldName': 'category'
});

// update encoder
sn_ml.EncoderStore.update('ml_x_snc_global_global_my_definition_4', myEncoder);

// print encoder name
gs.print('Encoder Name: ' + myEncoder.getName());
```

Output:

```
Encoder Name: ml_x_snc_global_global_my_definition_4
```

**Encoder - getProperties()**

Gets solution object properties.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Object</td>
<td>Contents of the Dataset and Encoder() object details in the EncoderStore.</td>
<td></td>
</tr>
</tbody>
</table>

```json
{
  "algorithmConfig": {Object},
  "datasetsProperties": [Array],
  "domainName": "String",
  "label": "String",
  "name": "String",
  "processingLanguage": "String",
  "scope": "String",
  "stopwords": [Array],
  "trainingFrequency": "String"
}
```

- `<Object>.algorithmConfig.algorithm`
  - Name of the algorithm for training this encoder.
  - Possible values:
    - `paravec`: Paragraph vector word embedding.
    - `tf-idf`: Term Frequency–Inverse Document Frequency (TF-IDF)-based text.
  - Data type: String.

- `<Object>.algorithmConfig` Optional. JavaScript object containing
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>algorithm configuration properties.</td>
</tr>
<tr>
<td></td>
<td>`{&quot;algorithmConfig&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;algorithm&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>} }</td>
</tr>
<tr>
<td></td>
<td>Data type: Object.</td>
</tr>
<tr>
<td></td>
<td><code>&lt;Object&gt;.datasetsProperties</code></td>
</tr>
<tr>
<td></td>
<td>List of <code>&lt;DatasetDefinition&gt;</code> properties associated with the encoder.</td>
</tr>
<tr>
<td></td>
<td>`{</td>
</tr>
<tr>
<td></td>
<td>&quot;encodedQuery&quot;:</td>
</tr>
<tr>
<td></td>
<td>&quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;fieldDetails&quot;:</td>
</tr>
<tr>
<td></td>
<td>[Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;fieldNames&quot;:</td>
</tr>
<tr>
<td></td>
<td>[Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;tableName&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: Array.</td>
</tr>
<tr>
<td></td>
<td><code>&lt;Object&gt;.datasetsProperties.tableName</code></td>
</tr>
<tr>
<td></td>
<td>Name of the table for the dataset. For example, &quot;tableName&quot; : &quot;Incident&quot;.</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
<tr>
<td></td>
<td><code>&lt;Object&gt;.datasetsProperties.fieldNames</code></td>
</tr>
<tr>
<td></td>
<td>List of field names from the specified table as strings. For example, &quot;fieldNames&quot; : [&quot;short_description&quot;, &quot;priority&quot;].</td>
</tr>
<tr>
<td></td>
<td>Data type: Array.</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;.datasetsProperties.fieldNames.fieldDetails</code></td>
<td>List of JavaScript objects that specify field properties.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><code>&lt;Object&gt;.datasetsProperties.fieldNames.fieldDetails.&lt;object&gt;.name</code></td>
<td>Name of the field defining the type of information to restrict this dataset to. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.datasetsProperties.fieldDetails.&lt;object&gt;.type</code></td>
<td>Machine-learning field type. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.datasetsProperties.fieldDetails.encodedQuery</code></td>
<td>Encoded query string in standard Glide format. See Encoded query strings. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.domainName</code></td>
<td>Domain name associated with this dataset. See Domain separation and Predictive Intelligence. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.label</code></td>
<td>Identifies the prediction task.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;label&quot;: &quot;my first prediction&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.name</td>
<td>System-assigned name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.predictedFieldName</td>
<td>Identifies a field to be trained for predictability.</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.processingLanguage</td>
<td>Processing language in two-letter ISO 639-1 language code format.</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.scope</td>
<td>Object scope. Currently the only valid value is <code>global</code>.</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.stopwords</td>
<td>Optional. Preset list of strings that the system automatically generates based on the <code>language</code> property setting. For details, see Create a custom stopwords list.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array.</td>
</tr>
<tr>
<td>&lt;Object&gt;.trainingFrequency</td>
<td>The frequency to retrain the model. Possible values:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>every_30_days</td>
<td></td>
</tr>
<tr>
<td>every_60_days</td>
<td></td>
</tr>
<tr>
<td>every_90_days</td>
<td></td>
</tr>
<tr>
<td>every_120_days</td>
<td></td>
</tr>
<tr>
<td>every_180_days</td>
<td></td>
</tr>
<tr>
<td>run_once</td>
<td></td>
</tr>
</tbody>
</table>

Default: run_once

Data type: String.

The following example gets properties of an encoder object in the store.

```javascript
var myEncoder = sn_ml.EncoderStore.get('ml_sn_global_global_encoder');

gs.print(JSON.stringify(JSON.parse(myEncoder.getProperties()), null, 2));
```

Output:

```json
*** Script: {
    "datasetsProperties": [
        {
            "tableName": "incident",
            "fieldNames": [
                "assignment_group",
                "short_description",
                "description"
            ],
            "encodedQuery": "activeANYTHING"
        }
    ],
    "domainName": "global",
    "label": "my encoder definition",
    "name": "ml_x_snc_global_global_my_encoder_definition",
    "processingLanguage": "en",
    "scope": "global",
    "stopwords": [
        "Default English Stopwords"
    ],
```
Encoder - getVersion(String version)

Gets an encoder by provided version number.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Existing version number of an encoder.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Specified version of the Encoder() object on which you can call EncoderVersion API methods.</td>
</tr>
</tbody>
</table>

The following example shows how to get the training status of an encoder by version number.

```javascript
var mlEncoder = sn_ml.EncoderStore.get('ml_x_snc_global_global_encoder');

gs.print(JSON.stringify(JSON.parse(mlEncoder.getVersion('1').getStatus()), null, 2));
```

Output:

```javascript
{
  "state": "encoder_complete",
  "percentComplete": "100",
  "hasJobEnded": "true"
}
```

Encoder - setActiveVersion(String version)

Activates a specified version of an encoder in the store.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Name of the Encoder() object version to activate.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Activating this version deactivates any other version.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to activate an encoder version in the store.

```javascript
sn_ml.Encoder.setActiveVersion("ml_incident_categorization");
```

**Encoder - submitTrainingJob()**

Submits a training job.

ℹ️ **Note:** Before running this method, you must first add an encoder to the store using the `EncoderStore - add()` method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>EncoderVersion object corresponding to the Encoder being trained.</td>
</tr>
</tbody>
</table>

The following example shows how to create a dataset, apply it to an encoder, add it to a store, and submit the training job.

```javascript
// Create a dataset
var myData = new sn_ml.DatasetDefinition({
  'tableName' : 'incident',
  'fieldNames' : ['assignment_group', 'short_description', 'description'],
  'encodedQuery' : 'activeANYTHING'
});
```
// Create an encoder
var myEncoder = new sn_ml.Encoder(
    
    'label': "my encoder definition",
    'datasets' : [myData],
    'predictedFieldNames' : 'assignment_group',
    'inputFieldNames': ['short_description']

);

// Add the encoder to the store to later be able to retrieve it.
var my_unique_name = sn_ml.EncoderStore.add(myEncoder);

// Train the encoder - this is a long running job
var myEncoderVersion = myEncoder.submitTrainingJob();

EncoderStore - Global

Enables storing and retrieving encoders.

The EncoderStore API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the sn_ml namespace.

EncoderStore - add(Object mlEncoder)

Adds a new encoder object to the store and returns a unique name.

⚠️ Note: Label values do not need to be unique. For example, if you run this method with the same label 10 times, this method adds 10 different uniquely-named objects to the store.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mlEncoder</td>
<td>Encoder</td>
<td>Encoder() object to add to the store.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>System-generated solution name.</td>
</tr>
</tbody>
</table>
The following example shows how to add an encoder to the store. Use `Encoder.submitTrainingJob()` to run the training job after adding it to the store.

```javascript
// Create a dataset
var myData = new sn_ml.DatasetDefinition({
    'tableName': 'incident',
    'fieldNames': ['assignment_group', 'short_description', 'description'],
    'encodedQuery': 'activeANYTHING'
});

// Create a solution
var myEncoder = new sn_ml.Encoder({
    'label': "my encoder definition",
    'datasets': [myData],
    'predictedFieldName': 'assignment_group',
    'inputFieldNames': ['short_description']
});

// Add the encoder to the store to later be able to retrieve it.
var my_unique_name = sn_ml.EncoderStore.add(myEncoder);
```

**EncoderStore - deleteObject(String name)**

Removes a specified encoder object from the store.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the <code>Encoder()</code> object to be deleted.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to delete an encoder from the store.

```javascript
sn_ml.EncoderStore.deleteObject("ml_sn_global_global_encoder");
```
EncoderStore - get(String name)
Gets an encoder object from a store.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of an encoder in a store.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Encoder object. Returns an error if the object does not exist.</td>
</tr>
</tbody>
</table>

The following example shows how to get an encoder object from the store using the get() method and view its training status using the Encoder - getActiveVersion() and EncoderVersion - getStatus() methods.

```javascript
// Get status
var myEncoder = sn_ml.EncoderStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(myEncoder.getActiveVersion().getStatus(), null, 2)));
```

Output:

```
{
    "state":"solution_complete",
    "percentComplete":"100",
    "hasJobEnded":"true"
}
```

EncoderStore - getAllNames(Object options)
Gets the names of all encoder definition records in the store.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Options for restricting results within the specified properties.</td>
</tr>
</tbody>
</table>

```javascript
{
    "label": "String",
    "domainName": "String",
}
```
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options.label</td>
<td>String</td>
<td>Optional. Label of your solution object.</td>
</tr>
<tr>
<td>options.domainName</td>
<td>String</td>
<td>Optional. Name of the domain for your solution object. Refer to Domain separation and Predictive Intelligence.</td>
</tr>
<tr>
<td>options.scope</td>
<td>String</td>
<td>Optional. Name of an application scope for your solution object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of strings representing encoder object names in the store.</td>
</tr>
</tbody>
</table>

In the following example, the `getAllNames()` method returns a list of all names in the store.

```javascript
gs.print(JSON.stringify(JSON.parse(sn_ml.EncoderStore.getAllNames()), null, 2));
```

**Output:**

```javascript
[
  "ml_x_snc_global_global_classification_word_corpus",
  "ml_x_snc_global_global_predictability_estimate",
  "GloVe",
  "ml_x_snc_global_global_encoder",
  "ml_x_snc_global_global_predictability_estimate_1"
]
```

In the following example, the `getAllNames()` method returns only names associated with values set in the `options` parameter.

```javascript
var options = {
  'label' : 'my encoder definition',
  'domainName' : 'global',
  'scope' : 'global'
};
```
Output:

```
[
  "ml_x_snc_global_global_my_encoder_definition"
]
```

**EncoderStore - update(String name, Object mlEncoder)**

Updates an encoder object in a store.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the encoder to update.</td>
</tr>
<tr>
<td>mlEncoder</td>
<td>Encoder</td>
<td>Encoder() object properties to update.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to update an encoder object in the store.

```javascript
var encoderUpdate = new sn_ml.Encoder({
  'label': 'my encoder definition',
  'dataset': myData,
  'predictedFieldName': 'assignment_group',
  'inputFieldNames': ['short_description']
});

sn_ml.EncoderStore.update('ml_sn_global_global_incident_service', encoderUpdate);
```

**EncoderVersion - Global**

Scriptable object used in Predictive Intelligence stores.

The EncoderVersion API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the sn_ml namespace.

This API is used for working with encoder versions based on Encoder API objects in the Encoder store.
The system activates the most recent version of the encoder when it completes training, and only allows one version to be active at a time. However, you can activate any previously trained version you want to use to make predictions.

Methods in this API are accessible using the following Encoder methods:

- `getActiveVersion()`
- `getAllVersions()`
- `getLatestVersion()`
- `getVersion()`

**EncoderVersion - getProperties()**

Gets encoder object properties and version number.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Contents of the Dataset and Encoder details. Results vary by object property setup.</td>
</tr>
</tbody>
</table>

```
{
    // Object
    "algorithmConfig": {Object},
    "datasetsProperties": [Array],
    "domainName": "String",
    "isActive": "String",
    "label": "String",
    "name": "String",
    "predictedFieldName": "String",
    "processingLanguage": "String",
    "scope": "String",
    "stopwords": [Array],
    "trainingFrequency": "String",
    "versionNumber": "Number"
}
```

## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| `{algorithmConfig: {  
  "algorithm": "String"  
}}` | Data type: Object. |

**algorithmConfig**

Name of the algorithm for training this encoder.

Possible values:

- **paravec**: Paragraph vector word embedding.
- **tf-idf**: Term Frequency–Inverse Document Frequency (TF-IDF)-based text.

Data type: String.

### <Object>.datasetsProperties

List of `DatasetDefinition()` properties associated with the encoder.

```json
{
  "encodedQuery": "String",
  "fieldDetails": [Array],
  "fieldNames": [Array],
  "tableName": "String"
}
```

Data type: Array.

**tableName**

Name of the table for the dataset. For example, "tableName" : "Incident".

Data type: String.

**fieldNames**

List of field names from the specified table as strings. For example, "fieldNames" : ["short_description", "priority"].

Data type: Array.

**fieldDetails**

List of JavaScript objects that specify field properties.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>}</td>
<td>Data type: Array.</td>
</tr>
</tbody>
</table>

`<Object>.datasetsProperties.fieldDetails.field@type`<br>Field of the type object that information to restrict this dataset to.<br>Data type: String.<br>

`<Object>.datasetsProperties.fieldDetails.<object>.type`<br>Machine-learning field type.<br>Data type: String.<br>

`<Object>.datasetsProperties.encodedQuery`<br>Encoded query string in standard Glide format. See Encoded query strings.<br>Data type: String.<br>

`<Object>.domainName`<br>Domain name associated with this dataset. See Domain separation and Predictive Intelligence.<br>Data type: String.<br>

`<Object>.isActive`<br>Flag that indicates whether this version is active. Valid values:<br>• true: Version is active.<br>• false: Version is not active.<br>Data type: String.<br>

`<Object>.label`<br>Identifies the prediction task.<br>{<br>  "label": "my first prediction"
}<br>Data type: String.<br>

`<Object>.name`<br>System-assigned name.<br>Data type: String.<br>

`<Object>.predictedFieldName`<br>Identifies a field to be trained for predictability.<br>Data type: String.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.processingLanguage</td>
<td>Processing language in two-letter ISO 639-1 language code format.</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.scope</td>
<td>Object scope. Currently the only valid value is <code>global</code>.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.stopwords</td>
<td>Optional. Preset list of strings that the system automatically generates based on the <code>language</code> property setting. For details, see <a href="#">Create a custom stopwords list</a>.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array.</td>
</tr>
<tr>
<td>&lt;Object&gt;.trainingFrequency</td>
<td>The frequency to retrain the model.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• every_30_days</td>
</tr>
<tr>
<td></td>
<td>• every_60_days</td>
</tr>
<tr>
<td></td>
<td>• every_90_days</td>
</tr>
<tr>
<td></td>
<td>• every_120_days</td>
</tr>
<tr>
<td></td>
<td>• every_180_days</td>
</tr>
<tr>
<td></td>
<td>• run_once</td>
</tr>
<tr>
<td></td>
<td>Default: run_once</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.versionNumber</td>
<td>Version number of the Encoder object.</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
</tbody>
</table>

The following example gets properties of the active object version in the store.

```javascript
// Get properties
var mlEncoder = sn_ml.EncoderStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlEncoder.getActiveVersion().getProperties()), null, 2));
```

Output:
EncoderVersion - getSentenceVectors(Array input)

Returns vectors for each input sentence.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>Array</td>
<td>Array of strings as sentences from which to receive vectors.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Array of sentence vectors.</td>
</tr>
</tbody>
</table>

The following example shows how to return a vector for a single sentence.

```javascript
var myEncoderName = 'GloVe';

var myEncoder = sn_ml.EncoderStore.get(myEncoderName);
```
```javascript
var input = "I like to code.";

var vectors = myEncoder.getActiveVersion().getSentenceVectors(input);

gs.print(vectors);
```

Output:

```javascript
*** Script: [-0.16243751347064972, 0.30614474415779114, 0.08489049971103668, 
-0.48100000619888306, -0.170997753739357, 0.08779674768447876, -0.07848624140024185, -0.15123701995581055, 
-0.07843250036239624, -1.9505999088287354, 0.3007825016975403, -0.07804800570011139, -0.04779449850320816, 
0.04803549498319626, 0.09848674386739731, 0.2427891194820404, -0.41138750314712524, 0.1088037490847266, 
... , 
0.21227750182151794, 0.18478751182556152, -0.3113832473754883, -0.16560424864292145, 0.09052124619483948]
```

**EncoderVersion - getSimilarWords(Array input, Object options)**

Returns words similar to each input word in the descending rank order of similarity.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>Array</td>
<td>Array of words for which to find similar words.</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Map to refine results.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{ &quot;topN&quot;:&quot;String&quot; }</td>
</tr>
<tr>
<td>options.topN</td>
<td>String</td>
<td>If provided, returns the top results up to the specified number of words. For example, use &quot;10&quot; to return the top 10 most similar words.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of elements containing the similar words for the input word in the corresponding position. These similar words are represented by an array of pairs in the format [word, similarity score].</td>
</tr>
</tbody>
</table>
The following example shows how to get similar words using the GloVe encoder.

```javascript
var myEncoderName = 'GloVe';
var myEncoder = sn_ml.EncoderStore.get(myEncoderName);
var input = ['apple'];
var options = {'topN': '5'};
gs.print(myEncoder.getActiveVersion().getSimilarWords(input, options));
```

Output:

```javascript
*** Script:

[[["iphone", 0.5987], ["macintosh", 0.5836], ["ipod", 0.5761], ["microsoft", 0.5664], ["ipad", 0.5628]]]
```

**EncoderVersion - getStatus(Boolean includeDetails)**

Gets training completion status.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>includeDetails</td>
<td>Boolean</td>
<td>Flag that indicates whether to return status <strong>details</strong>. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Return additional details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not return additional details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: False</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JavaScript object containing training status information for an <strong>Encoder</strong> object.</td>
</tr>
</tbody>
</table>

```javascript
{
  "state": "String",
  "percentComplete": "Number as a String",
  "hasJobEnded": "Boolean value as a String",
  "details": {Object}
}
```

<Object>.state

Training completion state. If the training job reaches a terminal state, the job does not leave that state. If the state is terminal, the **hasJobEnded** property is set to **true**.
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• fetching_files_for_training</td>
</tr>
<tr>
<td></td>
<td>• preparing_data</td>
</tr>
<tr>
<td></td>
<td>• retry</td>
</tr>
<tr>
<td></td>
<td>• solution_cancelled <em>(terminal)</em></td>
</tr>
<tr>
<td></td>
<td>• solution_complete <em>(terminal)</em></td>
</tr>
<tr>
<td></td>
<td>• solution_error <em>(terminal)</em></td>
</tr>
<tr>
<td></td>
<td>• solution_incomplete</td>
</tr>
<tr>
<td></td>
<td>• training_request_received</td>
</tr>
<tr>
<td></td>
<td>• training_request_timed_out <em>(terminal)</em></td>
</tr>
<tr>
<td></td>
<td>• training_solution</td>
</tr>
<tr>
<td></td>
<td>• uploading_solution</td>
</tr>
<tr>
<td></td>
<td>• waiting_for_training</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.hasJobEnded</td>
<td>Flag that indicates whether training is complete.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Training is complete.</td>
</tr>
<tr>
<td></td>
<td>• false: Training is incomplete.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean value as a String</td>
</tr>
<tr>
<td>&lt;Object&gt;.percentComplete</td>
<td>Number between zero and 100 representing training percent complete.</td>
</tr>
<tr>
<td></td>
<td>If the completion percentage is less than 100, the job might be in a terminal state. For example, if training times out.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number as a String</td>
</tr>
<tr>
<td>&lt;Object&gt;.details</td>
<td>Object containing a list of additional training details.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
</tbody>
</table>

The following example shows a successful result with training complete.
// Get status
var mlEncoder = sn_ml.EncoderStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlEncoder.getActiveVersion().getStatus(true), null, 2)));

Output:

{
  "state":"solution_complete",
  "percentComplete":"100",
  "hasJobEnded":"true",
  "details":{"stepLabel":"Encoder Complete"} // This information is only returned if
  getStatus(true);
}

The following example shows an unsuccessful result with training complete.

// Get status
var encoderName = 'ml_x_snc_global_global_encoder';
var mlEncoder = sn_ml.EncoderStore.get(encoderName);
var trainingStatus = mlEncoder.getLatestVersion().getStatus();

gs.print(JSON.stringify(JSON.parse(trainingStatus), null, 2));

Output:

{
  "state":"solution_error",
  "percentComplete":"100",
  "hasJobEnded":"true"
}

EncoderVersion - getVersionNumber()

Gets the version number of a solution object.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The following example shows how to get a version number.

```javascript
// Get version number
var mlEncoder = sn_ml.EncoderStore.get('ml_incident_categorization');

gs.print("Version number:
  "+JSON.stringify(JSON.parse(mlEncoder.getActiveVersion().getVersionNumber()), null, 2));
```

Output:

```
Version number: 1
```

**EncoderVersion - getWordVectors(Array input)**

Returns vectors for each input word.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>Array</td>
<td>List of strings as words from which to receive vectors.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of vectors for each word provided.</td>
</tr>
</tbody>
</table>

The following example shows how to get a vector from the word hello.

```javascript
var myEncoderName = 'GloVe';
var myEncoder = sn_ml.EncoderStore.get(myEncoderName);
var input = ["hello"];

gs.print(myEncoder.getActiveVersion().getWordVectors(input));
```

Output:

```
*** Script: [\[-0.337119996547699,-0.2169100046157837,-0.006636499892920256, -0.41624999954325684,-1.2554999589920044,-0.0284659992903471,-0.7219499945640564, -0.5288699865341187,0.0072085000574588776,0.3199700117111206,0.02942500077188015, -0.013236000202596188,0.4351100027561188,0.2571600079536438,0.3899500072002411,]
```
Event - Global

The Event API provides methods that enable the setting/getting of values within an event.

These events are sent from a MID Server to a ServiceNow instance. Before you are able to successfully send events, the connection between the MID Server and the ServiceNow instance must be defined.

Use the Event API to add/update fields within an event. Use the SNEventSenderProvider API to instantiate an event sender object. Then use the IEventSender API to send the event to a ServiceNow instance.

You must activate the Event Management (com.glideapp.itom.snac) plugin before attempting to access this API. The Event Management plugin requires a separate subscription and must be activated by ServiceNow personnel. This plugin includes demo data and activates related plugins if they are not already active.

For additional information on event management, see Event Management.

Event - Event()

Instantiates an Event constructor.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var event = new Event();
var esource = event.getField("source");
var eseverity = event.getField("severity");
```

Event - getField(String field)

Returns the current value of the passed in event management field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Name of the event management field value to return.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the requested event management field.</td>
</tr>
</tbody>
</table>

```javascript
var event = new Event();
var eventSource = event.getField("source");
var eventSeverity = event.getField("severity");
```

**Event - setAdditionalInfo(String additionalInfo)**

Replaces the existing `additional_info` field in the associated event with the passed in JSON string.

ℹ️ **Note:** You can use the `setfield()` method to update a specific field within the `additional_info` field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalInfo</td>
<td>String</td>
<td>JSON String (key/value pairs) to save in the <code>additional_info</code> field.</td>
</tr>
</tbody>
</table>

ℹ️ **Note:** This parameter replaces the existing information. It does not update the existing fields.

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
event.setAdditionalInfo("{"evtComponent":"Microsoft-Windows-WindowsUpdateClient","evtMessage":"Installation Failure: Windows failed. Error 0x80070490"}");
```

**Event - setField(String key, String value)**

Sets the specified field in the associated event.

If the passed in `key` matches a predefined event field, the method updates the corresponding event field. Otherwise, the method adds the `key` parameter to the `additional_info` section of the event before applying the passed in `value`. 
The following are the predefined event fields that you can modify using this method:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>additional_info</td>
<td>Key-value pair to add to the <code>additional_info</code> field of the associated event. If the passed in key already exists, its value is overwritten.</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000 characters</td>
</tr>
<tr>
<td></td>
<td>Associated UI field: Additional information</td>
</tr>
<tr>
<td>ci_identifier</td>
<td>JSON string that uniquely identifies a configuration item. For example, <code>{&quot;name&quot;:&quot;SAP ORA01&quot;,&quot;type&quot;:&quot;Oracle&quot;}</code>.</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 1,000 characters</td>
</tr>
<tr>
<td>cmdb_ci</td>
<td>Sys_id of the Configuration Item [cmdb_ci] record to bind the event to.</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>description</td>
<td>Free-form description of the event.</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000 characters</td>
</tr>
<tr>
<td></td>
<td>Associated UI field: Description</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>event_class</td>
<td>Enterprise Message Service (EMS) that generated the event. For example, &quot;Solarwinds&quot; or &quot;SCOM&quot;.</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td></td>
<td>Associated UI field: Source instance</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>message_key</td>
<td>Unique event identifier. To override an existing event severity, use the same <code>message_key</code> value.</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 1024 characters</td>
</tr>
<tr>
<td></td>
<td>Associated UI field: Message key</td>
</tr>
<tr>
<td></td>
<td>Default: Combination of <code>source</code>, <code>node</code>, <code>type</code>, <code>resource</code>, and <code>metric name</code>.</td>
</tr>
<tr>
<td>resolution_state</td>
<td>Event resolution state.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td></td>
<td>• New: Resolution state for the corresponding event is open.</td>
</tr>
<tr>
<td></td>
<td>• Closing: Resolution state for corresponding event is closing/closed.</td>
</tr>
<tr>
<td></td>
<td>Associated UI field: Resolution state</td>
</tr>
<tr>
<td></td>
<td>Default: New</td>
</tr>
<tr>
<td>resource</td>
<td>Node resource to associate with the event. For example, &quot;Disk C:&quot;, &quot;CPI-1&quot;, or the name of a process or service.</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100 characters</td>
</tr>
<tr>
<td></td>
<td>Associated UI field: Resource</td>
</tr>
<tr>
<td>severity</td>
<td>Event severity.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Clear - No action is required. An alert is not created from this event. Existing alerts are closed.</td>
</tr>
<tr>
<td></td>
<td>• 1: Ok - An alert is created. The resource is still functional.</td>
</tr>
<tr>
<td></td>
<td>• 2: Warning - Attention is required, even though the resource is still functional.</td>
</tr>
<tr>
<td></td>
<td>• 3: Minor - Partial, non-critical loss of functionality or performance degradation occurred.</td>
</tr>
<tr>
<td></td>
<td>• 4: Major - Major functionality is severely impaired or performance has degraded.</td>
</tr>
<tr>
<td></td>
<td>• 5: Critical - Immediate action is required. The resource is either not functional or critical problems are imminent.</td>
</tr>
<tr>
<td></td>
<td>Associated UI field: Severity</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>source</td>
<td>Instance of the EMS that triggered the event. Typically the connector instance name.</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 200 characters</td>
</tr>
<tr>
<td></td>
<td>Associated UI field: Source</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>time_of_event</td>
<td>UTC time that the event occurred in the source system.</td>
</tr>
</tbody>
</table>
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Metric type to which the event is related, such as Disk or CPU. This value is used to identify an event record from which alerts are created. Maximum length: 100 characters Associated UI field: Type Default: Null</td>
</tr>
<tr>
<td></td>
<td>Format: &quot;yyyy-MM-dd HH:mm:ss&quot; Maximum length: 40 characters Associated UI field: Time of event Default: Current date/time</td>
</tr>
</tbody>
</table>

**Note:** You can also use the `setText()` method to update the `description` field and the `setTimeOfEvent()` method to update the event time.

For additional information on events, see *Event Management*.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the key to add or update in the event. For additional information on the predefined event fields, see <em>Event field format for event collection</em>.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Information to store in the associated event field.</td>
</tr>
</tbody>
</table>

**Note:** The `value` string is trimmed before it is stored, whereby multiple spaces are replaced with a single space.

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var event = new Event();
event.setField("source","SAP Solman");
event.setField("severity","3");
```
event.setField("time_of_event", "2019-18-05 13:12:05");
event.setField("type", "SAP object");
event.setField("user", "admin"); // parameter will be added to additional_info section

Event - setText(String text)

Sets the description field of the associated event to the passed in value.

Note: You can also use the Event - setField(String key, String value) method to set the text field in the current event. Unlike the setField() method, this method does not trim the passed in text.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>Text to store in the description field of the associated event.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Max length: 4000 characters</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var event = new Event();
event.setText("SAP Solman");

Event - setTimeOfEvent(String timeOfEvent)

Sets the time_of_event field in the associated event to the passed in UTC time value.

Note: You can also use the Event - setField(String key, String value) method to set the time_of_event field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeOfEvent</td>
<td>String</td>
<td>UTC time value to set in the time_of_event field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Format: &quot;yyyy-MM-dd HH:mm:ss&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Max length: 40 characters</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var event = new Event();
```

**ExpenseAllocation - Global**

The `ExpenseAllocation` API is included with the Cost Management Plugin as a script include record. It is used by various cost management processes and can also be used for generating custom expense allocation records (`fm_expense_allocation`) from scripted expense allocation rules.

**ExpenseAllocation - createAllocation(GlideRecord target, Number amount)**

Creates an expense allocation (`fm_expense_allocation`) record referencing the parameters provided during instantiation and this method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>GlideRecord</td>
<td>GlideRecord target of the allocation, for example a cost center record to allocate an expense to Decimal amount - the amount of the allocation.</td>
</tr>
<tr>
<td>amount</td>
<td>Number</td>
<td>The amount of the allocation.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the expense allocation was successfully created.</td>
</tr>
</tbody>
</table>

```javascript
var allocation = new ExpenseAllocation(expenseGlideRecord, ruleGlideRecord);
allocation.createAllocation(costCenterGlideRecord, 2345.67);
```

**ExpenseAllocation - ExpenseAllocation(GlideRecord expense, GlideRecord rule)**

Called when you create a new `ExpenseAllocation` object.
This is not needed if scripting advanced allocation rules. This object is already available as the `allocation` variable.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>expense</td>
<td>GlideRecord</td>
<td>GlideRecord identifying the source of the expense.</td>
</tr>
<tr>
<td>rule</td>
<td>GlideRecord</td>
<td>GlideRecord identifying the rule to use in allocating the expense line.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExpenseAllocation object</td>
<td>The ExpenseAllocation object just created.</td>
</tr>
</tbody>
</table>

```javascript
var allocation = new ExpenseAllocation(expenseGlideRecord, ruleGlideRecord);
```

### ExpenseLine - Global

The `ExpenseLine` API is included with the Cost Management Plugin as a script include record. It is used by various cost management processes and can also be used for generating expense line (`fm_expense_line`) records from your own server-side scripts.

### ExpenseLine - createExpense()

Creates a new expense line record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the expense line was successfully created.</td>
</tr>
</tbody>
</table>
//get some random CI to be used as an expense source
var ci = new GlideRecord("cmdb_ci_server");
    ci.query();
    ci.next();

//create expense line
var exp = new ExpenseLine(ci, 234.56, "Test expense line");
    exp.setSummaryType("run_business");
    var success = exp.createExpense();

ExpenseLine - ExpenseLine (GlideRecord source, Number amount, String description)
Constructor for ExpenseLine object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>GlideRecord</td>
<td>GlideRecord identifying the source of the expense</td>
</tr>
<tr>
<td>amount</td>
<td>Number</td>
<td>Decimal number identifying the amount of the expense</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>(Optional) Description of the expense.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExpenseLine object</td>
<td>The ExpenseLine object just instantiated.</td>
</tr>
</tbody>
</table>

//get some random CI to be used as an expense source
    var ci = new GlideRecord("cmdb_ci_server");
    ci.query();
    ci.next();

//create expense line
    var exp = new ExpenseLine(ci, 234.56, "Test expense line");

ExpenseLine - processCIParents()
Used internally by the createExpense method to process CI relationships when the expense source is a cmdb_ci record.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**ExpenseLine - setCostSource(GlideRecord costSource)**

Identifies the source rate card or distribution cost that was the source of expense line generation.

This is not the source (CI, task) of the expense.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>costSource</td>
<td>GlideRecord</td>
<td>GlideRecord of CI rate card cost, distribution cost, or task rate card. This is generally only used for system-generated expense lines.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**ExpenseLine - setDescription(String description)**

Defines the description of an expense.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>String</td>
<td>Description of expense.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**ExpenseLine - setParent(GlideRecord expense)**

Sets the parent field on the expense line.

This is generally only used by the system when generating indirect expenses such as business service aggregated expenses.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>expense</td>
<td>GlideRecord</td>
<td>Parent expense line record.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**ExpenseLine - setRecurring(Boolean recurring)**

Flags the expense as recurring by setting the recurring field to true.

Expense lines are set to false by default so there is no need to call `setRecurring(false)`.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>recurring</td>
<td>Boolean</td>
<td>Set to true to identify expense line as a recurring expense.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
ExpenseLine - setSummaryType(String summaryType)
Sets a value for the expense line summary_type field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>summaryType</td>
<td>String</td>
<td>Typically you would set this to a value already specified in the expense line summary type field choice list.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
//get some random CI to be used as an expense source
var ci = new GlideRecord("cmdb_ci_server");
ci.query();
ci.next();

//create expense line
var exp = new ExpenseLine(ci, 234.56, "Test expense line");
exp.setSummaryType("run_business");
```

ExtractTermsFromAttachment - Global
Extracts terms from an attachment.

This class is called with the ScriptedExtractor object, SysAttachmentInputStream, the sys_id for the attachment, and the extension for the attachment. The `getTerms()` method is called to extract the terms from the attachment that should be indexed. The `getTerms()` method should just return a string that contains the terms. If you prefer to input a file rather than an input stream, call `extractor.getFile()` to get the File object containing the attachment.

Findings API - Scoped, Global
Provides methods for handling how an Instance Scan check produces findings. This API is included with the Instance Scan (com.glide.instance_scan) plugin. For more information, see Instance Scan.

In Instance Scan, checks run on each record in the scope of a scan on the instance to generate findings. Each finding holds information about an issue record and which check it violated. Findings API methods are used in Instance
Scan checks as part of the `engine` object passed to the user. Add the code to the `Script` field in the check form.

See also:
- Getting started with checks
- Findings

**Findings - increment()**

Increases the count of the current finding.

The finding count starts at zero for each record that a check analyzes in a scan. When called, this method increments the finding count. The count indicates that a finding is to be generated for the current record.

This method can be called multiple times to signify that this finding has multiple occurrences of a check violation in the current record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to count the number of records with a failed state in a table type check. For more information, see Create a table type check.

```javascript
(function(engine) {
    if (current.getValue("state") == "failed") {
        engine.finding.increment();
    }
})(engine);
```

**Findings - incrementWithNode(Object node)**

If the current finding is from a linter check, this method increments the current finding count and simultaneously passes the linter node object to the finding.
Use this method in the Script field in the Linter Check form. This method saves information about the given node, such as a line number in a current finding’s details column.

See also:
• Advanced linter check scripts
• LinterCheckAstNode API

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>node</td>
<td>Object</td>
<td>Node object from the linter check.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to increment linter type nodes in a linter check. For information on using linter check methods, see LinterCheckAstNode API.

```javascript
(function(engine) {
  engine.rootNode.visit(function(node) {
    if (node.getTypeName() === "NAME" &&
        node.getNameIdentifier() === "soughtFunction" &&
        node.getParent().getTypeName() === "CALL") {
      engine.finding.incrementWithNode(node);
    }
  });
})(engine);
```

**Findings - setCurrentSource(GlideRecord source)**

Sets the source of the current finding based on the provided GlideRecord.

Use this method in the **Script** field of a Script Only check. This method is not used for any other checks because they automatically set the source as the current record during the scan. For more information, see Create a script only check.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>GlideRecord</td>
<td>The record to set as the source record for the current finding. This value is added as a reference in the Source field of the Scan Findings [scan_finding] table.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the current source to an incident record in a script only check. For more information, see Create a script only check.

```javascript
(function(engine) {
    var gr = new GlideRecord('incident');
    gr.get('2f99f330730210100a5310c92bf6a798');
    engine.finding.setCurrentSource(gr);
    engine.finding.increment();
})(engine);
```

### FormInfoHeader - Global

FormInfoHeader allows you to add an HTML message as a form info message. The `addMessage` function is commonly used in record producers. The FormInfoHeader class is available to server-side script.

### FormInfoHeader - addMessage(String message)

Adds an HTML message to the form header, where form info messages are displayed.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>A message that may include HTML tags.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
gs.include("FormInfoHeader");
var fi = new FormInfoHeader();
fi.addMessage('This incident was opened on your behalf<br/>
The IT department will contact you for further information or when the incident is
resolved');
```

FlowAPI - Scoped, Global

Use FlowAPI methods to execute actions, flows, or subflows in server-side scripts using either blocking or non-blocking methods.

Access FlowAPI methods in global and scoped scripts using the sn_fd.FlowAPI object. Create calls to your flows using the Code Snippet action in Flow Designer, or use the methods detailed here to update scripts manually.

**Note:** In domain separated instances, flows, subflows, and actions triggered by this API run in the domain of the user who started the script. For example, if a user in the Acme domain starts a script that triggers a flow, the flow runs in the Acme domain and can only access Acme data, even if the flow runs as the System User.

**Note:** To optimize instance performance, avoid calling these methods from a script for an asynchronous business rule. Instead, create a scheduled job record within the Flow Designer UI.

FlowAPI - cancel(String contextId, String reason)

Cancels a paused or running flow, subflow, or action.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextId</td>
<td>String</td>
<td>Sys_id of the execution details record for the flow, subflow, or action. Access the execution details by navigating to the Flow Executions tab in Flow Designer, or pass the sys_id of the context record returned by the startFlow(), startSubflow(), or startAction() methods.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example uses the return value of the `startFlow()` method to cancel any long-running flows.

```javascript
(function() {
  var now_GR = new GlideRecord('incident');
  now_GR.addQuery('number', 'INC0000050');
  now_GR.query();
  now_GR.next();

  try {
    var inputs = {};
    inputs['current'] = now_GR; // GlideRecord of table:
    inputs['table_name'] = 'incident';

    // Starts the flow asynchronously.
    var contextId = sn_fd.FlowAPI.startFlow('global.myFlow', inputs);
  }
  catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
  }
})();

// Call the cancel() method using the context Id returned from the startFlow() method
sn_fd.FlowAPI.cancel(contextId, 'Flow took too long to execute.');
```

This example cancels any flows named Test Flow.
var now_GR = new GlideRecord("sys_flow_context");
now_GR.addQuery("name", "Test Flow");
now_GR.query();

while (now_GR.next()) {
    sn_fd.FlowAPI.cancel(now_GR.getUniqueValue(), 'Canceling Test Flows');
}

**FlowAPI - executeAction(String name, Map inputs, Number timeout)**

Runs an action from a server-side script synchronously.

Execute an action from within a business rule, script include, or any other server-side script. Actions run using this method run synchronously, so the method has access to outputs created by the action. Use `startAction` to run an action asynchronously.

ℹ️ **Note:** This API is replaced by `ScriptableFlowRunner`, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the `getRunner()` method in the `FlowAPI` class to return a `ScriptableFlowRunner` object and use the associated methods. Use the `ScriptableFlowRunner` methods if you need to support domain separation.

ℹ️ **Note:** This method runs the action as the user who initiates the session.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The scope and name of the action to be executed, for example <code>global.action_name</code>.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define action inputs. Use the input name, not the input label.</td>
</tr>
<tr>
<td>timeout</td>
<td>Number</td>
<td>Optional. Timeout in milliseconds. This value overrides the 30 second default timeout specified by the <a href="https://service-now.com/">com.glide.hub.flow_api.default_execution_time</a> system property. After the timeout expires, an exception is thrown.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The action outputs.</td>
</tr>
</tbody>
</table>
In this example, the script uses `sn_fd.FlowAPI.executeAction` to execute an action called `actionforpassword2test` in the global scope. A variable called `inputs` contains the inputs for the action. In this case, a name and password. The outputs for the action are stored in the `outputs` variable, which in this case, is an encrypted password object. The code is wrapped in a try/catch statement to capture any errors that might occur when the flow executes.

```javascript
(function() {
    try {
        var inputs = {};
        inputs['name'] = ; // String
        inputs['password2'] = ; // Password (2 Way Encrypted)

        // Execute Synchronously: Run in foreground. Code snippet has access to outputs.
        // var timeout = ; //timeout in ms
        //sn_fd.FlowAPI.executeAction('global.actionforpassword2test', inputs, timeout)
        var outputs = sn_fd.FlowAPI.executeAction('global.actionforpassword2test', inputs);

        // Get Outputs:
        // Note: outputs can only be retrieved when executing synchronously.
        var output = outputs['output']; // Password (2 Way Encrypted)
    } catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    }
})();
```

**FlowAPI - executeActionQuick(String name, Map inputs, Number timeout)**

Run an action from a server-side script synchronously from the current user session without creating execution details or other related records. Improve performance by eliminating record-keeping overhead. Use this API to increase the speed of high-volume processing, for example multiple executions per second, in a production environment.

⚠️ **Note:** This API is replaced by `ScriptableFlowRunner`, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the `getRunner()` method in the `FlowAPI` class to return a `ScriptableFlowRunner` object and use the associated methods. Use the `ScriptableFlowRunner` methods if you need to support domain separation.
**Note:** Execution details and context records are not created, regardless of Flow Designer settings.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Scope and internal name of the action to execute. For example, <code>global.action_name</code>. Locate the Internal name field in the list of Flow Designer actions.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define action inputs. You can find the available action inputs and required data types under Inputs in the action outline. Use the input name, not the input label. For example, <code>{&quot;table&quot;:&quot;incident&quot;,&quot;sys_id&quot;:&quot;a39d8e3cf0212300964feefe80ff0ed&quot;}</code>.</td>
</tr>
<tr>
<td>timeout</td>
<td>Number</td>
<td>Optional. Timeout in milliseconds. This value overrides the 30 second default timeout specified by the <code>com.glide.hub.flow_api.default_execution_time</code> system property. After the timeout expires, an exception is thrown.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing outputs defined by the action. You can find the outputs for the action under Outputs in the action outline.</td>
</tr>
</tbody>
</table>

```javascript
(function() {
  try {

    var grIncident = new GlideRecord('incident');
    grIncident.get('57af7aec73d423002728660c4cf6a71c');

    var inputs = {};
    inputs['variable'] = grIncident;

    var outputs = sn_fd.FlowAPI.executeActionQuick('global.update_record_test', inputs);

    // Get Outputs:
    // Note: outputs can only be retrieved when executing synchronously.
    var output1 = outputs['output1'];

  } catch (ex) {
  }
})
```
var message = ex.getMessage();
    gs.error(message);
}

FlowAPI - executeDataStreamAction(String name, Map inputs, Number timeout)
Runs a Data Stream action synchronously from a server-side script and returns a ScriptableDataStream object.

For more information about Data Stream actions, see Data Stream actions and pagination.

Note: This API is replaced by ScriptableFlowRunner, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the getRunner() method in the FlowAPI class to return a ScriptableFlowRunner object and use the associated methods. Use the ScriptableFlowRunner methods if you need to support domain separation.

Note: Always wrap data stream logic in a try/catch block to catch errors. Always include a finally statement that ends with the close() method from the ScriptableDataStream class to close the data stream and prevent performance issues.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The scope and name of the Data Stream action to execute. For example, global.data_stream_action_name.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define action inputs. Use the input name, not the input label. If the action does not have any inputs, do not include this parameter.</td>
</tr>
<tr>
<td>timeout</td>
<td>Number</td>
<td>Optional. Amount of time before the action times out. After the timeout expires, an exception is thrown. The timeout only applies to the executeDataStreamAction method, not to methods in the ScriptableDataStream class. Default: 30000, specified by the com.glide.hub.flow_api.default_execution_time system property Unit: Milliseconds</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableDataStream</td>
<td>An object used to iterate through items in the data stream. Use the methods in the ScriptableDataStream class to interact with this object. See ScriptableDataStream API.</td>
</tr>
</tbody>
</table>

This example creates an incident record for each item returned in the data stream.

```javascript
(function() {

  try {

    // Execute Data Stream Action.
    var stream = sn_fd.FlowAPI.executeDataStreamAction('x_my_scope.data_stream_name');

    // Process each item in the data stream
    while (stream.hasNext()) {

      // Get a single item from the data stream.
      var item = stream.next();

      // Use the item.
      var now_GR = new GlideRecord('incident');
      now_GR.setValue('number',item.id);
      now_GR.setValue('short_description',item.name);
      now_GR.insert();

      // By default, this code snippet will terminate after 10 items.
      // Remove or modify this limit after testing your code.
      if (stream.getItemIndex() >= 9) {
        break;
      }
    }
    catch (ex) {
      var message = ex.getMessage();
      gs.error(message);
    }
  }
}
```
FlowAPI - executeFlow(String name, Map inputs, Number timeout)

Runs a flow from a server-side script synchronously.

Execute a flow from within a business rule, script include, or any other server-side script. Flows run using this method run synchronously. Use startFlow to run a flow asynchronously.

⚠️ **Note:** This API is replaced by ScriptableFlowRunner, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the getRunner() method in the FlowAPI class to return a ScriptableFlowRunner object and use the associated methods. Use the ScriptableFlowRunner methods if you need to support domain separation.

⚠️ **Note:** This method runs the flow as the user specified in flow properties.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The scope and name of the flow to be executed, for example global.flow_name.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define trigger inputs. Use the input name, not the input label.</td>
</tr>
<tr>
<td>timeout</td>
<td>Number</td>
<td>Optional. Timeout in milliseconds. This value overrides the 30 second default timeout specified by the com.glide.hub.flow_api.default_execution_time system property. After the timeout expires, an exception is thrown.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Normal operation produces no return value.</td>
</tr>
<tr>
<td>Exception</td>
<td>The API throws an exception when a flow called synchronously pauses. The current execution is in the waiting state. In most cases, the exception is removed when the flow resumes. However, the API cannot resume a flow that has been sent to a MID Server.</td>
</tr>
</tbody>
</table>

This example uses sn_fd.FlowAPI.executeFlow to execute a global flow called test_flow. This flow is normally triggered when a record on the incident table is
updated. Because you are activating the flow from a script, you must provide this information. The code creates an inputs variable that contains the current record, and the table for the record. The code is wrapped in a try/catch statement to capture any errors that might occur when the flow executes.

```javascript
(function() {
    try {
        var inputs = {};
        inputs['current'] = ; // GlideRecord of table:
        inputs['table_name'] = 'incident';
        // Execute Synchronously: Run in foreground.
        // var timeout = ; //timeout in ms
        // sn_fd.FlowAPI.executeFlow('global.test_flow', inputs, timeout)
        sn_fd.FlowAPI.executeFlow('global.test_flow', inputs);
    } catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    }
})();
```

**FlowAPI - executeFlowQuick(String name, Map inputs, Number timeout)**

Runs a flow, subflow, or action from a server-side script synchronously or asynchronously without creating execution details or other related records. Improves performance by eliminating record-keeping overhead. Use this API to increase the speed of high-volume processing, for example multiple executions per second, in a production environment.

This method runs the flow as the user who initiates the session. Setting the flow to run as the system user, or impersonating a user, is not supported. Flows that include wait conditions, for instance the Wait for a duration of time flow logic or Ask For Approval action, are not supported.

**Note:** This API is replaced by ScriptableFlowRunner, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the getRunner() method in the FlowAPI class to return a ScriptableFlowRunner object and use the associated methods. Use the ScriptableFlowRunner methods if you need to support domain separation.

**Note:** Execution details and context records are not created, regardless of Flow Designer settings.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Scope and internal name of the flow to execute. For example, global.flow_name. Locate the <strong>Internal name</strong> field in the list of Flow Designer flows.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define trigger inputs. You can find the available trigger inputs and required data types in the Trigger section of the flow. Use the input name, not the input label. For example, { 'table': 'incident', 'sys_id': 'a39d8e3cf0212300964feefef80ff0ed' }.</td>
</tr>
<tr>
<td>timeout</td>
<td>Number</td>
<td>Optional. Timeout in milliseconds. This value overrides the 30 second default timeout specified by the com.glide.hub.flow_api.default_execution_time system property. After the timeout expires, an exception is thrown.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
(function() {

  try {
    var grIncident = new GlideRecord('incident');
    grIncident.get('ed92e8d173d023002723002728660c4cf6a7bc');

    var inputs = {};
    inputs['current'] = grIncident;
    inputs['table_name'] = 'incident';

    sn_fd.FlowAPI.executeFlowQuick('global.test_quick_flow', inputs);

  } catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
  }
})();
```
FlowAPI - executeSubflow(String name, Map inputs, Number timeout)

Runs an subflow from a server-side script synchronously.

Execute a subflow from within a business rule, script include, or any other server-side script. Subflows run using this method run synchronously. Use startSubflow to run an subflow asynchronously.

⚠️ Note: This API is replaced by ScriptableFlowRunner, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the getRunner() method in the FlowAPI class to return a ScriptableFlowRunner object and use the associated methods. Use the ScriptableFlowRunner methods if you need to support domain separation.

⚠️ Note: This method runs the flow as the user specified in flow properties.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The scope and name of the subflow to be executed, for example global.subflow_name.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define subflow inputs. Use the input name, not the input label.</td>
</tr>
<tr>
<td>timeout</td>
<td>Number</td>
<td>Optional. Timeout in milliseconds. This value overrides the 30 second default timeout specified by the com.glide.hub.flow_api.default_execution_time system property. After the timeout expires, an exception is thrown.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing the subflow outputs.</td>
</tr>
<tr>
<td>Exception</td>
<td>The API throws an exception when a flow called synchronously pauses. The current execution is in the waiting state. In most cases, the exception is removed when the flow resumes. However, the API cannot resume a flow that has been sent to a MID Server.</td>
</tr>
</tbody>
</table>

In this example, the script uses sn_fd.FlowAPI.executeSubflow to execute an subflow called subflowTest in the global scope. A variable called inputs contains the inputs for the subflow. In this case, a name and password. The code is wrapped in a try/catch statement to capture any errors that might occur when the flow executes.
(function() {
    try {
        var inputs = {};  
        inputs['name'] = ; // String
        inputs['password2'] = ; // Password (2 Way Encrypted)

        // Execute Synchronously: Run in foreground.
        // var timeout = ; //timeout in ms
        // sn_fd.FlowAPI.executeSubflow('global.subflowTest', inputs, timeout)
        var outputs = sn_fd.FlowAPI.executeSubflow('global.subflowTest', inputs);
    } catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    }
})() ;

**FlowAPI - executeSubflowQuick(String name, Map inputs, Number timeout)**

Run a subflow from a server-side script synchronously from the current user session without creating execution details or other related records. Improve performance by eliminating record-keeping overhead. Use this API to increase the speed of high-volume processing, for example multiple executions per second, in a production environment.

This method runs the subflow as the user who initiates the session. Setting the subflow to run as the System User, or impersonating a user, is not applied. Subflows that include wait conditions, for instance the Wait for a duration of time flow logic or Ask For Approval action, are not supported.

⚠️ **Note:** This API is replaced by ScriptableFlowRunner, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the getRunner() method in the FlowAPI class to return a ScriptableFlowRunner object and use the associated methods. Use the ScriptableFlowRunner methods if you need to support domain separation.

⚠️ **Note:** Execution details and context records are not created, regardless of Flow Designer settings.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Scope and internal name of the subflow to execute. For example, <code>global.subflow_name</code>. Locate the <strong>Internal name</strong> field in the list of Flow Designer subflows.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define subflow inputs. You can find the available subflow inputs and required data types under Inputs in the subflow. Use the input name, not the input label. For example, <code>{'table': 'incident', 'sys_id': 'a39d8e3cf02130964f3f3f80ff0ed'}</code>.</td>
</tr>
<tr>
<td>timeout</td>
<td>Number</td>
<td>Optional. Timeout in milliseconds. This value overrides the 30 second default timeout specified by the <code>com.glide.hub.flow_api.default_execution_time</code> system property. After the timeout expires, an exception is thrown.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing outputs defined by the subflow. You can find the outputs for the subflow under <strong>Subflow Inputs &amp; Outputs</strong> in the subflow outline.</td>
</tr>
</tbody>
</table>

{(function() {

    try {
        var grIncident = new GlideRecord('incident');
        grIncident.get('57af7aec73d423002728660c4cf6a71c');

        var inputs = {};
        inputs['variable'] = grIncident;

        var outputs = sn_fd.FlowAPI.executeSubflowQuick('global.test_quick_run_subflow', inputs);

        // Get Outputs:
        // Note: outputs can only be retrieved when executing synchronously.
        var output1 = outputs['output1'];

    } catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    }
})()}

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
FlowAPI - getOutputs(String contextId)

Returns the outputs of a completed action or subflow.

You can use the return values from either the startAction() or startSubflow() methods as the contextId parameter.

**Note:** This API is replaced by ScriptableFlowRunner, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the getRunner() method in the FlowAPI class to return a ScriptableFlowRunner object and use the associated methods. Use the ScriptableFlowRunner methods if you need to support domain separation.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextId</td>
<td>String</td>
<td>The sys_id of the action or subflow whose outputs you want to get.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing the action or subflow outputs.</td>
</tr>
</tbody>
</table>

This example starts a global subflow called test_subflow, waits for the flow to finish, and then gets its output values. Normally, a single script does not both run a flow and then get its output values. Since a flow may not complete before the getOutputs() call, this example uses a wait time. Typically, either another script or Flow Designer would have already run the flow. The code is wrapped in a try/catch statement to capture any errors that might occur when the flow executes.

```javascript
(function() {
    try {
        // Gather inputs to call flow
        var inputs = {};
        inputs['ah_task'] = myTaskRecord; // GlideRecord of table: task
        inputs['ah_comment'] = 'Test Comment'; // String
```
// Call flow
var contextId = sn_fd.FlowAPI.startSubflow('global.test_subflow', inputs);

// Wait for the flow to finish running

// Get flow outputs
var outputs = sn_fd.FlowAPI.getOutputs(contextId);
var output1 = outputs['output1'];
return output1;

} catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
}
})

FlowAPI - hasApprovals(String scopedFlowName)
Checks if a flow within a give scope contains any Ask for Approval actions.

The hasApprovals() method determines if a flow within a given scope contains any Ask for Approval actions. This method also checks if any Ask for Approval actions within the flow are nested under If flow logic blocks. For more information, see Ask for Approval actions.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopedFlowName</td>
<td>String</td>
<td>Scope and internal name of the flow to execute. For example, global.flow_name. Locate the Internal name field in the list of Flow Designer flows.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Returns one of the following string values:</td>
</tr>
<tr>
<td></td>
<td>• ALWAYS: The flow contains an Ask for Approval action that isn’t nested within a conditional If flow logic block.</td>
</tr>
<tr>
<td></td>
<td>• CONDITIONALLY: The flow contains an Ask for Approval action that is nested within a conditional If flow logic block.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• NO: The flow doesn't contain any Ask for Approval actions.</td>
<td></td>
</tr>
<tr>
<td>• UNKNOWN: There was a compiler error, and the system can't determine whether the flow contains any Ask for Approval actions.</td>
<td></td>
</tr>
</tbody>
</table>

The following example checks if the `example_flow` flow within the `global` scope contains any Ask for Approval actions and logs the result.

```javascript
(function() {
  try {
    var result = sn_fd.FlowAPI.hasApprovals('global.example_flow');
    gs.log('Result: ' + result);
  }

  catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
  }
})();
```

**FlowAPI - setEncryptedOutput(String password)**

Builds password2 values inside a script step.

Identify an encrypted password2 value returned from a GlideRecord, enabling the system to display the value as a masked password rather than an encrypted string.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>password</td>
<td>String</td>
<td>Encrypted password2 value.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Encrypted password2 value, recognised by the engine as a password value.</td>
</tr>
</tbody>
</table>

This example returns value of a password2 field from a user record and stores it in a variable. This variable is passed into the setEncryptedOutput method, which is called using `sn_fd.GlideActionUtil.setEncryptedOutput`. The instance recognises the returned value as a password.

```javascript
{function execute(inputs, outputs) {
  // ...code...
  var now_GR = new GlideRecord('sys_user');
  now_GR.addQuery('first_name', 'Abel');
  now_GR.query();
  now_GR.next();
  var pwVal = now_GR.getValue('pw2');
  outputs['usedSetEncrypted'] = sn_fd.GlideActionUtil.setEncryptedOutput(pwVal);
  outputs['justSetDirectly'] = pwVal;
})(inputs, outputs);
```

**FlowAPI - startAction(String name, Map inputs)**

Runs an action from a server-side script asynchronously.

Execute an action from within a business rule, script include, or any other server-side script. Actions run using this method run asynchronously, so scripts using this method do not have access to any outputs created by the action. Use `executeAction` to run an action synchronously and access the outputs it generates.

ℹ️ **Note:** This API is replaced by `ScriptableFlowRunner`, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the `getRunner()` method in the `FlowAPI` class to return a `ScriptableFlowRunner` object and use the associated methods. Use the `ScriptableFlowRunner` methods if you need to support domain separation.

ℹ️ **Note:** This method runs the action as the user who initiates the session.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The scope and name of the action to be executed, for example <code>global.action_name</code>.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define action inputs. Use the input name, not the input label.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys Id of the context record for the action. Access the context record by navigating to the Flow Executions tab in Flow Designer, selecting a flow execution, and clicking <strong>Open Context Record</strong>.</td>
</tr>
</tbody>
</table>

This example uses `sn_fd.FlowAPI.startAction` to execute an action called `add_comment` in the `sn_itsm_spoke` scope. The inputs object contains a target record and a comment to add to that record. The code is wrapped in a try/catch statement to capture any errors that might occur when the flow executes.

```javascript
(function() {
    try {
        var inputs = {};
        inputs['ah_task'] = myTaskRecord; // GlideRecord of table: task
        inputs['ah_comment'] = 'Test Comment'; // String

        var contextId = sn_fd.FlowAPI.startAction('sn_itsm_spoke.add_comment', inputs);
    } catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    }
})();
```

#### FlowAPI - startActionQuick(String name, Map inputs)

Runs an action from a server-side script asynchronously without creating execution details or other related records. Improve performance by eliminating record-keeping overhead. Use this API to increase the speed of high-volume processing, for example multiple executions per second, in a production environment.
Note: This API is replaced by ScriptableFlowRunner, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the getRunner() method in the FlowAPI class to return a ScriptableFlowRunner object and use the associated methods. Use the ScriptableFlowRunner methods if you need to support domain separation.

Note: Execution details and context records are not created, regardless of Flow Designer settings.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Scope and internal name of the action to execute. For example, global.action_name. Locate the Internal name field in the list of Flow Designer actions.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define action inputs. You can find the available action inputs and required data types under Inputs in the action outline. Use the input name, not the input label. For example, {'table':'incident','sys_id':'a39d8e3cf0212300964feefe80ff0ed'}.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
(function() {

try {
   var grIncident = new GlideRecord('incident');
   grIncident.get('57af7aec73d423002728660c4cf6a71c');

   var inputs = {};
   inputs['variable'] = grIncident;

   sn_fd.FlowAPI.startActionQuick('global.update_record_test', inputs);

} catch (ex) {
   var message = ex.getMessage();
   gs.error(message);
}
```
FlowAPI - startFlow(String name, Map inputs)
Runs a flow from a server-side script.

Execute a flow from within a business rule, script include, or any other server-side script. Flows executed with this method run asynchronously.

**Note:** This API is replaced by ScriptableFlowRunner, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the getRunner() method in the FlowAPI class to return a ScriptableFlowRunner object and use the associated methods. Use the ScriptableFlowRunner methods if you need to support domain separation.

**Note:** This method runs the flow as the user specified in flow properties.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The scope and name of the flow to be executed, for example global.flow_name.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define trigger inputs. Use the input name, not the input label.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys Id of the context record for the flow. Access the context record by navigating to the Flow Executions tab in Flow Designer, selecting a flow execution, and clicking <strong>Open Context Record</strong>.</td>
</tr>
</tbody>
</table>

This example uses sn_fd.FlowAPI.startFlow to execute a global flow called test_flow. The code creates an inputs variable that contains inputs required by the flow. In this case, the current record and the table for the record. The code is wrapped in a try/catch statement to capture any errors that might occur when the flow executes.

```javascript
(function() {
    var now_GR = new GlideRecord('incident');
    now_GR.addQuery('number', 'INC0009009');
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
```javascript
now_GR.query();
now_GR.next();

try {
    var inputs = {};
    inputs['current'] = now_GR; // GlideRecord of table: Incident
    inputs['table_name'] = 'incident';

    var contextId = sn_fd.FlowAPI.startFlow('global.test_flow', inputs);
    }
    catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    }
})();
```

**FlowAPI - startFlowQuick(String name, Map inputs)**

Runs a flow from a server-side script asynchronously without creating execution details or other related records. Improve performance by eliminating record-keeping overhead. Use this API to increase the speed of high-volume processing, for example multiple executions per second, in a production environment.

This method runs the flow as the System User. Setting the flow to run as the user who initiates the session, or impersonating a user, is not applied. Flows that include wait conditions, for instance the Wait for a duration of time flow logic or Ask For Approval action, are not supported.

ℹ️ **Note:** This API is replaced by `ScriptableFlowRunner`, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the `getRunner()` method in the `FlowAPI` class to return a `ScriptableFlowRunner` object and use the associated methods. Use the `ScriptableFlowRunner` methods if you need to support domain separation.

ℹ️ **Note:** Execution details and context records are not created, regardless of Flow Designer settings.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Scope and internal name of the flow to execute. For example, <code>global.flow_name</code>. Locate the <strong>Internal name</strong> field in the list of Flow Designer flows.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define trigger inputs. You can find the available trigger inputs and required data types in the Trigger section of the flow. Use the input name, not the input label. For example,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
(function() {
    try {
        var grIncident = new GlideRecord('incident');
        grIncident.get('ed92e8d173d023002728660c4cf6a7bc');

        var inputs = {};
        inputs['current'] = grIncident;
        inputs['table_name'] = 'incident';

        sn_fd.FlowAPI.startFlowQuick('global.test_quick_flow', inputs);
    } catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    }
})();
```

**FlowAPI - startSubflow(String name, Map input)**

Runs a subflow from a server-side script.

Execute a subflow from within a business rule, script include, or any other server-side script. Subflows run using this method run asynchronously.
that include this method do not have access to outputs created by the flow. Use `executeSubflow` to run a subflow synchronously and access the outputs it generates.

**Note:** This API is replaced by `ScriptableFlowRunner`, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the `getRunner()` method in the `FlowAPI` class to return a `ScriptableFlowRunner` object and use the associated methods. Use the `ScriptableFlowRunner` methods if you need to support domain separation.

**Note:** This method runs the flow as the user specified in flow properties.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The scope and name of the subflow to be executed, for example <code>global.subflow_name</code>.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define subflow inputs. Use the input name, not the input label.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys Id of the context record for the subflow. Access the context record by navigating to the Flow Executions tab in Flow Designer, selecting a flow execution, and clicking <strong>Open Context Record</strong>.</td>
</tr>
</tbody>
</table>

This example uses `sn_fd.FlowAPI.startSubflow` to execute a global flow called `test_subflow`. The code is wrapped in a try/catch statement to capture any errors that might occur when the flow executes.

```javascript
(function() {
    try {
        var inputs = {};
        inputs['ah_task'] = myTaskRecord; // GlideRecord of table: task
        inputs['ah_comment'] = 'Test Comment'; // String

        var contextId = sn_fd.FlowAPI.startSubflow('global.test_subflow', inputs);
    }
    catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    }

```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
FlowAPI - startSubflowQuick(String name, Map inputs)

Runs a subflow from a server-side script asynchronously without creating execution details or other related records. Improve performance by eliminating record-keeping overhead. Use this API to increase the speed of high-volume processing, for example multiple executions per second, in a production environment.

This method runs the subflow as the System User. Setting the subflow to run as the user who initiates the session, or impersonating a user, is not applied. Subflows that include wait conditions, for instance the Wait for a duration of time flow logic or Ask For Approval action, are not supported.

⚠️ **Note:** This API is replaced by ScriptableFlowRunner, which deprecates the existing methods used to build objects and execute Flow Designer flows and actions. Use the getRunner() method in the FlowAPI class to return a ScriptableFlowRunner object and use the associated methods. Use the ScriptableFlowRunner methods if you need to support domain separation.

⚠️ **Note:** Execution details and context records are not created, regardless of Flow Designer settings.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Scope and internal name of the subflow to execute. For example, <code>global.subflow_name</code>. Locate the <strong>Internal name</strong> field in the list of Flow Designer subflows.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define subflow inputs. You can find the available subflow inputs and required data types under Inputs in the subflow. Use the input name, not the input label. For example, <code>{'table': 'incident', 'sys_id': 'a239d8e3cf0212300964feeeefe0ff0ed'}</code>.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
(function() {

try {

    var grIncident = new GlideRecord('incident');
    grIncident.get('57af7aec73d423002728660c4cf6a71c');

    var inputs = {};
    inputs['variable'] = grIncident;

    sn_fd.FlowAPI.startSubflowQuick('global.test_quick_run_subflow', inputs);

} catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
}
})();

FlowAPI - getRunner()

Returns a ScriptableFlowRunner builder object for a flow or action that you want to run.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableFlowRunner</td>
<td>Builder object used to run a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

This example runs a flow synchronously.

(function() {
    try {

        var inputs = {};

...
inputs['sys_id'] = '57af7aec73d423002728660c4cf6a71c'; // Pass the record’s sys_id in as input.

var result = sn_fd.FlowAPI.getRunner() // Create a ScriptableFlowRunner builder object.
    .action('global.markapproved') // Run the global scope action named markapproved.
    .inForeground()
    .inDomain('TOP/ACME') // Run the action from the TOP/ACME domain.
    .withInputs(inputs)
    .run(); // Run the action and return a FlowRunnerResult object.

var contextId = result.getContextId(); // Retrieve the context ID from the result
var dateRun = result.getDate();
var domainUsed = result.getDomainId(); // Retrieve the Domain ID from the result.
var flowName = result.getFlowObjectName();
var flowObjectType = result.getFlowObjectType();

var outputs = result.getOutputs(); // Retrieve any outputs from the action execution.
var newApprovalStatus = outputs['approval']; // Echo back the approval status for verification.

} catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
}

());

---

**FlowScriptAPI - Scoped**

A FlowScriptAPI object allows you to access Flow Designer context details from script steps and inline scripts.

You cannot instantiate objects of this type. Objects of this type are created automatically and are accessible only in script steps and inline scripts.

**FlowScriptAPI - getContextID()**

Returns the context ID of the running flow.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_id of the running flow.</td>
</tr>
</tbody>
</table>

This example assumes you have created a flow containing a Log action. This inline script generates the content of the Log message.

```javascript
var contextId = FlowScriptAPI.getContextID();
return contextId;
```

Output:

4cead85c4da0010598d0c7c6bf73554

Flow - Scoped

Runs activated Flow Designer flows.

This API is deprecated and replaced by the FlowAPI - Scoped, Global and ScriptableFlowRunner - Scoped APIs.

The Flow API can only be used in server scripts.

Use the `sn_fd` namespace to access the Flow API.

Before interacting with a flow using the Flow API, you must first create and activate the flow in the Flow Designer interface. Because the Flow API only interacts with pre-built flows, there is no constructor for the class.

⚠️ Note: To optimize instance performance, avoid calling these methods from a script for an asynchronous business rule. Instead, create a scheduled job record within the Flow Designer UI.

 Scoped Flow - startAsync(String scopeName.flowName, Map flowInputs)

Ignores the trigger and runs an activated flow asynchronously.

Asynchronous calls are non-blocking, allowing the client to execute other code in the script without having to wait for the flow to complete.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopeName.flowName</td>
<td>String</td>
<td>The application scope for the flow and the internal name of the flow to run. If scopeName is not included, the scope of the user currently logged in is used. Retrieve the internal name of the flow using the <strong>Internal name</strong> column on the Flow Designer landing page.</td>
</tr>
<tr>
<td>flowInputs</td>
<td>Map</td>
<td>Name-value pairs in <code>&lt;String, Object&gt;</code> format that define record-based flow inputs. To call a flow with a record-based trigger, use the format:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>var flowInputs = {};&lt;br&gt;flowInputs['current'] = glideRecord;&lt;br&gt;flowInputs['table_name'] = glideRecord.getTableName();</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The GlideRecord object must be named 'current'. To call a flow with a Service Catalog trigger, use the format:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>var flowInputs = {};&lt;br&gt;flowInputs['request_item'] = glideRecord;&lt;br&gt;flowInputs['table_name'] = glideRecord.getTableName();</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The GlideRecord object must be named 'request_item'.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>PlanResponse object containing the following properties:&lt;br&gt;• contextId: sys_id of the execution details record for the executed flow. Access the execution details by navigating to the Flow Executions tab in Flow Designer and filtering by sys_id.</td>
</tr>
<tr>
<td></td>
<td>An exception occurs when the flow:</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Does not exist within the specified application scope, or the flow or scope name has been misspelled.</td>
<td></td>
</tr>
<tr>
<td>• Is not activated.</td>
<td></td>
</tr>
<tr>
<td>• Exceeds the recursion limit set by the <code>com.glide.hub.flow_engine.indirect_recursion_limit</code> system property. The default value is three.</td>
<td></td>
</tr>
</tbody>
</table>

//Example 1: Run a flow with a record-based trigger
(function startFlowAsync() {
  try {
    // You MUST fetch the GlideRecord that will be passed to the flow
    var glideRecordInput = new GlideRecord('sys_user');
    glideRecordInput.get('62826bf03710200044e0bfc8bcbe5df1');

    var flowInputs = {};
    flowInputs['current'] = glideRecordInput;
    flowInputs['table_name'] = glideRecordInput.getTableName();

    var result = sn_fd.Flow.startAsync('global.recordtriggeredflow', flowInputs);

    //The Sys ID of a flow execution (contextId)
    var contextId = result.contextId;
  }
  catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
  }
})();

//Example 2: Run a flow with a schedule-based trigger
(function startFlowAsync() {
  try {
    var result = sn_fd.Flow.startAsync('global.scheduletriggeredflow');

    //The Sys ID of a flow execution (contextId)
    var contextId = result.contextId;
  } catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
  }
})();
try {
  // You MUST fetch the GlideRecord that will be passed to the flow
  var glideRecordInput = new GlideRecord('sc_req_item');
  glideRecordInput.get(aeed229047801200e0ef563dbb9a71c2);

  var flowInputs = {};
  flowInputs['request_item'] = glideRecordInput;
  flowInputs['table_name'] = glideRecordInput.getTableName();

  var result = sn_fd.Flow.startAsync('global.catalogtriggeredflow', flowInputs);

  //The Sys ID of a flow execution (contextId)
  var contextId = result.contextId;
} catch (ex) {
  var message = ex.getMessage();
  gs.error(message);
}

//Example 4: Run a flow with a MetricBase trigger
(function startMetricBaseFlowAsync() {
  try {

    var oilLevelTriggerRecord = new GlideRecord('oil_levels');
    oilLevelTriggerRecord.get('a4b36222bc72113007b237f48cb97635f');

    var metricTriggerDefinition = new GlideRecord('sys_metric_trigger_definition');
    metricTriggerDefinition.get('21f2eae7c72113007b237f48cb976352');

    var event_time = oilLevelTriggerRecord.getValue('sys_created_on');

    //...
var level = 4;

var metricBaseFlowInputs = {};  
//The record that triggered the metric event
metricBaseFlowInputs['current'] = oilLevelTriggerRecord;
//The MetricBase Trigger Definition record
metricBaseFlowInputs['metric'] = metricTriggerDefinition;
//The time that the 'record' reached a specific metric event level and triggered this flow
metricBaseFlowInputs['event_time'] = event_time;
//The target event level to reach in order for a metric flow to trigger
metricBaseFlowInputs['level'] = level;

var result = sn_fd.Flow.startAsync('global.metricbasedtriggeredflow', metricBaseFlowInputs);

//The Sys ID of a flow execution (contextId)
var contextId = result.contextId;

} catch (ex) {
  var message = ex.getMessage();
  gs.error(message);
}

})();

**g_service_catalog - Client**

The **g_service_catalog** API enables you to access data in a multi-row variable set (MRVS) when a model is open.

This API is available in all environments, such as, Service Portal, Now Platform, Workspace, and Now® Mobile.

**g_service_catalog - getValue(String variableName)**

Returns the value of the specified field on the catalog item form when used in a client script on multi-row variable sets (MRVS).

Use this method when an MRVS modal is open for editing or creating and you want to modify data within the MRVS based on the value of a field on the parent catalog item form. For example, when you need to modify the contents of the cells within an MRVS based on a check box on the parent form. You can also use this method to access the data of other MRVS elements within the same parent form.
Note: This method can only be called from the parent object, such as

```
g_service_catlog.parent.getValue();
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td>variableName</td>
<td>String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Array of objects, with each object containing the name-value pairs of the requested variable.</td>
</tr>
<tr>
<td></td>
<td>For example: [&quot;city&quot;: &quot;San Diego&quot;, &quot;country&quot;: &quot;USA&quot;],</td>
</tr>
<tr>
<td></td>
<td>{&quot;city&quot;: &quot;New Delhi&quot;, &quot;country&quot;: &quot;India&quot;}, {&quot;city&quot;: &quot;Melbourne&quot;, &quot;country&quot;: &quot;Australia&quot;}]</td>
</tr>
</tbody>
</table>

In this example, a catalog item for blocking multiple IP addresses on a firewall has a variable `address_type` with two choices - IPv4 and IPv6. The MRVS has two variables (`ipv4_address` and `ipv6_address`) for the respective address types. If the Address type field on the parent form is set to IPv4, the field IPv6 address is hidden on the MRVS.

```
function onLoad() {
    if (g_service_catalog.parent.getValue("address_type") == "ipv4") {
        g_form.setValue("ipv4_address", "XX.XX.XX.XX");
        g_form.setVisible("ipv6_address", "false");
    }
}
```

**GenericUPSAlarmsReconciler - Global**
Concrete reconciler for generic UPS alarms.
Use this API for SNMP-related discovery.

**GenericUPSAlarmsReconciler - getReconciliationField()**
Returns the reconciliation field.
### GenericUPSAlarmsReconciler - getReconciliationKey()

Returns the reconciliation key.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The reconciliation field</td>
</tr>
</tbody>
</table>

### GenericUPSAlarmsReconciler - hasChanged()

Determines if the generic UPS Bypass information has changed.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the information has changed; otherwise, false.</td>
</tr>
</tbody>
</table>
GenericUPSAlarmsReconciler - readDatabaseFields()

Reads the database fields.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GenericUPSAlarmsReconciler - readDiscovered()

Reads the discovered information.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GenericUPSAlarmsReconciler - setDatabaseFields()

Sets the database fields.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GenericUPSBypassReconciler - Global**

Concrete reconciler for generic UPS bypasses.

Use this API for SNMP-related discovery.

**GenericUPSBypassReconciler - getReconciliationField()**

Returns the reconciliation field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The reconciliation field</td>
</tr>
</tbody>
</table>

**GenericUPSBypassReconciler - getReconciliationKey()**

Returns the reconciliation key.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The reconciliation key</td>
</tr>
</tbody>
</table>
**GenericUPSBypassReconciler - hasChanged()**

Determines if the generic UPS Bypass information has changed.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the information has changed; otherwise, false.</td>
</tr>
</tbody>
</table>

**GenericUPSBypassReconciler - readDatabaseFields()**

Reads the database fields.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GenericUPSBypassReconciler - readDiscovered()**

Reads the discovered information.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**GenericUPSReconciler - setDatabaseFields()**

Sets the database fields.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GenericUPSInputsReconciler - Global**

Concrete reconciler for generic UPS inputs.

Use this API for SNMP-related discovery.

**GenericUPSInputsReconciler - getReconciliationField()**

Returns the reconciliation field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The reconciliation field</td>
</tr>
</tbody>
</table>
GenericUPSInputsReconciler - getReconciliationKey()

Returns the reconciliation key.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The reconciliation key</td>
</tr>
</tbody>
</table>

GenericUPSInputsReconciler - hasChanged()

Determines if the generic UPS Bypass information has changed.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the information has changed; otherwise, false.</td>
</tr>
</tbody>
</table>

GenericUPSInputsReconciler - readDatabaseFields()

Reads the database fields.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### GenericUPSInputsReconciler - readDiscovered()
Reads the discovered information.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### GenericUPSInputsReconciler - setDatabaseFields()
Sets the database fields.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### GenericUPSOutputsReconciler - Global
Concrete reconciler for generic UPS outputs.
Use this API for SNMP-related discovery.
**GenericUPSOutputsReconciler - getReconciliationField()**

Returns the reconciliation field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The reconciliation field</td>
</tr>
</tbody>
</table>

**GenericUPSOutputsReconciler - getReconciliationKey()**

Returns the reconciliation key.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GenericUPSOutputsReconciler - hasChanged()**

Determines if the generic UPS bypass information has changed.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the information has changed; otherwise, false.</td>
</tr>
</tbody>
</table>

**GenericUPSOutputsReconciler - readDatabaseFields()**

Reads the database fields.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GenericUPSOutputsReconciler - readDiscovered()**

Reads the discovered information.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GenericUPSOutputsReconciler - setDatabaseFields()**

Sets the database fields.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GeniusResultAnswer - Scoped**

The `GeniusResultAnswer` API provides methods for creating an answer object in a Genius Result configuration. Answers created with this API define search requests or key-value pair maps that can populate Genius Result answer cards.

You can populate the answer object with query details retrieved from the context of the Genius Result configuration using the `GeniusResultContext` API.

The `GeniusResultAnswer` API methods you must invoke to create the answer object vary by Genius Result answer type:

- For the **Search** answer type, you must invoke the `setTable()` method and either the `setSearchPhrase()` method or the `addSearchPhrases()` method. All other methods are optional except for `addDataMap()`, which is not relevant for this use case.

- For the **Script** answer type, you must invoke the `addDataMap()` method. No other methods are relevant for this use case.

Use this API in Genius Result server-side scripts with the `sn ais` namespace identifier. For more information on scripting logic for Genius Results, see Create a new Genius Result configuration.

**GeniusResultAnswer – addDataMap(Object map)**

Adds a set of arbitrary key-value pair fields to a Genius Result **Script** answer. You can utilize any public Now Platform API to generate these fields.

ℹ️ **Note:** When constructing logic for a Genius Result with the **Script** answer type, you must invoke this method. If you do not invoke this method, the Genius Result answer script fails.

You can add multiple field maps to a single Genius Result answer by invoking this method more than once. Each field map populates a separate Genius Result answer card.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>map</td>
<td>Object</td>
<td>Defines an arbitrary set of key-value pair fields for addition to the Genius Result Script answer. All field values must be Strings.</td>
</tr>
</tbody>
</table>

```javascript
{
  "fieldName1": String,
  "fieldName2": String
}
```

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether invocation of the method succeeded. Valid values: • true: Invocation of the method succeeded. • false: Invocation of the method failed.</td>
</tr>
</tbody>
</table>

This Genius Result Script answer logic adds custom title and URL key-value pair fields to the answer object for display on a Genius Result answer card.

```javascript
function process(context) {
    var answer = new sn_ais.GeniusResultAnswer();

    var customFieldMap = {
        "title": "My custom Genius Result answer",
        "url": "https://example.com/myGeniusResultLink"
    }

    answer.addDataMap(customFieldMap);

    return answer;
}
```

**GeniusResultAnswer – addSearchPhrases(Array searchPhrases)**

Adds terms to the search query defined for a Genius Result Search answer.
**Note:** When constructing logic for a Genius Result with the **Search** answer type, you must invoke the `setTable()` method and either this method or the `setSearchPhrase()` method. If you do not invoke both of the required methods, the Genius Result answer search query fails.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>searchPhraseArray</code></td>
<td>Array</td>
<td>Array of arbitrary Strings to add as terms for the search query made by the Genius Result <strong>Search</strong> answer. Search treats these terms as if separated by the OR operator. Example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><img src="image" alt="Example" /></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether invocation of the method succeeded. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Invocation of the method succeeded.</td>
</tr>
<tr>
<td></td>
<td>• false: Invocation of the method failed.</td>
</tr>
</tbody>
</table>

This Genius Result **Search** answer logic function constructs a search query for records on the User `[sys_user]` table that contain the terms Beth or Anglin.

```javascript
function process(context) {
  var answer = new sn_ais.GeniusResultAnswer();

  answer.setTable('sys_user');
  answer.addSearchPhrases(['Beth', 'Anglin']);

  return answer;
}
```
GeniusResultAnswer – setEncodedQuery(String query)
Sets and encodes the search query for a Genius Result `Search` answer.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>String</td>
<td>Query to use for the search issued by the Genius Result <code>Search</code> answer.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether invocation of the method succeeded.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Invocation of the method succeeded.</td>
</tr>
<tr>
<td></td>
<td>• false: Invocation of the method failed.</td>
</tr>
</tbody>
</table>

This Genius Result `Search` answer logic constructs an encoded search query against the User `[sys_user]` table. The final query joins together multiple subqueries generated from entities detected in the user's original search query.

```javascript
function process(context) {
  var answer = new sn_ais.GeniusResultAnswer();

  answer.setTable('sys_user');
  answer.setSearchLimit(2);
  answer.spellCheck(false);
  var queries = [];
  var matchingSegments;

  var predictionResult = context.getPredictionResult();
  if (predictionResult && predictionResult.length == 1) {
    var detail = predictionResult[0];
    if (detail['result'] && detail['result']['entities']) {
      var entities = detail['result']['entities'];
      for (var i = 0; i < entities.length; i++) {
        if (entities[i]['name'].endsWith('personName')) {
          matchingSegments = entities[i]['matchingSegments'];
          for (var j = 0; j < matchingSegments.length; j++) {
            answer.addSearchPhrases([matchingSegments[j]['value']]);
          }
        } else if (entities[i]['name'].endsWith('cityLocation')) {
        }
      }
    }
  }
}
```
matchingSegments = entities[i]['matchingSegments'];
if (matchingSegments && matchingSegments.length > 0)
    queries.push('location.nameLIKE' + matchingSegments[0]['value']);
} else if (entities[i]['name'].endsWith('departmentName')) {
    matchingSegments = entities[i]['matchingSegments'];
    if (matchingSegments && matchingSegments.length > 0)
        queries.push('department.nameLIKE' + matchingSegments[0]['value']);
} else if (entities[i]['name'].endsWith('jobTitle')) {
    matchingSegments = entities[i]['matchingSegments'];
    if (matchingSegments && matchingSegments.length > 0)
        queries.push('titleLIKE' + matchingSegments[0]['value']);
    }
}
}
}

if (queries.length > 0)
    answer.setEncodedQuery(queries.join('^^'));

return answer;

GeniusResultAnswer – setSearchLimit(Number searchLimit)

Limits the maximum number of search results returned for a Genius Result Search answer. By default, the search query returns up to three results.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>searchLimit</td>
<td>Number</td>
<td>Positive integer value for the maximum number of search results to return for a Genius Result Search answer. Values above 3 have no effect. Default: 3</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether invocation of the method succeeded. Valid values:</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• true: Invocation of the method succeeded.</td>
</tr>
<tr>
<td></td>
<td>• false: Invocation of the method failed.</td>
</tr>
</tbody>
</table>

This Genius Result **Search** answer logic function searches for records from the Catalog Item [sc_cat_item] table that contain the term laptop and returns a maximum of two results.

```javascript
function process(context) {
  var answer = new sn_ais.GeniusResultAnswer();

  answer.setTable('sc_cat_item');
  answer.setSearchPhrase('laptop');
  answer.setSearchLimit(2);

  return answer;
}
```

**GeniusResultAnswer – setSearchPhrase(String searchPhrase)**

Specifies terms for the search query defined for a Genius Result **Search** answer.

**Note:** When constructing logic for a Genius Result with the **Search** answer type, you must invoke the setTable() method and either this method or the addSearchPhrases() method. If you do not invoke both of the required methods, the Genius Result answer search query fails.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>searchPhraseString</td>
<td>String</td>
<td>Terms to use for the search query made by the Genius Result <strong>Search</strong> answer. If you want to retain the original search query terms submitted by the user, set this parameter to the String value returned by the GeniusResultContext.getOriginalSearchPhrase() method.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether invocation of the method succeeded.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Invocation of the method succeeded.</td>
</tr>
<tr>
<td></td>
<td>• false: Invocation of the method failed.</td>
</tr>
</tbody>
</table>

This Genius Result **Search** answer logic function constructs a search query for records from the User [sys_user] table that contain the term Anglin.

```javascript
function process(context) {
    var answer = new sn_ais.GeniusResultAnswer();

    answer.setTable('sys_user');
    answer.setSearchPhrase('Anglin');

    return answer;
}
```

**GeniusResultAnswer – setTable(String table)**

Limits the search query defined for a Genius Result **Search** answer to a specific table. The query only returns search results from the specified table as Genius Result answer cards.

**Note:** When constructing logic for a Genius Result with the **Search** answer type, you must invoke this method and either the `setSearchPhrase()` method or the `addSearchPhrases()` method. If you do not invoke both of the required methods, the Genius Result answer search query fails.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>Now Platform table name. Only indexed records on this table are included in results for the Genius Result <strong>Search</strong> answer.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether invocation of the method succeeded.</td>
</tr>
</tbody>
</table>

Valid values:

• true: Invocation of the method succeeded.
• false: Invocation of the method failed.

This Genius Result **Search** answer logic function constructs a search query for records from the User [sys_user] table that contain the term Anglin.

```javascript
function process(context) {
    var answer = new sn_ais.GeniusResultAnswer();
    answer.setTable('sys_user');
    answer.setSearchPhrase('Anglin');
    return answer;
}
```

**GeniusResultAnswer – spellCheck(Boolean spellCheck)**

Specifies whether to perform typo handling auto-correction (spell checking) for a search query defined in a Genius Result **Search** answer.

If you do not invoke this method for a Genius Result **Search** answer, the defined search query inherits the typo handling flag from the search query that activated the Genius Result configuration.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spellCheck</td>
<td>Boolean</td>
<td>Flag that indicates whether to perform typo handling auto-correction for the search query defined in a Genius Result <strong>Search</strong> answer.</td>
</tr>
</tbody>
</table>

Valid values:

• true: Perform typo handling auto-correction for the search query.
• false: Bypass typo handling auto-correction for the search query.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Default: true</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This Genius Result `Search` answer logic disables typo handling auto-correction for its search query against the User `sys_user` table.

```javascript
function process(context) {
    var answer = new sn_ais.GeniusResultAnswer();

    answer.setTable('sys_user');
    answer.setSearchLimit(2);
    answer.spellCheck(false);
    var queries = [];
    var matchingSegments;
    var predictionResult = context.getPredictionResult();
    if (predictionResult && predictionResult.length == 1) {
        var detail = predictionResult[0];
        if (detail['result'] && detail['result']['entities']) {
            var entities = detail['result']['entities'];
            for (var i = 0; i < entities.length; i++) {
                if (entities[i]['name'].endsWith('personName')) {
                    matchingSegments = entities[i]['matchingSegments'];
                    for (var j = 0; j < matchingSegments.length; j++)
                        answer.addSearchPhrases([matchingSegments[j]['value']]);
                } else if (entities[i]['name'].endsWith('cityLocation')) {
                    matchingSegments = entities[i]['matchingSegments'];
                    if (matchingSegments && matchingSegments.length > 0)
                        queries.push('location.nameLIKE' + matchingSegments[0]['value']);
                } else if (entities[i]['name'].endsWith('departmentName')) {
                    matchingSegments = entities[i]['matchingSegments'];
                    if (matchingSegments && matchingSegments.length > 0)
                        queries.push('department.nameLIKE' + matchingSegments[0]['value']);
                } else if (entities[i]['name'].endsWith('jobTitle')) {
                    matchingSegments = entities[i]['matchingSegments'];
                    if (matchingSegments && matchingSegments.length > 0)
                        queries.push('titleLIKE' + matchingSegments[0]['value']);
                }
            }
        }
    }
```
if (queries.length > 0)
    answer.setEncodedQuery(queries.join('^'));

return answer;
}

GeniusResultContext - Scoped

The GeniusResultContext API provides methods for retrieving search query information from the context of a Genius Result configuration.

You can use search query details retrieved with this API to populate Genius Result answer objects created with the GeniusResultAnswer API.

Use this API in Genius Result server-side scripts with the sn_ais namespace identifier. For more information on scripting logic for Genius Results, see Create a new Genius Result configuration.

GeniusResultContext – getOriginalSearchPhrase()

Retrieves the user's original search query terms from the search query. You can use these terms to populate Genius Result answers using GeniusResultAnswer API methods.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Original terms from the search query.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

This Genius Result Search answer logic function gets the search query's original terms and uses them as the query terms for a new snippet search.

```javascript
function process(context) {
    var answer = new sn_ais.GeniusResultAnswer();
}
var searchPhrase = context.getOriginalSearchPhrase();
answer.setSearchPhrase(searchPhrase);
answer.snippetSearch(true);

return answer;
}

GeniusResultContext – getPredictionResult()

Retrieves NLU model prediction results for the search query.

Use this method to retrieve the intent detected for the search query. You can populate Genius Result answers with details from the detected intent using GeniusResultAnswer API methods.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| [Array]| Array of objects in which each object represents an NLU model intent prediction result for the search query.  

Data type: Array

```javascript
[
{
  "intentName": "String",
  "result": {Object},
  "solutionName": "String",
}
]
```

[Array].intentName

Name of the NLU model intent detected in the search query.  

Data type: String

[Array].result

Object representing the NLU model intent prediction result for the search query.  

Data type: Object
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;result&quot;: {</td>
<td></td>
</tr>
<tr>
<td>&quot;application&quot;: {Object},</td>
<td></td>
</tr>
<tr>
<td>&quot;result&quot;: {Object},</td>
<td></td>
</tr>
<tr>
<td>&quot;solutionLabel&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;solutionName&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;solutionType&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;type&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;version&quot;: Number</td>
<td></td>
</tr>
<tr>
<td>]</td>
<td></td>
</tr>
</tbody>
</table>

[Array].result.application

Object containing metadata for the search application that produced the prediction result.

Data type: Object

"application": {
  "language": "String",
  "type": "String"
}

[Array].result.application.language

Language context for the Now Platform application that produced the prediction result.

Data type: String

[Array].result.application.type

Type of the Now Platform application that produced the prediction result.

Data type: String

[Array].result.result

Object including details for the NLU model intent prediction result.

Data type: Object

"result": {
  "entities": [Array],
  "intentName": "String",
  "intents": [Array],
  "nluModelName": "String",
  "score": Number
}

[Array].result.result.entities

Array of objects in which each object represents an NLU model intent entity detected in the search query.
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Data type: Array</strong></td>
</tr>
<tr>
<td></td>
<td>&quot;entities&quot;: [</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;matchingSegments&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;parts&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;score&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;startingPosition&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
</tbody>
</table>

**[Array].result.result.entities.matchingSegments**

Array of objects in which each object represents a matching segment of the detected NLU entity.

**Data type: Array**

```
"matchingSegments": [ |
|   |
|   "directMatch": Boolean |
|   "meta": {Object}, |
|   "value": "String" |
| ] |
```

**[Array].result.result.entities.matchingSegments.directMatch**

Flag that indicates whether the NLU entity matching segment represents an exact match for the entity value.

**[Array].result.result.entities.matchingSegments.meta**

Object containing key-value pair fields representing metadata for the NLU entity matching segment.

**Data type: Object**

```
"meta": { |
|   "augmentedBy": "String", |
|   "colMeta:first_name": "String", |
|   "colMeta:last_name": "String", |
|   "compositionStrategy": "String", |
|   "fieldName": "String", |
|   "searchPreProcessors": "String", |
|   "SUB_PHRASE.penalty": "String", |
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.

Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;tableName&quot;: &quot;String&quot;</td>
</tr>
</tbody>
</table>

[Array].result.result.entities.matchingSegments.value

String value of the NLU entity matching segment.
Data type: String

[Array].result.result.entities.name

Name for the detected NLU entity, in entity:[NLU_model].[intent].[entity_type] format.
Data type: String

[Array].result.result.entities.parts

Array of objects in which each object represents an NLU entity part found in the prediction result.
Data type: Array

"parts": [
  {
    "meta": {Object},
    "value": "String"
  }
]

[Array].result.result.entities.parts.meta

Object containing key-value pair fields representing metadata for the NLU entity part.
Data type: Object

"meta": {
  "augmentedBy": "String",
  "colMeta:name": "String",
  "compositionStrategy": "String",
  "fieldName": "String",
  "searchPreProcessors": "String",
  "SUB_PHRASE.penalty": "String",
  "tableName": "String"
}

[Array].result.result.entities.parts.value

String value of the NLU entity part.
Data type: String

[Array].result.result.entities.score

Numeric score from 0 through 1 indicating confidence for the detected NLU entity.
Data type: Number
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Array].result.result.entities.startingPosition</td>
<td>Numeric index for the character position in the search query at which the detected NLU entity begins. Data type: Number</td>
</tr>
<tr>
<td>[Array].result.result.entities.value</td>
<td>String value of the NLU detected in the search query. Data type: String</td>
</tr>
<tr>
<td>[Array].result.result.intentName</td>
<td>Name of the NLU model intent detected in the search query. Data type: String</td>
</tr>
<tr>
<td>[Array].result.result.intents</td>
<td>Array of objects in which each object represents a nested NLU model intent detected in the search query. Nested intent objects include the same parameter names and data types as the parent NLU model intent object. Data type: Array</td>
</tr>
<tr>
<td>[Array].result.result.nluModelName</td>
<td>Name of the NLU model that produced the prediction result. Data type: String</td>
</tr>
<tr>
<td>[Array].result.result.score</td>
<td>Numeric score from 0 through 1 indicating confidence for the NLU model prediction result. Data type: Number</td>
</tr>
<tr>
<td>[Array].result.solutionLabel</td>
<td>Label for the NLU model that produced the prediction result. Data type: String</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Array].result.solutionName</td>
<td>Name for the NLU model that produced the prediction result.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>[Array].result.solutionType</td>
<td>Type for the NLU model that produced the prediction result.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>[Array].result.type</td>
<td>Type of the prediction result.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>[Array].result.version</td>
<td>Version of the NLU model that produced the prediction result.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>[Array].solutionName</td>
<td>Name of the NLU model that produced the prediction result.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

This Genius Result Search answer logic function checks whether the search query includes an NLU model prediction result. When it finds a prediction result, it adds the matching segments from all detected catalogItem entities as search terms.

```javascript
function process(context) {
  var answer = new sn_ais.GeniusResultAnswer();

  answer.setTable('sc_cat_item');
  answer.setSearchLimit(2);
  var predictionResult = context.getPredictionResult();
  if (predictionResult && predictionResult.length == 1) {
    var detail = predictionResult[0];
    if (detail['result'] && detail['result']['entities']) {
      var entities = detail['result']['entities'];
      for (var i = 0; i < entities.length; i++) {
        if (entities[i]['name'].endsWith('catalogItem')) {
          var matchingSegments = entities[i]['matchingSegments'];
          for (var j = 0; j < matchingSegments.length; j++)
            answer.addSearchPhrases([matchingSegments[j]['value']]);
        }
      }
    }
  }
}
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
GlideAgentWorkspace (g_aw) - Client

The g_aw API enables a UI Action or client script to open a specified record in an Agent Workspace tab.

There is no constructor for the GlideAgentWorkspace class. Access GlideAgentWorkspace methods using the g_aw global object.

GlideAgentWorkspace - closeRecord()

Closes the currently open record, such as a form, in a subtab within Agent Workspace.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example saves the content of the tab and then closes it.

```javascript
function onClick(g_form) {
  g_form.save().then(function()
    g_aw.closeRecord();
  );
}
```

GlideAgentWorkspace - openRecord(String table, String sysId, Object params)

Opens a specified record, such as a form, in a subtab within Agent Workspace.
**Note:** This method is only available in the Agent Workspace client scripting environment or in a UI action on the workspace client script field.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>Name of the table that contains the record to open.</td>
</tr>
<tr>
<td>sysId</td>
<td>String</td>
<td>Sys ID of the record to open.</td>
</tr>
<tr>
<td>params</td>
<td>Object</td>
<td>Optional. Name/value pairs of the parameters to pass to the record.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>params.readOnlyForm</td>
<td>Boolean</td>
<td>Flag that indicates whether all fields on the opened record are read-only; regardless of the UI policy and ACLs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: All fields are read only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Fields adhere to the associated UI policy and ACLs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>params.defaultTab</td>
<td>String</td>
<td>Name of the initial tab to display in the workspace. You can only specify related items or related lists.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If not specified, the details tab appears unless <strong>hideDetails</strong> is set to true.</td>
</tr>
</tbody>
</table>
|                   |          | For more information on the method to use to obtain a related list name, see `getRelatedListNames()`.
| params.hideDetails | Boolean  | Flag that indicates whether to hide the details tab and the UI actions.     |
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true: Only the form header, all other tabs, and the first available tab appear on the form.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>false: Details tab and UI actions appear on the form.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default: false</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Open a sys_user record in a subtab.

```
g_aw.openRecord('sys_user', '62826bf03710200044e0bfc8bcbe5df1');
```

Open a record in a subtab where all fields are read-only.

```
g_aw.openRecord('sys_user', '62826bf03710200044e0bfc8bcbe5df1', {readOnlyForm: true});
```

Open a record in a subtab and go directly to the "Groups" related list.

```
g_aw.openRecord('sys_user', '62826bf03710200044e0bfc8bcbe5df1', {defaultTab: "sys_user_grmember.user"});
```

Open a record in a subtab but only show the form header and other tabs.

```
g_aw.openRecord('sys_user', '62826bf03710200044e0bfc8bcbe5df1', {hideDetails: true});
```

**GlideAggregate - Global**

GlideAggregate enables creating database aggregation queries.

The GlideAggregate class is an extension of GlideRecord and provides database aggregation (AVG, COUNT, MIN, MAX, STDDEV, SUM) queries. This functionality can be helpful when creating customized reports or in calculations for calculated fields. The GlideAggregate class works only on number fields.

When you use GlideAggregate on currency or price fields, you are working with the reference currency value. Be sure to convert the aggregate values to the user's session currency for display. Because the conversion rate between
the currency or price value (displayed value) and its reference currency value (aggregation value) might change, the result may not be what the user expects.

Note: When using an on-premise system, the database server time zone must be set to GMT/UTC for this class to work properly.

GlideAggregate - addAggregate(String agg, String name)

Adds an aggregate to a database query.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| agg   | String  | Name of an aggregate to include in a database query. Valid values:  
  • AVG  
  • COUNT  
  • MIN  
  • MAX  
  • STDDEV  
  • SUM  |
| name  | String  | Optional. Name of the field to group the results of the aggregation by. Default: Null |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add aggregates to a query on the category and software fields in the Incident [incident] table.

```javascript
var incidentGA = new GlideAggregate('incident');

incidentGA.addQuery('category', 'software');

incidentGA.setGroup(false);
```
incidentGA.addAggregate('COUNT', 'sys_mod_count');
incidentGA.addAggregate('SUM', 'sys_mod_count');
incidentGA.addAggregate('AVG', 'sys_mod_count');
incidentGA.addAggregate('MIN', 'sys_mod_count');
incidentGA.addAggregate('MAX', 'sys_mod_count');
incidentGA.addAggregate('STDDEV', 'sys_mod_count');

incidentGA.query();

if (incidentGA.next()) {
    gs.info('COUNT: ' + incidentGA.getAggregate('COUNT', 'sys_mod_count'));
    gs.info('SUM: ' + incidentGA.getAggregate('SUM', 'sys_mod_count'));
    gs.info('AVG: ' + incidentGA.getAggregate('AVG', 'sys_mod_count'));
    gs.info('MIN: ' + incidentGA.getAggregate('MIN', 'sys_mod_count'));
    gs.info('MAX: ' + incidentGA.getAggregate('MAX', 'sys_mod_count'));
    gs.info('STDDEV: ' + incidentGA.getAggregate('STDDEV', 'sys_mod_count'));
}

Output
COUNT: 13
SUM: 273
AVG: 21.0000
MIN: 3
MAX: 95
STDDEV: 32.7694

Scoped equivalent
To use the addAggregate() method in a scoped application, use the corresponding scoped method: Scoped GlideAggregate - addAggregate(String agg, String name).

GlideAggregate - addEncodedQuery(String query)
Adds an encoded query to the other queries that may have been set for this aggregate.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>String</td>
<td>Encoded query string to add to the aggregate.</td>
</tr>
</tbody>
</table>
var agg = new GlideAggregate('incident');
agg.addAggregate('count','category');
agg.orderByAggregate('count', 'category');
agg.orderBy('category');
agg.addQuery('opened_at', '>=', 'javascript:gs.monthsAgoStart(2)');
agg.addQuery('opened_at', '<=', 'javascript:gs.monthsAgoEnd(2)');
agg.query();
while (agg.next()) {
    var category = agg.category;
    var count = agg.getAggregate('count','category');
    var query = agg.getQuery();
    var agg2 = new GlideAggregate('incident');
    agg2.addAggregate('count','category');
    agg2.orderByAggregate('count', 'category');
    agg2.orderBy('category');
    agg2.addQuery('opened_at', '>=', 'javascript:gs.monthsAgoStart(3)');
    agg2.addQuery('opened_at', '<=', 'javascript:gs.monthsAgoEnd(3)');
    agg2.addEncodedQuery(query);
    agg2.query();
    var last = "";
    while (agg2.next()) {
        last = agg2.getAggregate('count','category');
    }
    gs.log(category + ": Last month:" + count + " Previous Month:" + last);
}

Scoped equivalent
To use the `addEncodedQuery()` method in a scoped application, use the corresponding scoped method: **Scoped GlideAggregate** - `addEncodedQuery(String query)`.

**GlideAggregate - addHaving(String name, String operator, String value)**

Adds a "having" element to the aggregate, such as select category, count(*) from incident group by category HAVING count(*) > 5.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Aggregate to filter on. For example, COUNT.</td>
</tr>
<tr>
<td>operator</td>
<td>String</td>
<td>Operator symbol. For example &lt;, &gt;, =, !=.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value to query on. For example, '5'.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
var trend = new GlideAggregate('incident');
trend.addTrend ('opened_at','Month');
trend.addAggregate('COUNT');

//addHaving limits the results returned to those in which the aggregate COUNT is greater than 2
trend.addHaving('COUNT', '>', '2');
trend.setGroup(false);
trend.query();
while(trend.next()) {
    gs.print(('Incidents by month ' + trend.getValue('timeref') + ' where count is more than 2 count is: ' + trend.getAggregate('COUNT')));
}
```

Output

- Incidents by month 9/2018 where count is more than 2 count is: 3
- Incidents by month 10/2018 where count is more than 2 count is: 8
- Incidents by month 11/2018 where count is more than 2 count is: 14

**GlideAggregate - addTrend(String fieldName, String timeInterval, Number numUnits)**

Adds a trend for a field.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field for which trending should occur.</td>
</tr>
<tr>
<td>timeInterval</td>
<td>String</td>
<td>Time interval for the trend. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• date</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• dayofweek</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• hour</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• minute</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• quarter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• week</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• year</td>
</tr>
<tr>
<td>numUnits</td>
<td>Number</td>
<td>Optional. Only valid when <code>timeInterval = minute</code>. Number of minutes to include in the trend. Default: 1</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var trend = new GlideAggregate('incident');
trend.addTrend ('opened_at','month');
trend.addAggregate('COUNT');
trend.setGroup(false);
trend.query();
while(trend.next()) {
    gs.print(trend.getValue('timeref') + ': ' + trend.getAggregate('COUNT'));
}
```

### Output

9/2018: 3  
10/2018: 8  
11/2018: 14
Scoped equivalent

To use the `addTrend(String fieldName, String timeInterval)` method in a scoped application, use the corresponding scoped method: `Scoped GlideAggregate - addTrend(String fieldName, String timeInterval, Number numUnits)`. 

**GlideAggregate - getAggregate(String agg, String name)**

Gets the value of an aggregate from the current record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agg</td>
<td>String</td>
<td>Type of the aggregate. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AVG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• COUNT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MIN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MAX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• STDDEV</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SUM</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field to get the aggregate from.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the aggregation. If the values being aggregated are FX Currency values, the returned value is in the format <code>&lt;currency_code;currency_value&gt;</code>, such as: USD;134.980000.</td>
</tr>
</tbody>
</table>

**Note:** If the specified field contains FX Currency values of mixed currency types, the method is not able to aggregate the values and returns a semicolon (;).

This example shows how to obtain the COUNT aggregate.

```javascript
function doMyBusinessRule(assigned_to, number) {
    var agg = new GlideAggregate('incident');
    agg.addQuery('assigned_to', assigned_to);
    // ...
agg.addQuery('category', number);
agg.addAggregate("COUNT");
agg.query();
var answer = 'false';
if (agg.next()) {
    answer = agg.getAggregate("COUNT");
    if (answer > 0)
        answer = 'true';
    else
        answer = 'false';
}
return answer;

This example shows the aggregation of an FX Currency field.

var ga = new GlideAggregate('laptop_tracker');
ga.addAggregate('SUM', 'cost');
ga.groupBy('name');
ga.query();
while (ga.next()) {
    gs.info('Aggregate results ' + ga.getValue('name') + ' => ' + ga.getAggregate('SUM', 'cost'));
}

Output:

*** Script: Aggregate results Apple MacBook Air => USD;1651.784280000000
*** Script: Aggregate results Apple MacBook Pro => USD;1651.784280000000
*** Script: Aggregate results Dell XPS => USD;470.852672000000
*** Script: Aggregate results LG =>
*** Script: Aggregate results Samsung Galaxy => USD;225.320000000000
*** Script: Aggregate results Surface3 => USD;2895.560369520000
*** Script: Aggregate results Toshiba => USD;9385.202875800000

Scoped equivalent

To use the getAggregate() method in a scoped application, use the corresponding scoped method: Scoped GlideAggregate - getAggregate(String agg, String name).

GlideAggregate - getQuery()

Retrieves the query necessary to return the current aggregate.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The query.</td>
</tr>
</tbody>
</table>

```javascript
var agg = new GlideAggregate('incident');
agg.addAggregate('count', 'category');
agg.orderByAggregate('count', 'category');
agg.orderBy('category');
agg.addQuery('opened_at', '>=', 'javascript:gs.monthsAgoStart(2)');
agg.addQuery('opened_at', '<=', 'javascript:gs.monthsAgoEnd(2)');
agg.query();
while (agg.next()) {
    var category = agg.category;
    var count = agg.getAggregate('count', 'category');
    var query = agg.getQuery();
    var agg2 = new GlideAggregate('incident');
    agg2.addAggregate('count', 'category');
    agg2.orderByAggregate('count', 'category');
    agg2.orderBy('category');
    agg2.addQuery('opened_at', '>=', 'javascript:gs.monthsAgoStart(3)');
    agg2.addQuery('opened_at', '<=', 'javascript:gs.monthsAgoEnd(3)');
    agg2.addEncodedQuery(query);
    agg2.query();
    var last = "";
    while (agg2.next()) {
        last = agg2.getAggregate('count', 'category');
    }
    gs.log(category + "": Last month:" + count + " Previous Month:" + last);
}
```

**GlideAggregate - getTotal(String agg, String name)**

Returns the number of records by summing an aggregate.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agg</td>
<td>String</td>
<td>Aggregate</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field to aggregate</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of records.</td>
</tr>
</tbody>
</table>

### GlideAggregate - getValue(String name)

Returns the value of a field.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the field.</td>
</tr>
</tbody>
</table>

```javascript
var trend = new GlideAggregate('incident');
trend.addTrend ('opened_at','Month');
trend.addAggregate('COUNT');

//addHaving limits the results returned to those in which the aggregate COUNT is greater than 2
trend.addHaving('COUNT', '>', '2');
trend.setGroup(false);
trend.query();
while(trend.next()) {
    gs.print(('Incidents by month ' + trend.getValue('timeref') + ' where count is more than 2 count is: ' + trend.getAggregate('COUNT')));
}
```
Incidents by month 9/2018 where count is more than 2 count is: 3
Incidents by month 10/2018 where count is more than 2 count is: 8
Incidents by month 11/2018 where count is more than 2 count is: 14

Scoped equivalent
To use the `getValue()` method in a scoped application, use the corresponding scoped method: **Scoped GlideAggregate - getValue(String name).**

**GlideAggregate - groupBy(String name)**
Provides the name of a field to use in grouping the aggregates.

May be called numerous times to set multiple group fields.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var count = new GlideAggregate('incident');
count.addAggregate('MIN', 'sys_mod_count');
count.addAggregate('MAX', 'sys_mod_count');
count.addAggregate('AVG', 'sys_mod_count');
count.groupBy('category');
count.query();
while (count.next()) {
    var min = count.getAggregate('MIN', 'sys_mod_count');
    var max = count.getAggregate('MAX', 'sys_mod_count');
    var avg = count.getAggregate('AVG', 'sys_mod_count');
    var category = count.category.getDisplayValue();
    gs.log(category + ' Update counts: MIN = ' + min + ' MAX = ' + max + ' AVG = ' + avg);
}
```

Scoped equivalent
To use the `groupBy()` method in a scoped application, use the corresponding scoped method: **Scoped GlideAggregate - groupBy(String name).**
**GlideAggregate - orderBy(String name)**

Provides the name of a field that should be used to order the aggregates. The field will also be added to the group-by list.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field used to order the aggregates.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
var agg = new GlideAggregate('incident');
agg.addAggregate('count','category');
agg.orderByAggregate('count', 'category');
agg.orderBy('category');
agg.addQuery('opened_at', '>=', 'javascript:gs.monthsAgoStart(2)');
agg.addQuery('opened_at', '<=', 'javascript:gs.monthsAgoEnd(2)');
agg.query();
while (agg.next()) {
    var category = agg.category;
    var count = agg.getAggregate('count','category');
    var query = agg.getQuery();
    var agg2 = new GlideAggregate('incident');
    agg2.addAggregate('count','category');
    agg2.orderByAggregate('count', 'category');
    agg2.orderBy('category');
    agg2.addQuery('opened_at', '>=', 'javascript:gs.monthsAgoStart(3)');
    agg2.addQuery('opened_at', '<=', 'javascript:gs.monthsAgoEnd(3)');
    agg2.addEncodedQuery(query);
    agg2.query();
    var last = "";
    while (agg2.next()) {
        last = agg2.getAggregate('count','category');
    }
    gs.log(category + " : Last month:" + count + " Previous Month:" + last);
}
```
**Scoped equivalent**

To use the `orderBy()` method in a scoped application, use the corresponding scoped method: `Scoped GlideAggregate - orderBy(String name)`.

**GlideAggregate - orderByAggregate(String agg, String name)**

Orders the aggregates based on the specified aggregate and field.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agg</td>
<td>String</td>
<td>Type of aggregation, for example SUM, COUNT, MIN, MAX.</td>
</tr>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field to aggregate.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var agg = new GlideAggregate('incident');
agg.addAggregate('count','category');
agg.orderByAggregate('count', 'category');
agg.orderBy('category');
agg.addQuery('opened_at', '>=', 'javascript:gs.monthsAgoStart(2)');
agg.addQuery('opened_at', '<=', 'javascript:gs.monthsAgoEnd(2)');
agg.query();
while (agg.next()) {
  var category = agg.category;
  var count = agg.getAggregate('count','category');
  var query = agg.getQuery();
  var agg2 = new GlideAggregate('incident');
  agg2.addAggregate('count','category');
  agg2.orderByAggregate('count', 'category');
  agg2.orderBy('category');
  agg2.addQuery('opened_at', '>=', 'javascript:gs.monthsAgoStart(3)');
  agg2.addQuery('opened_at', '<=', 'javascript:gs.monthsAgoEnd(3)');
  agg2.addEncodedQuery(query);
  agg2.query();
  var last = "";
  while (agg2.next()) {
    last = agg2.getAggregate('count','category');
  }
}
```
Scoped equivalent

To use the `orderByAggregate()` method in a scoped application, use the corresponding scoped method: `Scoped GlideAggregate - orderByAggregate(String agg, String fieldName)`. 

GlideAggregate - query()

Issues the query and gets the results.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var agg = new GlideAggregate('incident');
agg.addAggregate('count','category');
agg.orderByAggregate('count', 'category');
agg.orderBy('category');
agg.addQuery('opened_at', '>=', 'javascript:gs.monthsAgoStart(2)');
agg.addQuery('opened_at', '<=', 'javascript:gs.monthsAgoEnd(2)');
agg.query();
while (agg.next()) {
    var category = agg.category;
    var count = agg.getAggregate('count','category');
    var query = agg.getQuery();
    var agg2 = new GlideAggregate('incident');
    agg2.addAggregate('count','category');
    agg2.orderByAggregate('count', 'category');
    agg2.orderBy('category');
    agg2.addQuery('opened_at', '>=', 'javascript:gs.monthsAgoStart(3)');
    agg2.addQuery('opened_at', '<=', 'javascript:gs.monthsAgoEnd(3)');
```
agg2.addEncodedQuery(query);
agg2.query();
var last = "";
while (agg2.next()) {
  last = agg2.getAggregate('count', 'category');
}
gs.log(category + " Last month:" + count + " Previous Month:" + last);

Scoped equivalent

To use the `query()` method in a scoped application, use the corresponding scoped method: `Scoped GlideAggregate - query()`.

GlideAggregate - `setGroup(Boolean b)`

Sets whether to group the results.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| b    | Boolean | Flag that indicates whether to group the results. Valid values:  
|      |         | • true: Group the results.  
|      |         | • false: Do not group the results.               |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var ga = new GlideAggregate('incident');
ga.addAggregate('COUNT', 'category');
ga.setGroup(true);

Scoped equivalent

To use the `setGroup()` method in a scoped application, use the corresponding scoped method: `Scoped GlideAggregate - setGroup(Boolean b)`. 

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
GlideAggregate - Scoped

GlideAggregate enables creating database aggregation queries.

The scoped GlideAggregate class is an extension of GlideRecord and provides database aggregation (AVG, COUNT, MIN, MAX, STDDEV, SUM) queries. This functionality can be helpful when creating customized reports or in calculations for calculated fields. The GlideAggregate class works only on number fields.

When you use GlideAggregate on currency or price fields, you are working with the reference currency value. Be sure to convert the aggregate values to the user’s session currency for display. Because the conversion rate between the currency or price value (displayed value) and its reference currency value (aggregation value) might change, the result may not be what the user expects.

Note: When using an on-premise system, the database server time zone must be set to GMT/UTC for this class to work properly.

Scoped GlideAggregate - addAggregate(String agg, String name)

Adds an aggregate to a database query.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| agg   | String| Name of an aggregate to include in a database query. Valid values:  
|       |       | • AVG  
|       |       | • COUNT  
|       |       | • MIN  
|       |       | • MAX  
|       |       | • STDDEV  
|       |       | • SUM  
| name  | String| Optional. Name of the field to group the results of the aggregation by. Default: Null  

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The following example shows how to add aggregates to a query on the category and software fields in the Incident [incident] table.

```javascript
var incidentGA = new GlideAggregate('incident');

incidentGA.addQuery('category', 'software');

incidentGA.setGroup(false);

incidentGA.addAggregate('COUNT', 'sys_mod_count');
incidentGA.addAggregate('SUM', 'sys_mod_count');
incidentGA.addAggregate('AVG', 'sys_mod_count');
incidentGA.addAggregate('MIN', 'sys_mod_count');
incidentGA.addAggregate('MAX', 'sys_mod_count');
incidentGA.addAggregate('STDDEV', 'sys_mod_count');

incidentGA.query();

if (incidentGA.next()) {
    gs.info('COUNT: ' + incidentGA.getAggregate('COUNT', 'sys_mod_count'));
    gs.info('SUM: ' + incidentGA.getAggregate('SUM', 'sys_mod_count'));
    gs.info('AVG: ' + incidentGA.getAggregate('AVG', 'sys_mod_count'));
    gs.info('MIN: ' + incidentGA.getAggregate('MIN', 'sys_mod_count'));
    gs.info('MAX: ' + incidentGA.getAggregate('MAX', 'sys_mod_count'));
    gs.info('STDDEV: ' + incidentGA.getAggregate('STDDEV', 'sys_mod_count'));
}
```

Output

| COUNT   | 13  |
| SUM     | 273 |
| AVG     | 21.0000 |
| MIN     | 3  |
| MAX     | 95 |
| STDDEV  | 32.7694 |
Scoped GlideAggregate - addEncodedQuery(String query)

Adds an encoded query to the other queries that may have been set for this aggregate.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>String</td>
<td>An encoded query to add to the aggregate.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
//Number of incidents varies depending on the current state
//of the incident table
var count = new GlideAggregate('incident');
count.addEncodedQuery('active=true');
count.addAggregate('COUNT');
count.query();
var incidents = 0;
if (count.next())
    incidents = count.getAggregate('COUNT');
gs.info(incidents);
```

Scoped GlideAggregate - addQuery(String name, String operator, String value)

Adds a query to the aggregate.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The query to add.</td>
</tr>
<tr>
<td>operator</td>
<td>String</td>
<td>The operator for the query.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The list of values to include in the query.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>The query condition.</td>
</tr>
</tbody>
</table>

```javascript
//Number of incidents varies depending on the current state
//of the incident table
var count = new GlideAggregate('incident');
count.addQuery('active', '=''true'');
count.addAggregate('COUNT', 'category');
count.query();
while (count.next()) {
    var category = count.category;
    var categoryCount = count.getAggregate('COUNT', 'category');
    gs.info("There are currently \" + categoryCount + \" incidents with a category of \" +
    category);
}
```

Output:

There are currently 1 incidents with a category of database
There are currently 5 incidents with a category of hardware
There are currently 42 incidents with a category of inquiry
There are currently 4 incidents with a category of network
There are currently 4 incidents with a category of request
There are currently 7 incidents with a category of software

**Scoped GlideAggregate - addNotNullQuery(String fieldName)**

Adds a not null query to the aggregate.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The name of the field.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>The scoped query condition.</td>
</tr>
</tbody>
</table>
```javascript
var count = new GlideAggregate('incident');
count.addNotNullQuery('short_description');
count.query();   // Issue the query to the database to get all records
while (count.next()) {
    // add code here to process the aggregate
}
```

**Scoped GlideAggregate - addNullQuery(String fieldName)**

Adds a null query to the aggregate.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The name of the field.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>The scoped query condition.</td>
</tr>
</tbody>
</table>

```javascript
var count = new GlideAggregate('incident');
count.addNullQuery('short_description');
count.query();   // Issue the query to the database to get all records
while (count.next()) {
    // add code here to process the aggregate
}
```

**Scoped GlideAggregate - addTrend(String fieldName, String timeInterval, Number numUnits)**

Adds a trend for a specified field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field for which trending should occur.</td>
</tr>
<tr>
<td>timeInterval</td>
<td>String</td>
<td>Time interval for the trend. Valid values:</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• date</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• dayofweek</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• hour</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• minute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• quarter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• week</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• year</td>
<td></td>
<td></td>
</tr>
<tr>
<td>numUnits</td>
<td>Number</td>
<td>Optional. Only valid when <code>timeInterval = minute</code>. Number of minutes to include in the trend. Default: 1</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var ga = new GlideAggregate('incident');
ga.addAggregate('COUNT'); // Count all incidents opened each quarter
ga.addTrend('opened_at', 'quarter');
ga.query();
while(ga.next()) {
  gs.info([ga.getValue('timeref'), ga.getAggregate('COUNT')]);
}
```

**Output:**

- 3/2018, 9
- 4/2018, 2
- 1/2019, 38
- 2/2019, 310

---

**Scoped GlideAggregate - getAggregate(String agg, String name)**

Returns the value of an aggregate from the current record.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agg</td>
<td>String</td>
<td>The type of the aggregate. For example, SUM or COUNT.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field on which to perform the aggregation.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The value of the aggregation.</td>
</tr>
<tr>
<td></td>
<td>If the values being aggregated are FX Currency values, the returned value</td>
</tr>
<tr>
<td></td>
<td>is in the format <code>&lt;currency_code;curren_value&gt;</code>, such as: USD;134.980000.</td>
</tr>
</tbody>
</table>

ℹ️ **Note:** If the specified field contains FX Currency values of mixed currency types, the method is not able to aggregate the values and returns a semicolon (;).

Shows an aggregation that returns the number of records in the Incident table.

```javascript
var count = new GlideAggregate('incident');
count.addAggregate('COUNT');
count.query();
var incidents = 0;
if (count.next()) {
  incidents = count.getAggregate('COUNT');
}
//Number of incidents varies depending on the current state of the incident table
gs.info('Number of incidents: ' + incidents);
```

**Output:** Number of incidents: 63

Shows the aggregation of an FX Currency field.

```javascript
var ga = new GlideAggregate('laptop_tracker');
ga.addAggregate('SUM', 'cost');
ga.groupBy('name');
ga.query();
while (ga.next()) {
  gs.info('Aggregate results ' + ga.getValue('name') + ' => ' + ga.getAggregate('SUM', 'cost'));
}
```
Output: *** Script: Aggregate results Apple MacBook Air => USD:1651.784280000000
*** Script: Aggregate results Apple MacBook Pro => USD:1651.784280000000
*** Script: Aggregate results Dell XPS => USD:470.852672000000
*** Script: Aggregate results LG =>
*** Script: Aggregate results Samsung Galaxy => USD:225.320000000000
*** Script: Aggregate results Surface3 => USD:2895.560369520000
*** Script: Aggregate results Toshiba => USD:9385.202875800000

Scoped GlideAggregate - getAggregateEncodedQuery()

Gets the query necessary to return the current aggregate.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The encoded query to get the aggregate.</td>
</tr>
</tbody>
</table>

```javascript
var count = new GlideAggregate('incident');
count.addAggregate('MIN', 'sys_mod_count');
count.groupBy('category');
count.query();
while (count.next()) {
    gs.info(count.getAggregateEncodedQuery());
}
```
**Scoped GlideAggregate - getEncodedQuery()**
Retrieves the encoded query.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The encoded query.</td>
</tr>
</tbody>
</table>

```javascript
var count = new GlideAggregate('incident');
count.addAggregate('MIN', 'sys_mod_count');
count.addAggregate('MAX', 'sys_mod_count');
count.addAggregate('AVG', 'sys_mod_count');
count.groupBy('category');
count.query();
gs.info(count.getEncodedQuery());
```

Output:

```javascript
ORDERBYcategory^GROUPBYcategory
```

**Scoped GlideAggregate - getRowCount()**
Retrieves the number of rows in the GlideAggregate object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of rows in the GlideAggregate object.</td>
</tr>
</tbody>
</table>

```javascript
var count = new GlideAggregate('incident');
count.addAggregate('MIN', 'sys_mod_count');
```
```javascript
var count = new GlideAggregate('incident');
count.addAggregate('MIN', 'sys_mod_count');
count.addAggregate('MAX', 'sys_mod_count');
count.addAggregate('AVG', 'sys_mod_count');
count.groupBy('category');
gs.info(count.getRowCount());
while (count.next()) {
    var min = count.getAggregate('MIN', 'sys_mod_count');
    var max = count.getAggregate('MAX', 'sys_mod_count');
    var avg = count.getAggregate('AVG', 'sys_mod_count');
    var category = count.category.getDisplayValue();
    gs.info(category + " Update counts: MIN = " + min + " MAX = " + max + " AVG = " + avg);
}
```

Output:

```plaintext
6
Database Update counts: MIN = 8 MAX = 48 AVG = 28.0000
Hardware Update counts: MIN = 4 MAX = 14 AVG = 6.6250
Inquiry / Help Update counts: MIN = 0 MAX = 34 AVG = 6.5714
Network Update counts: MIN = 3 MAX = 37 AVG = 18.6000
Request Update counts: MIN = 5 MAX = 39 AVG = 13.4000
Software Update counts: MIN = 4 MAX = 98 AVG = 24.0000
```

Scoped GlideAggregate - getTableName()
Retrieves the table name associated with this GlideAggregate object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>none</td>
<td>none</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The table name.</td>
</tr>
</tbody>
</table>

```javascript
var count = new GlideAggregate('incident');
count.addAggregate('MIN', 'sys_mod_count');
count.addAggregate('MAX', 'sys_mod_count');
count.addAggregate('AVG', 'sys_mod_count');
count.groupBy('category');
```
count.query();
gs.info(count.getTableName());

Scoped GlideAggregate - getValue(String name)
Returns the value of the specified field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field within the current table to return.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the specified field.</td>
</tr>
</tbody>
</table>

var count = new GlideAggregate('incident');
count.addAggregate('MAX', 'sys_mod_count');
count.groupBy('category');
count.query();
while (count.next()) {
    gs.info(count.getValue('category') + " category had " + count.getAggregate('MAX', 'sys_mod_count') + " updates");
}

Output:
category had 12 updates
hardware category had 15 updates
inquiry category had 36 updates
network category had 37 updates
software category had 95 updates

Scoped GlideAggregate - GlideAggregate(String tableName)
Creates a GlideAggregate object on the specified table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table.</td>
</tr>
</tbody>
</table>
```javascript
var count = new GlideAggregate('incident');

count.addAggregate('MIN', 'sys_mod_count');
count.addAggregate('MAX', 'sys_mod_count');
count.addAggregate('AVG', 'sys_mod_count');
count.groupBy('category');
count.query();

while (count.next()) {
    var min = count.getAggregate('MIN', 'sys_mod_count');
    var max = count.getAggregate('MAX', 'sys_mod_count');
    var avg = count.getAggregate('AVG', 'sys_mod_count');
    var category = count.category.getDisplayValue();
    gs.info(category + " Update counts: MIN = " + min + " MAX = " + max + " AVG = " + avg);
}
```

**Output:**

- **Database Update counts:** MIN = 8 MAX = 48 AVG = 28.0000
- **Hardware Update counts:** MIN = 4 MAX = 14 AVG = 6.6250
- **Inquiry / Help Update counts:** MIN = 0 MAX = 34 AVG = 6.5714
- **Network Update counts:** MIN = 3 MAX = 37 AVG = 18.6000
- **Request Update counts:** MIN = 5 MAX = 39 AVG = 13.4000
- **Software Update counts:** MIN = 4 MAX = 98 AVG = 24.0000
Scoped GlideAggregate - hasNext()

Determines if there are any more records in the GlideAggregate object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if there are more results in the query set.</td>
</tr>
</tbody>
</table>

```javascript
var agg = new GlideAggregate('incident');
agg.addAggregate('AVG', 'sys_mod_count');
agg.groupBy('category');
agg.query();
while (agg.hasNext()) {
    agg.next();
    var avg = agg.getAggregate('AVG', 'sys_mod_count');
    var category = agg.category.getDisplayValue();
    gs.info(category + ': AVG = ' + avg);
}
```

**Output:**

Database: AVG = 32.5000
Hardware: AVG = 12.0000
Inquiry / Help: AVG = 7.6667
Network: AVG = 24.0000
Request: AVG = 16.4000
Software: AVG = 27.0833

Scoped GlideAggregate - next()

Moves to the next record in the GlideAggregate.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if there are more records in the query set; otherwise, false.</td>
</tr>
</tbody>
</table>

```javascript
var count = new GlideAggregate('incident');
count.addAggregate('COUNT');
count.query();
var incidents = 0;
if (count.next()) {
  incidents = count.getAggregate('COUNT');
  gs.info(incidents);
}
```

Scoped GlideAggregate - orderBy(String name)

Orders the aggregates using the value of the specified field. The field will also be added to the group-by list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field to order the aggregates by.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var agg = new GlideAggregate('incident');
agg.addAggregate('count', 'category');
agg.orderBy('category');
agg.query();
while (agg.next()) {
  var category = agg.category;
  var count = agg.getAggregate('count', 'category');
  var agg2 = new GlideAggregate('incident');
  agg2.addAggregate('count', 'category');
  agg2.orderBy('category');
  gs.info(category + " Current number of incidents:" + count);
}
```
Output:

database: Current number of incidents: 2
hardware: Current number of incidents: 8
inquiry: Current number of incidents: 28
network: Current number of incidents: 5
request: Current number of incidents: 5
software: Current number of incidents: 11

**Scoped GlideAggregate - orderByAggregate(String agg, String fieldName)**

Orders the aggregates based on the specified aggregate and field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agg</td>
<td>String</td>
<td>Type of aggregation.</td>
</tr>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field to aggregate.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```plaintext
ga.addAggregate('COUNT', 'category');
ga.orderByAggregate('count', 'category');
ga.query();

while(ga.next()) {
   gs.info('Category ' + ga.category + ' ' + ga.getAggregate('COUNT', 'category'));
}
```

Output:

Category inquiry 18
Category software 11
Category hardware 7
Category network 5
Category request 5
**Scoped GlideAggregate - orderByDesc(String name)**

Sorts the aggregates in descending order based on the specified field. The field will also be added to the group-by list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var agg = new GlideAggregate('incident');
agg.addAggregate('count', 'category');
agg.orderByDesc('category');
agg.query();
while (agg.next()) {
    var category = agg.category;
    var count = agg.getAggregate('count', 'category');
    var agg2 = new GlideAggregate('incident');
    agg2.addAggregate('count', 'category');
    agg2.orderBy('category');
    gs.info(category + " Current number of incidents:" + count);
}
```

**Output:**

```
software: Current number of incidents:11
request: Current number of incidents:5
network: Current number of incidents:5
inquiry: Current number of incidents:28
hardware: Current number of incidents:8
database: Current number of incidents:2
```
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var count = new GlideAggregate('incident');
count.addAggregate('COUNT');
count.query();
var incidents = 0;
if (count.next()) {
    incidents = count.getAggregate('COUNT');
}
gs.info('Number of incidents: ' + incidents);
```

### Scoped GlideAggregate - setGroup(Boolean b)
Sets whether the results are to be grouped.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>When true the results are grouped.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var ga = new GlideAggregate('incident');
ga.addAggregate('COUNT', 'category');
ga.setGroup(true);
ga.groupBy("category");
```
ga.query();

while(ga.next()) {
    gs.info('Category ' + ga.category + ' ' + ga.getAggregate('COUNT', 'category'));
}

Output:

Category database 2
Category hardware 7
Category inquiry 18
Category network 5
Category request 5
Category software 11

**GlideApplicationProperty - Scoped, Global**

The *GlideApplicationProperty* API provides the ability to get and set application property values for a specific domain.

Application properties enable service providers to customize application functionality based on the domain. Instead of only having a single system property (System Property [sys_properties] table) that defines the functionality for all domains and users, application properties can be implemented to define functionality for specific domains.

To access this API you must activate the Domain Support - Domain Extensions Installer (com.glide.domain.msp_extensions.installer) plugin.

For additional information on domain-specific application properties, see Domain-separated application properties.

**GlideApplicationProperty - getValue(String name, String domainSysId)**

Returns the value for the specified application property and domain sys_id.

The list of available application properties is located in the Application Properties [sys_application_property] and Application Property Values [sys_application_property_value] tables.

⚠️ **Note:** If the specified domain is a child domain, and there is not an application property for the child domain, the method returns the parent domain application property, if available.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the application property to return.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td></td>
<td>If the application is not global, you must prefix the name of the sys_application_property with the application's scope.</td>
</tr>
<tr>
<td>domainSysId</td>
<td>String</td>
<td>Optional. Sys_id of the domain associated with the application property.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Session domain</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the specified application property for the specified domain. If domainSysId is not passed in the call, then returns the session domain. If the specified application property doesn't exist, returns the same-name system property value. If the user does not have access rights to the specified domain, throws an exception and no value is returned.</td>
</tr>
</tbody>
</table>

This example shows how to create a new application property and override its value for ACME and Cisco domains.

```javascript
var MY_APP_PROP_NAME = 'my-app-prop';
var ACME_DOMAIN_ID = 'c90d4b084a362312013398f051272c0d';
var CISCO_DOMAIN_ID = 'c90f91924a362312001bdefae0f35d68';

// create a new app_config entry:
var now_GR = new GlideRecord('sys_application_property');
now_GR.name = MY_APP_PROP_NAME;
now_GR.default_value = 'my-app-prop-default_value';
now_GR.insert();

// Override the value for ACME domain, catch exception if user can't access the specified domain
try {
    GlideApplicationProperty.setValue(MY_APP_PROP_NAME, 'prop-value-ACME', ACME_DOMAIN_ID);
    gs.info('Value for ACME domain: ' + GlideApplicationProperty.getValue(MY_APP_PROP_NAME, ACME_DOMAIN_ID));
} catch (e) {
    gs.error('Exception caught: ' + e.name + ' - ' + e.message);
}
```
} catch (ex) {
    gs.info('Exception: ' + ex);
}

// Override the value for Cisco domain -- assuming that the current session domain is Cisco. Notice that we don't pass domain-id explicitly
session.setDomainID(CISCO_DOMAIN_ID); // NOTE: this API is not available from non-global scope (use domain-picker to switch to Cisco)
gs.info('Domain: ' + session.getCurrentDomainID());
GlideApplicationProperty.setValue(MY_APP_PROP_NAME, 'prop-value-Cisco');
gs.info('Value for current session (Cisco) domain: ' + GlideApplicationProperty.getValue(MY_APP_PROP_NAME));

Output:

*** Script: Value for ACME domain: prop-value-ACME
*** Script: Domain: c90f91924a362312001bdefae0f35d68
*** Script: Value for current session (Cisco) domain: prop-value-Cisco

GlideApplicationProperty - setValue(String name, String value, String domainSysId)

Stores the specified value in the specified application property for the specified domain, or the current session domain if not specified.

The following are guidelines for using this method:

- The application property is not created if its name contains one of the following strings:
  - glide.properties.blacklist
  - glide.properties.no_db_override
  - glide.properties.safe_overrides
  - glide.properties.maint_write_roles
- The application property is not created if there is an existing system property with the same name, and the system property is marked as private.
- The application property is not created if there is an existing system property and the application property does not reference it.
- The name of the application property in the sys_application_property table is prefixed with the application's scope if it is not global.
- This method cannot be called from a different scope.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the application property to save.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> If the application is not global, you must prefix the name of</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the sys_application_property with the application's scope.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value to store in the specified application property.</td>
</tr>
<tr>
<td>domainSysId</td>
<td>String</td>
<td>Optional. Sys_id of the domain associated with the application property.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Current session domain</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the save operation was successful. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Value was successfully saved.</td>
</tr>
<tr>
<td></td>
<td>• false: An error occurred and the application property was not created/updated.</td>
</tr>
<tr>
<td></td>
<td>If the user does not have access rights to the specified domain, throws an exception and no value is returned.</td>
</tr>
</tbody>
</table>

This example shows how to create a new application property and override its value for ACME and Cisco domains.

```javascript
var MY_APP_PROP_NAME = 'my-app-prop';
var ACME_DOMAIN_ID = 'c90d4b084a362312013398f051272c0d';
var CISCO_DOMAIN_ID = 'c90f91924a362312001bdefae0f35d68';

// create a new app_config entry:
var now_GR = new GlideRecord('sys_application_property');
now_GR.name = MY_APP_PROP_NAME;
now_GR.default_value = 'my-app-prop-default_value';
now_GR.insert();

// Override the value for ACME domain, catch exception if user can't access the specified domain
```
try {
    GlideApplicationProperty.setValue(MY_APP_PROP_NAME, 'prop-value-ACME', ACME_DOMAIN_ID);
    gs.info('Value for ACME domain: ' + GlideApplicationProperty.getValue(MY_APP_PROP_NAME, ACME_DOMAIN_ID));
} catch (ex) {
    gs.info('Exception: ' + ex);
}

// Override the value for Cisco domain -- assuming that the current session domain is Cisco. Notice that we don't pass domain-id explicitly
session.setDomainID(CISCO_DOMAIN_ID); // NOTE: this API is not available from non-global scope (use domain-picker to switch to Cisco)
gs.info('Domain: ' + session.getCurrentDomainID());
GlideApplicationProperty.setValue(MY_APP_PROP_NAME, 'prop-value-Cisco');
gs.info('Value for current session (Cisco) domain: ' + GlideApplicationProperty.getValue(MY_APP_PROP_NAME));

Output:

*** Script: Value for ACME domain: prop-value-ACME
*** Script: Domain: c90f91924a362312001bdefae0f35d68
*** Script: Value for current session (Cisco) domain: prop-value-Cisco

GlideAjax - Client

The GlideAjax class enables a client script to call server-side code in a script include. To use GlideAjax in a client script, follow these general steps.

1. Create a GlideAjax instance by calling the GlideAjax constructor. As the argument to the constructor, specify the name of the script include class that contains the method you want to call.

2. Call the addParam method with the sysparm_name parameter and the name of the script-include method you want to call.

3. (Optional) Call the addParam method one or more times to provide the script-include code with other parameters it needs.
4. Execute the server-side code by calling `getXML()`.

**Note:** `getXML()` is the preferred method for executing the code, because it is asynchronous and does not hold up the execution of other client code. Another method, `getXMLWait()`, is also available but is not recommended. Using `getXMLWait()` ensures the order of execution, but can cause the application to seem unresponsive, significantly degrading the user experience of any application that uses it. `getXMLWait()` is not available to scoped applications.

```javascript
var ga = new GlideAjax('HelloWorld'); // HelloWorld is the script include class
ga.addParam('sysparm_name','helloWorld'); // helloWorld is the script include method
ga.addParam('sysparm_user_name','Bob'); // Set parameter sysparm_user_name to 'Bob'
ga.getXML(HelloWorldParse); /* Call HelloWorld.helloWorld() with the parameter
   sysparm_user_name set to 'Bob'
   and use the callback function HelloWorldParse() to return the result when ready */

// the callback function for returning the result from the server-side code
function HelloWorldParse(response) {
  var answer = response.responseXML.documentElement.getAttribute("answer");
  alert(answer);
}
```

**GlideAjax - getAnswer()**

Retrieves the results from a server-side method called from the client via `getXMLWait()`.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>The result returned by the server-side method previously called with <code>getXMLWait()</code></td>
</tr>
</tbody>
</table>
GlideAjax - addParam(String parm_name, String parm_value)

Specifies a parameter name and value to be passed to the server-side function associated with this GlideAjax object.

You can execute addParam multiple times with different parameters and values.

⚠️ Note: The first call to addParam should be with the parameter sysparm_name and the name of the server-side method you want to call. The server-side code does not execute until the client script calls getXML().

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parm_name</td>
<td>String</td>
<td>The name of the parameter to pass. (The name must begin with the sysparm_.)</td>
</tr>
<tr>
<td>parm_value</td>
<td>String</td>
<td>The value to assign to parm_name.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideAjax - getXML(Function callback)

Sends the server a request to execute the method and parameters associated with this GlideAjax object.

The server processes the request asynchronously and -- when ready -- returns the results via the function specified as the callback_function.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callback</td>
<td>Function</td>
<td>The name of the callback function to process the results returned by the server.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
```javascript
var comments = gel("dialog_comments").value;
var ga = new GlideAjax('validateComments'); // Call script include to escape text
ga.addParam('sysparm_name', 'validateComments');
ga.addParam('sysparm_comments', comments);
ga.getXML(callback);

return false;

function callback(response) {
    var comments = response.responseXML.documentElement.getAttribute("answer");
    comments = trim(comments);
    if (comments == "") {
        // If comments are empty, alert the user and stop submission
        alert("Please enter your comments before submitting.");
    } else {
        // If there are comments, close the dialog window and submit them
        GlideDialogWindow.get().destroy(); // Close the dialog window
        g_form.setValue("comments", comments); // Set the "Comments" field with comments in the dialog
    }
}
```

**GlideAjax - getXMLAnswer(Function callback, Object additionalParam, Object responseParam)**

Calls the processor asynchronously and gets the answer element of the response in XML format.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callback</td>
<td>Function</td>
<td>Callback function. The function receives the answer element of the response in XML format as an argument.</td>
</tr>
<tr>
<td>additionalParam</td>
<td>Object</td>
<td>Optional. Name-value pair of additional parameters.</td>
</tr>
<tr>
<td>responseParam</td>
<td>Object</td>
<td>Optional. Second argument for the callback function.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Returns the number of attachments

```javascript
function updateAttachmentCount(sysid) {
    var ga = new GlideAjax('AttachmentAjax');
    ga.addParam('sysparm_type', 'attachmentCount');
    ga.addParam('sysparm_value', sysid);
    ga.getXMLAnswer(numberOfAttachments, null, sysid); // callback: numberOfAttachments with args (answer, sysid)
}
```

```javascript
function numberOfAttachments(answer, sysid) {
    // we want to know there are 5 attachments, not 5.0 attachments
    var number = parseInt(answer);
    var buttons = $$('.attachmentNumber_' + sysid);
    if (buttons[0] == undefined)
        $('header_attachment_list_label').down().innerHTML = number;
    else {
        for (var i = 0; i < buttons.length; i++) {
            buttons[i].innerHTML = number;
        }
    }
}
```

GlideAjax - getXMLWait()

Sends the server a request to execute the method and parameters associated with this GlideAjax object.

The server processes the request synchronously and will not process further requests from the client until finished. To retrieve the results, the client must call `getAnswer()`. Using `getXMLWait()` ensures the order of execution, but can cause the application to seem unresponsive, significantly degrading the user experience of any application that uses it. We recommend using `getXML()` instead.

⚠️ Note: `getXMLWait()` is not available to scoped applications.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var ga = new GlideAjax('HelloWorld');
ga.addParam('sysparm_name','helloWorld');
ga.addParam('sysparm_user_name','Bob');
ga.getXMLWait();
alert(ga.getAnswer());
```

Scoped equivalent

`getXMLWait()` is not available to scoped applications. Instead use the `getXML()` method.

GlideAjax - GlideAjax(String class_name)

Constructor for GlideAjax.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>class_name</td>
<td>String</td>
<td>The name of the server-side class that contains the method you want to execute.</td>
</tr>
</tbody>
</table>

GlideAjaxV3 - Client

The GlideAjaxV3 API provides the ability to asynchronously execute server-side scripts from a client-side script.

The GlideAjaxV3 API can be used in client-side scripts using ListV2 and ListV3 APIs.

GlideAjaxV3 - addParam(String name, String value)

Set a name-value pair to be sent to the processor.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the parameter. This usually has the prefix 'sysparm_'</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The parameter value.</td>
</tr>
</tbody>
</table>

### GlideAjaxV3 - getJSON(Function callback)

Call the processor asynchronously and get the answer element of the response in JSON format.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callback</td>
<td>Function</td>
<td>The callback function. The function receives the answer element of the response as a JSON object.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### GlideAjaxV3 - getParam(String name)

Returns the value of the specified parameter.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the parameter to return.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The specified parameter's value.</td>
</tr>
</tbody>
</table>

GlideAjaxV3 - getParams()

Returns the name-value pairs for the request.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The request's name-value pairs.</td>
</tr>
</tbody>
</table>

GlideAjaxV3 - getProcessor()

Returns the server-side script that the request is going to use.

This returns the 'sysparm_processor' parameter.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The value of the request's sysparm_processor parameter.</td>
</tr>
</tbody>
</table>

GlideAjaxV3 - getURL()

Returns the target URL.
### GlideAjaxV3 - getXML(Function callback)

Call the processor asynchronously and get the response in XML format.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callback</td>
<td>Function</td>
<td>The callback function. The function receives the response as an argument.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The URL where the Ajax request will be sent.</td>
</tr>
</tbody>
</table>

### GlideAjaxV3 - getXMLAnswer(Function callback)

Call the processor asynchronously and get the answer element of the response in XML format.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callback</td>
<td>Function</td>
<td>The callback function. The function receives the answer element of the response in XML format as an argument.</td>
</tr>
</tbody>
</table>
function autofillPhoneNumber(sysid) {
    var ga = new GlideAjax('x_abc_myscope.AjaxUtils');
    ga.addParam('sysparm_type', 'getPhoneNumberForUser');
    ga.addParam('sysparm_user', sysid);
    ga.getXMLAnswer(function(answer) {
        g_form.setValue('phone_number', answer);
    });
}

GlideAjaxV3 - GlideAjax(String processor)
Creates an instance of the GlideAjaxV3 class.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>processor</td>
<td>String</td>
<td>The name of the processor (server-side script) to call.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideAjaxV3 - GlideAjax(String processor, String targetURL)
Creates an instance of the GlideAjaxV3 class.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>processor</td>
<td>String</td>
<td>The name of the processor (server-side script) to call.</td>
</tr>
<tr>
<td>targetURL</td>
<td>String</td>
<td>(Optional) Override the xmlhttp processor url. If this parameter is not specified, the default is xmlhttp.do.</td>
</tr>
</tbody>
</table>
**GlideAjaxV3 - setErrorCallback(Function callback)**

Sets a callback function to be called if the Ajax request fails.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callback</td>
<td>Function</td>
<td>The function to be called if the Ajax request fails. The callback function has one parameter, the XMLHttpRequest object.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideAjaxV3 - setProcessor(String serverScript)**

Sets the request's server-side script. The server-side script is also called the processor.

This sets the 'sysparm_processor' parameter.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverScript</td>
<td>String</td>
<td>The server-side script (processor) to receive the request.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
**GlideappCalculationHelper - Global**

The GlideappCalculationHelper API is a scriptable object that provides methods that add items to an existing request or request the recalculation of the price of a request.

The methods for this API are used in global server-side scripts (script includes, business rules, etc.). There is currently no support for scoped applications.

**GlideappCalculationHelper - addItemToExistingRequest(String requestID, String catalogID, String quantity)**

Adds a specific catalog item to an existing request. This API should be used only before a request (REQ) is approved. After the REQ is approved, the RITM process is not initiated.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestID</td>
<td>String</td>
<td>Sys_id of the request to which to add the catalog item(s)</td>
</tr>
<tr>
<td>catalogID</td>
<td>String</td>
<td>Sys_id of the catalog item to add to the request</td>
</tr>
<tr>
<td>quantity</td>
<td>String</td>
<td>Number (quantity) of the specified catalog item to be add to the request</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var catItemId = "04b7e94b4f7b4200086eeed18110c7fd";
var requestId = "6eed229047801200e0ef563dbb9a71c2";
var helper = new GlideappCalculationHelper();
helper.addItemToExistingRequest(requestId, catItemId, "1");
```

**GlideappCalculationHelper - rebalanceRequest(String requestID)**

Recalculates the price of all of the items in a specified request.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestID</td>
<td>String</td>
<td>Sys_id of the request for which to recalculate the price</td>
</tr>
</tbody>
</table>
var catItemId = "04b7e94b4f7b4200086eed18110c7fd";
var requestId = "6eed229047801200e0ef563dbb9a71c2";
var helper = new GlideappCalculationHelper();
// Add an item to the request
helper.addItemToExistingRequest(requestId, catItemId, "1");
// Re-calculate the price of the request after adding the item
helper.rebalanceRequest(requestId);

**GlideCalendarDateTime - Scoped**

The GlideCalendarDateTime class provides methods for performing operations on GlideCalendarDateTime objects, such as instantiating GlideCalendarDateTime objects or working with glide_date_time fields.

Use the GlideCalendarDateTime methods to perform date-time operations, such as instantiating a GlideCalendarDateTime object, performing date-time calculations, formatting a date-time, or converting between date-time formats.

**GlideCalendarDateTime - add(GlideTime time)**

Adds a GlideTime object to the current GlideCalendarDateTime object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>GlideTime</td>
<td>Time to add.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
GlideCalendarDateTime - add(Number milliseconds)

Adds a specified number of milliseconds to the GlideCalendarDateTime object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>milliseconds</td>
<td>Number</td>
<td>Number of milliseconds to add</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideCalendarDateTime - addDaysLocalTime(Number days)

Adds a specified number of days to the current GlideCalendarDateTime object. A negative parameter subtracts days.

The method determines the local date and time equivalent to the value stored by the GlideCalendarDateTime object, then adds or subtracts days using the local date and time values.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>days</td>
<td>Number</td>
<td>Number of days to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gcdt.addDaysLocalTime(-1);
gs.info(gcdt.getLocalDate());
```

Output:

2011-08-30

**GlideCalendarDateTime - addDaysUTC(Number days)**

Adds a specified number of days to the current GlideCalendarDateTime object. A negative parameter subtracts days.

The method determines the UTC date and time equivalent to the value stored by the GlideCalendarDateTime object, then adds or subtracts days using the UTC date and time values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>days</td>
<td>Number</td>
<td>Number of days to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gcdt.addDaysUTC(-1);
gs.info(gcdt.getDate());
```
GlideCalendarDateTime - addMonthsLocalTime(Number months)

Adds a specified number of months to the current GlideCalendarDateTime object. A negative parameter subtracts months.

The method determines the local date and time equivalent to the value stored by the GlideCalendarDateTime object, then adds or subtracts months using the local date and time values.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>months</td>
<td>Number</td>
<td>Number of months to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gcdt.addMonthsLocalTime(2);
gs.info(gcdt.getDate());
```

Output:

2011-10-31

GlideCalendarDateTime - addMonthsUTC(Number months)

Adds a specified number of months to the current GlideCalendarDateTime object. A negative parameter subtracts months.

The method determines the UTC date and time equivalent to the value stored by the GlideCalendarDateTime object, then adds or subtracts months using the UTC date and time values.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>months</td>
<td>Number</td>
<td>Number of months to add. Use a negative number to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gcdt.addMonthsUTC(2);
gs.info(gcdt.getDate());

Output:

2011-10-31
```

**GlideCalendarDateTime - addSeconds(Number seconds)**

Adds a specified number of seconds to the GlideCalendarDateTime object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>seconds</td>
<td>Number</td>
<td>Number of seconds to add</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-12-07 08:00:00");
gcdt.addSeconds(1000);
gs.info(gcdt.getValue());

Output:

2011-12-07 08:16:40
```
GlideCalendarDateTime - addWeeksLocalTime(Number weeks)

Adds a specified number of weeks to the current GlideCalendarDateTime object. A negative parameter subtracts weeks.

The method determines the local date and time equivalent to the value stored by the GlideCalendarDateTime object, then adds or subtracts weeks using the local date and time values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>weeks</td>
<td>Number</td>
<td>Number of weeks to add. Use a negative number to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gcdt.addWeeksLocalTime(-1);
gs.info(gcdt.getDate());
```

Output:

2011-08-24

GlideCalendarDateTime - addWeeksUTC(Number weeks)

Adds a specified number of weeks to the current GlideCalendarDateTime object. A negative parameter subtracts weeks.

The method determines the UTC date and time equivalent to the value stored by the GlideCalendarDateTime object, then adds or subtracts weeks using the UTC date and time values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>weeks</td>
<td>Number</td>
<td>Number of weeks to add. Use a negative number to subtract.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gcdt.addWeeksUTC(-1);
gs.info(gcdt.getDate());
```

Output:

2011-08-24

**GlideCalendarDateTime - addYearsLocalTime(Number years)**

Adds a specified number of years to the current GlideCalendarDateTime object. A negative parameter subtracts years.

The method determines the local date and time equivalent to the value stored by the GlideCalendarDateTime object, then adds or subtracts years using the local date and time values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>years</td>
<td>Number</td>
<td>Number of years to add. To subtract use a negative value.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2010-08-31 08:00:00");
gcdt.addYearsLocalTime(1);
gs.info(gcdt.getDate());
```

Output:

2011-08-31
GlideCalendarDateTime - addYearsUTC(Number years)

Adds a specified number of years to the current GlideCalendarDateTime object. A negative parameter subtracts years.

The date and time value stored by GlideCalendarDateTime object is interpreted as being in the UTC time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>years</td>
<td>Number</td>
<td>Number of years to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
var gcdt = new GlideCalendarDateTime("2010-08-31 08:00:00");
gcdt.addYearsUTC(1);
gs.info(gcdt.getDate());
```

Output:

2011-08-31

GlideCalendarDateTime - after(GlideCalendarDateTime gdt)

Determines if the GlideCalendarDateTime object's date and time occurs after the specified GlideCalendarDateTime object's date and time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gdt</td>
<td>GlideCalendarDateTime</td>
<td>Date and time to check against.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the GlideCalendarDateTime object's date and time is after the date and time specified by the parameter. Possible values:</td>
</tr>
</tbody>
</table>

var gcdt = new GlideCalendarDateTime("2010-08-31 08:00:00");
gcdt.addYearsUTC(1);
gs.info(gcdt.getDate());
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• true: GlideCalendarDateTime object date and time is after the specified object's date and time.</td>
<td></td>
</tr>
<tr>
<td>• false: GlideCalendarDateTime object date and time is before or equal to the specified object's date and time.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt1 = new GlideCalendarDateTime("2016-05-09 10:11:12");
var gdt2 = new GlideCalendarDateTime("2015-06-12 15:11:12");
gs.info(gdt1.after(gdt2));
```

**Output**

```
true
```

### GlideCalendarDateTime - before(GlideCalendarDateTime gdt)

Determines if the GlideCalendarDateTime object's date and time occurs before the specified GlideCalendarDateTime object's date and time.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gdt</td>
<td>GlideCalendarDateTime</td>
<td>Date and time to check against.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the GlideCalendarDateTime object's date and time is before the date and time specified by the parameter. Possible values:</td>
</tr>
<tr>
<td>• true: GlideCalendarDateTime object date and time is before the specified object's date and time.</td>
<td></td>
</tr>
<tr>
<td>• false: GlideCalendarDateTime object date and time is after or equal to the specified object's date and time.</td>
<td></td>
</tr>
</tbody>
</table>
var gdt1 = new GlideCalendarDateTime("2016-05-09 10:11:12");
var gdt2 = new GlideCalendarDateTime("2015-06-12 15:11:12");
gs.info(gdt1.before(gdt2));

Output
false

GlideCalendarDateTime - compareTo(Object dateTime)
Compares two date and time objects to determine whether one occurs before the other or if they are equivalent.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>Object</td>
<td>Date time in a GlideCalendarDateTime object.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>• 0 = Dates are equal</td>
</tr>
<tr>
<td></td>
<td>• 1 = Object's date is after the date specified in the parameter</td>
</tr>
<tr>
<td></td>
<td>• -1 = Object's date is before the date specified in the parameter</td>
</tr>
</tbody>
</table>

var initDate = new GlideCalendarDateTime("2011-08-01 12:00:00");
var compDate1 = new GlideCalendarDateTime("2011-08-01 12:00:00");
var compDate2 = new GlideCalendarDateTime("2011-07-31 12:00:00");
var compDate3 = new GlideCalendarDateTime("2011-08-04 16:00:00");

gs.info(initDate.compareTo(compDate1)); // Equals (0)
gs.info(initDate.compareTo(compDate2)); // initDate is after compDate2 (1)
gs.info(initDate.compareTo(compDate3)); // initDate is before compDate3 (-1)

GlideCalendarDateTime - equals(Object GCDT)
Compares an object with an existing value for equality.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCDT</td>
<td>Object</td>
<td>Object to compare. Can be a GlideCalendarDateTime object or a valid date time string.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if they are equal, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 00:00:00");
gs.info(gcdt.equals("2011-09-30 00:12:01"));
```

Output:

false

---

**GlideCalendarDateTime - getDate()**

Returns the date stored by the GlideCalendarDateTime object, expressed in the standard format, yyyy-MM-dd, and the system time zone, UTC by default.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDate</td>
<td>Date in the system time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gs.info(gcdt.getDate());
```

Output:

2011-08-31
GlideCalendarDateTime - getDayOfMonthLocalTime()

Returns the day of the month stored by the GlideCalendarDateTime object, expressed in the current user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Day of the month in the user's time zone, from 1 to 31.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-12-02 12:00:00");
gs.info(gcdt.getDayOfMonthLocalTime());
```

Output:

2

GlideCalendarDateTime - getDayOfMonthUTC()

Gets the day of the month stored by the GlideCalendarDateTime object, expressed in the UTC time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Day of the month in the UTC time zone, from 1 to 31.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-12-02 12:00:00");
gs.info(gcdt.getDayOfMonthUTC());
```

Output:
GlideCalendarDateTime - getDayOfWeekLocalTime()

Returns the day of the week stored by the GlideCalendarDateTime object, expressed in the user's time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Day of the week value - Monday = 1, ... Sunday = 7</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-12-01 12:00:00");
gs.info(gcdt.getDayOfWeekLocalTime());
```

Output:

4

GlideCalendarDateTime - getDayOfWeekUTC()

Returns the day of the week stored by the GlideCalendarDateTime object, expressed in the UTC time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Day of the week value - Monday = 1, ... Sunday = 7</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.

1947
```javascript
var gcdt = new GlideCalendarDateTime("2011-12-01 12:00:00");
gs.info(gcdt.getDayOfWeekUTC());
```

Output:

4

### GlideCalendarDateTime - getDaysInMonthLocalTime()

Returns the number of days in the month stored by the GlideCalendarDateTime object, expressed in the current user’s time zone.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of days in the current month in the user’s time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime(); //December
gs.info(gcdt.getDaysInMonthLocalTime());
```

Output:

31

### GlideCalendarDateTime - getDaysInMonthUTC()

Returns the number of days in the month stored by the GlideCalendarDateTime object, expressed in the UTC time zone.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of days in the month stored by the GlideCalendarDateTime object, expressed in the UTC time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime(); //December
gs.info(gcdt.getDaysInMonthUTC());
```

**Output:**

31

### GlideCalendarDateTime - getDisplayValue()

Returns the date and time value in the current user's display format and time zone. Referring to the GlideCalendarDateTime object directly returns the date and time value in the GMT time zone.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Date and time in the user's format and time zone. Keep in mind when designing business rules or script includes that this method may return values in different formats for different users.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gs.info(gcdt.getDisplayValue());
```

**Output:**

2011-08-31 01:00:00

### GlideCalendarDateTime - getDisplayValueInternal()

Returns the display value in the internal format (yyyy-MM-dd HH:mm:ss).
This method is useful for date/time fields, but not for date fields.

### Parameters
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Date and time values for the GlideCalendarDateTime object in the current user's time zone and the internal date and time format of yyyy-MM-dd HH:mm:ss.</td>
</tr>
</tbody>
</table>

```java
// Wednesday
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gs.info(gcdt.getDisplayValueInternal());
```

**Output:**

```
2011-08-31 01:00:00
```

#### GlideCalendarDateTime - getDSTOffset()

Returns the amount of time that daylight saving time is offset.

### Parameters
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Amount of time, in milliseconds, that daylight saving is offset. Returns 0 if there is no offset or if the time is not during daylight saving time.</td>
</tr>
</tbody>
</table>

```java
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gs.info(gcdt.getDSTOffset());
```

**Output:**

```
```
GlideCalendarDateTime - getErrorMsg()

Returns the current error message.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Error message</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 aa:00:00"); //bad
gs.info(gcdt.isValid()); //false
gs.info(gcdt.getErrorMsg()); //reason
```

Output:

false
Could not parse DateTime: 2011-08-31 aa:00:00

GlideCalendarDateTime - getInternalFormattedLocalTime()

Returns the object's time in the local time zone and in the internal format.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Object's time in the local time zone and the internal format.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gs.info(gcdt.getInternalFormatedLocalTime());

Output:
2011-08-31 08:00:00

GlideCalendarDateTime - getLocalDate()

Returns the date stored by the GlideCalendarDateTime object, expressed in the standard format, yyyy-MM-dd, and the current user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDate</td>
<td>Date in the user's time zone.</td>
</tr>
</tbody>
</table>

var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gs.info(gcdt.getLocalDate());

Output:
2011-08-31

GlideCalendarDateTime - getMonthLocalTime()

Returns the month stored by the GlideCalendarDateTime object, expressed in the current user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### GlideCalendarDateTime - getMonthLocalTime()

Returns the month stored by the GlideCalendarDateTime object, expressed in the UTC time zone.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Numerical value of the month, Jan=1, Dec=12.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime(); //December
gs.info(gcdt.getMonthLocalTime());
```

Output:

```
12
```

### GlideCalendarDateTime - getMonthUTC()

Returns the month stored by the GlideCalendarDateTime object, expressed in the UTC time zone.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Numerical value of the month, Jan=1, Dec=12.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime(); //December
gs.info(gcdt.getMonthUTC());
```

Output:

```
12
```

### GlideCalendarDateTime - getNumericValue()

Returns the number of milliseconds since January 1, 1970, 00:00:00 GMT.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of milliseconds since January 1, 1970, 00:00:00 GMT.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gs.info(gcdt.getNumericValue());
```

Output:

```plaintext
1314777600000
```

**GlideCalendarDateTime - getTime()**

Returns the Unix duration stamp.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideTime</td>
<td>Unix duration stamp in system format based on GMT time.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gs.info(gcdt.getTime());
```

Output:

```plaintext
1970-01-01 08:00:00
```

**GlideCalendarDateTime - getTZOffset()**

Returns the time zone offset in milliseconds.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of milliseconds of the time zone offset</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime();
gcdt.getLocalTime(); // PST local time
gs.info(gcdt.getTZOffset());
```

Output:

```
-28800000
```

**GlideCalendarDateTime - getUserFormattedLocalTime()**

Returns the object's time in local time zone in the user's format.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Object's time in local time and the user's format.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gs.info(gcdt.getUserFormatedLocalTime());
```

Output:

```
2011-08-31 05:00:00
```
**GlideCalendarDateTime - getValue()**

Returns the date and time value stored by the GlideCalendarDateTime object in the internal format, yyyy-MM-dd HH:mm:ss, and the system time zone; UTC by default.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gs.info(gcdt.getValue());
```

Output:

```
2011-08-31 08:00:00
```

**GlideCalendarDateTime - getWeekOfYearLocalTime()**

Returns the number of the week stored by the GlideCalendarDateTime object, expressed in the current user's time zone.

All weeks begin on Sunday. The first week of the year is the week that contains at least one day of the new year. The week beginning Sunday 2015-12-27 is considered the first week of 2016 as that week contains January 1 and 2.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of the current week. The highest week number in a year is either 52 or 53.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-12-01 12:00:00");
gs.info(gcdt.getWeekOfYearUTC());
```

Output:

49

### GlideCalendarDateTime - getWeekOfYearUTC()

Returns the number of the week stored by the GlideCalendarDateTime object, expressed in the UTC time zone.

All weeks begin on Sunday. The first week of the year is the week that contains at least one day of the new year. The week beginning Sunday 2015-12-27 is considered the first week of 2016 as that week contains January 1 and 2.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of the current week in UTC time. The highest week number in a year is either 52 or 53.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-12-01 12:00:00");
gs.info(gcdt.getWeekOfYearUTC());
```

Output:

49
GlideCalendarDateTime - getYearLocalTime()

Returns the year stored by the GlideCalendarDateTime object, expressed in the current user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Four-digit year value in the user's time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime(); //2011
gs.info(gcdt.getYearLocalTime());
```

Output:

2011

GlideCalendarDateTime - getYearUTC()

Returns the year stored by the GlideCalendarDateTime object, expressed in the UTC time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>number</td>
<td>Four-digit year value in the UTC time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime(); //2011
gs.info(gcdt.getYearUTC());
```

Output:
GlideCalendarDateTime - GlideCalendarDateTime()

Instantiates a new GlideCalendarDateTime object with the current date and time in GMT format.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime();
```

GlideCalendarDateTime - GlideCalendarDateTime(GlideCalendarDateTime gcDT)

Instantiates a new GlideCalendarDateTime object set to the time of a specified GlideCalendarDateTime object in GMT format.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gcDT</td>
<td>GlideCalendarDateTime</td>
<td>Object used to set the time of the new object.</td>
</tr>
</tbody>
</table>

```javascript
var start = new GlideCalendarDateTime("2011-01-01 12:00:00");
var end = new GlideCalendarDateTime(start);
gs.info(end);
```

Output:

```
2011-01-01 12:00:00
```

GlideCalendarDateTime - GlideCalendarDateTime(String dateTime)

Instantiates a new GlideCalendarDateTime object from a date and time value in the UTC time zone specified with the format yyyy-MM-dd HH:mm:ss.
### GlideCalendarDateTime - hasDate()

Determines if an object's date is set.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the object's date is set, false otherwise.</td>
</tr>
</tbody>
</table>

```
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gs.info(gcdt.hasDate());
```

Output:

```
true
```

### GlideCalendarDateTime - isDST()

Determines if the object's time uses a daylight saving offset.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gs.info(gcdt.isDST());
```
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the time is daylight saving time, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 00:00:00");
gs.info(gcdt.isDST()); //true
```

Output:

```javascript
true
```

GlideCalendarDateTime - isValid()

Determines if a value is a valid date and time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if value is valid, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 aa:00:00"); //bad
gs.info(gcdt.isValid()); //false
```

Output:

```javascript
false
```

GlideCalendarDateTime - onOrAfter(GlideCalendarDateTime gdt)

Determines if the GlideCalendarDateTime object's data and time occurs on or after the specified GlideCalendarDateTime object's date and time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gdt</td>
<td>GlideCalendarDateTime</td>
<td>Date and time to check against.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the GlideCalendarDateTime object's date and time is on or after the date and time specified by the parameter.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: GlideCalendarDateTime object date and time is on or after the specified object's date and time.</td>
</tr>
<tr>
<td></td>
<td>• false: GlideCalendarDateTime object date and time is before the specified object's date and time.</td>
</tr>
</tbody>
</table>

```javascript
var gdt1 = new GlideCalendarDateTime("2016-05-09 10:11:12");
var gdt2 = new GlideCalendarDateTime("2017-06-12 15:11:12");
gs.info(gdt1.onOrAfter(gdt2));
```

Output

defalse

`GlideCalendarDateTime - onOrBefore(GlideCalendarDateTime gdt)`

Determines if the GlideCalendarDateTime object's data and time occurs on or before the specified GlideCalendarDateTime object's date and time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gdt</td>
<td>GlideCalendarDateTime</td>
<td>Date and time to check against.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the GlideCalendarDateTime object's date and time is on or before the date and time specified by the parameter.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: GlideCalendarDateTime object date and time is on or before the specified object's date and time.</td>
</tr>
<tr>
<td></td>
<td>• false: GlideCalendarDateTime object date and time is after the specified object's date and time.</td>
</tr>
</tbody>
</table>
var gdt1 = new GlideCalendarDateTime("2016-05-09 10:11:12");
var gdt2 = new GlideCalendarDateTime("2017-06-12 15:11:12");
gs.info(gdt1.onOrBefore(gdt2));

Output:

true

GlideCalendarDateTime - setDayOfMonthLocalTime(Number day)
Sets the day of the month to a specified value in the current user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>day</td>
<td>Number</td>
<td>Day of month to change to, from 1 to 31. If this value is greater than the maximum number of days in the month, the value is set to the last day of the month.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gcdt = new GlideCalendarDateTime();
gcdt.setDayOfMonthLocalTime(9);
gs.info(gcdt.getDayOfMonthLocalTime());

Output:

9

GlideCalendarDateTime - setDayOfMonthUTC(Number day)
Sets the day of the month to a specified value in the UTC time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>day</td>
<td>Number</td>
<td>Day of month to change to, from 1 to 31. If this value is greater than the maximum number of days in the month, the value is set to the last day of the month.</td>
</tr>
</tbody>
</table>
**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gcdt = new GlideCalendarDateTime();
gcdt.setDayOfMonthUTC(9);
gs.info(gcdt.getDayOfMonthUTC());

Output:

9

**GlideCalendarDateTime - setDisplayValue(String asDisplayed)**

Sets a date and time value using the current user's display format and time zone.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| asDisplayed | String | Date and time in the current user's display format and time zone.  

The parameter must be formatted using the current user's preferred display format, such as MM-dd-yyyy HH:mm:ss.  

To assign the current date and time to a variable in a workflow script, use `<variable>.setDisplayValue(gs.nowDateTime)`.

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gcdt = new GlideCalendarDateTime("2011-02-02 12:00:00");
gcdt.setDisplayValue("2011-01-01 12:00:00");
gs.info(gcdt.getValue());

Output:

2011-01-01 20:00:00
**GlideCalendarDateTime - setDisplayValue(String dateTime, String format)**

Sets a date and time value using the current user's time zone and the specified date and time format.

This method throws a runtime exception if the date and time format in the dateTime parameter does not match the format parameter. You can retrieve the error message by calling `getErrorMsg()` on the GlideCalendarDateTime object after the exception is caught.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>String</td>
<td>Date and time in the current user's time zone.</td>
</tr>
<tr>
<td>format</td>
<td>String</td>
<td>Format to use to parse the dateTime parameter.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-02-02 12:00:00");
gcdt.setDisplayValue("20-5-2011 12:00:00", "dd-MM-yyyy HH:mm:ss");
gs.info(gcdt.getValue());
```

**Output:**

| 2011-05-20 19:00:00 |

**GlideCalendarDateTime - setDisplayValueInternal(String dateTime)**

Sets a date and time value using the internal format (yyyy-MM-dd HH:mm:ss) and the current user's time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>String</td>
<td>Date and time in internal format</td>
</tr>
</tbody>
</table>
**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-02-02 12:00:00");
gcdt.setDisplayValueInternal("2011-01-01 12:00:00");
gs.info(gcdt.getValue());
```

Output:

```
2011-01-01 20:00:00
```

**GlideCalendarDateTime - setGlideDateTime(GlideCalendarDateTime gcDT)**

Sets the date and time of the current object using an existing GlideCalendarDateTime object.

This method is equivalent to instantiating a new object with a GlideCalendarDateTime parameter.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gcDT</td>
<td>GlideCalendarDateTime</td>
<td>GlideCalendarDateTime object</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var dt1 = new GlideCalendarDateTime("2011-01-01 12:00:00");
var dt2 = new GlideCalendarDateTime("2011-02-02 08:00:00");
dt1.setGlideDateTime(dt2);
gs.info(dt1.getValue());
```

Output:

```
2011-02-02 08:00:00
```
GlideCalendarDateTime - setMonthLocalTime(Number month)

Sets the month stored by the GlideCalendarDateTime object to a specified value using the current user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>month</td>
<td>Number</td>
<td>Month to change to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime();
gcdt.setMonthLocalTime(1);
gs.info(gcdt.getMonthLocalTime());
```

Output:

```
1
```

GlideCalendarDateTime - setMonthUTC(Number month)

Sets the month stored by the GlideCalendarDateTime object to a specified value using the UTC time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>month</td>
<td>Number</td>
<td>Month to change to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime();
gcdt.setMonthUTC(1);
gs.info(gcdt.getMonthUTC());
```
GlideCalendarDateTime - setValue(Object dateTime)

Sets the date and time of the GlideCalendarDateTime object.

This method is equivalent to setInitialValue().

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>Object</td>
<td>Date and time to use. This parameter may be one of several types.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• A string in the UTC time zone and the internal format of yyyy-MM-dd HH:mm:ss: sets the value of the object to the specified date and time. Using the method this way is equivalent to instantiating a new GlideCalendarDateTime object using the GlideCalendarDateTime(value) constructor. If the date and time format used does not match the internal format, the method attempts to set the date and time using other available formats. Resolving the date and time this way can lead to inaccurate data due to ambiguity in the day and month values. When using a non-standard date and time format, use setValue(dt, format) instead.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• A GlideCalendarDateTime object: sets the value of the object to the date and time stored by the GlideCalendarDateTime passed in the parameter. Using the method this way is equivalent to instantiating a new GlideCalendarDateTime object using the GlideCalendarDateTime(g) constructor.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• A Java Date object: sets the value of the object using the value stored by the Java Date object. Using the method this way is equivalent to passing the value returned by the Java Date object getTime() to the GlideCalendarDateTime setNumericValue() method. This method does not accept JavaScript Date objects.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|      |      | - A JavaScript Number: sets the value of the object using the Number value as milliseconds past January 1, 1970 00:00:00 GMT. Using the method this way is equivalent to the `setNumericValue(milliseconds)` method. 
|      |      | - A Java Integer or Long: sets the value of the object using the Integer or Long value as milliseconds past January 1, 1970 00:00:00 GMT. Using the method this way is equivalent to the `setNumericValue(milliseconds)` method. |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-01-01 12:00:00");
gcdt.setValue("2011-02-02 08:00:00");
gs.info(gcdt.getValue());
```

Output:

```
2011-02-02 08:00:00
```

**GlideCalendarDateTime - setValueUTC(String dateTime, String format)**

Sets a date and time value using the UTC time zone and the specified date and time format.

This method throws a runtime exception if the date and time format used in the `dateTime` parameter does not match the `format` parameter. You can retrieve the error message by calling `getErrorMsg()` on the GlideCalendarDateTime object after the exception is caught.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>String</td>
<td>Date and time to use.</td>
</tr>
<tr>
<td>format</td>
<td>String</td>
<td>Format to use.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-01-01 12:00:00");
gcdt.setValueUTC("15-02-2011 08:00:00", "dd-MM-yyyy HH:mm:ss");
gs.info(gcdt.getValue());
```

Output:

2011-02-15 08:00:00

GlideCalendarDateTime - setYearLocalTime(Number year)

Sets the year stored by the GlideCalendarDateTime object to a specified value using the current user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>year</td>
<td>Number</td>
<td>Year to change to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime();
gcdt.setYearLocalTime(2013);
gs.info(gcdt.getYearLocalTime());
```

Output:

2013
GlideCalendarDateTime - setYearUTC(Number year)

Sets the year stored by the GlideCalendarDateTime object to a specified value using the UTC time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>year</td>
<td>Number</td>
<td>Year to change to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime();
gcdt.setYearUTC(2013);
gs.info(gcdt.getYearUTC());
```

Output:

```
2013
```

GlideCalendarDateTime - subtract(GlideCalendarDateTime start, GlideCalendarDateTime end)

Returns the duration difference between two GlideCalendarDateTime values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>GlideCalendarDateTime</td>
<td>Start value.</td>
</tr>
<tr>
<td>end</td>
<td>GlideCalendarDateTime</td>
<td>End value.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDuration</td>
<td>Time between the two values.</td>
</tr>
</tbody>
</table>
var gcdt1 = new GlideCalendarDateTime("2011-08-28 09:00:00");
var gcdt2 = new GlideCalendarDateTime("2011-08-31 08:00:00");
var dur = new GlideDuration();

var dur = GlideCalendarDateTime.subtract(gcdt1, gcdt2); //the difference between gcdt1 and gcdt2
gs.info(dur.getDisplayValue());

Output:

2 Days 23 Hours

GlideCalendarDateTime - subtract(GlideTime time)
Subtracts a specified amount of time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>GlideTime</td>
<td>Time to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
var gtime1 = new GlideTime();
gtime1.setValue("00:00:20");
gcdt.subtract(gtime1);
gs.info(gcdt.getTime());

Output:

1970-01-01 07:59:40

GlideCalendarDateTime - subtract(Number milliseconds)
Subtracts a specified number of milliseconds from the GlideCalendarDateTime object.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>milliseconds</td>
<td>Number</td>
<td>Number of milliseconds to subtract.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-12-07 08:00:00");
gcdt.subtract(1000);
gs.info(gcdt.getValue());
```

**Output:**

```
2011-12-07 07:59:59
```

### GlideCalendarDateTime - toString()

Returns the date and time value stored by the GlideCalendarDateTime object in the internal format, `yyyy-MM-dd HH:mm:ss`, and the system time zone; UTC by default.

This method is equivalent to `getValue()`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Date and time stored by the GlideCalendarDateTime object in the system time zone and format.</td>
</tr>
</tbody>
</table>

```javascript
var gcdt = new GlideCalendarDateTime("2011-08-31 08:00:00");
gs.info(gcdt.toString());
```

**Output:**

```
2011-08-31 08:00:00
```
GlideCertificateEncryption - Global

APIs available for encrypting certificates.

Use these methods to generate a hash for the certificate, sign data using a private key, and generate a message authentication code.

GlideCertificateEncryption - generateMac(String key, String algorithm, String data)

Generates the Message Authentication Code (MAC), which is used to authenticate a message.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Key to use to sign the message. Must be Base64 encoded.</td>
</tr>
<tr>
<td>algorithm</td>
<td>String</td>
<td>Algorithm to use to generate the MAC: HmacSHA256, HmacSHA1, HmacMD5, and so on.</td>
</tr>
<tr>
<td>data</td>
<td>String</td>
<td>Data to process.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>MAC in base64 format.</td>
</tr>
</tbody>
</table>

### Example

```javascript
var mac = new GlideCertificateEncryption;
var key = "sample_key";
key = GlideStringUtil.base64Encode(key);
mac.generateMac(key, "HmacSHA256", "sample_data");
```

### Scoped equivalent

To use the `generateMac()` method in a scoped application, use the corresponding scoped method: `CertificateEncryption - generateMac(String key, String algorithm, String data)`. 
GlideCertificateEncryption - getThumbPrint(String certificateID, String algorithm)
Generates a hash (SHA-1, SHA-256, and so on) for the certificate from Trust Store Cert.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>algorithm</td>
<td>String</td>
<td>SHA-1, SHA-256, and so on</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Thumbprint in base64 format.</td>
</tr>
</tbody>
</table>

Scoped equivalent
To use the getThumbPrint() method in a scoped application, use the corresponding scoped method: CertificateEncryption - getThumbPrint(String certificateID, String algorithm).

GlideCertificateEncryption - getThumbPrintFromKeystore(String certificateID, String alias, String algorithm)
Generates a hash (SHA-1, SHA-256, and so on) for the certificate from the key store entry.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias</td>
<td>String</td>
<td>Alias name for the certificate.</td>
</tr>
<tr>
<td>algorithm</td>
<td>String</td>
<td>SHA-1, SHA-256, and so on.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Thumbprint in base64 format.</td>
</tr>
</tbody>
</table>

Scoped equivalent

To use the `getThumbPrintFromKeystore()` method in a scoped application, use the corresponding scoped method: `CertificateEncryption - getThumbPrintFromKeystore(String certificateID, String alias, String algorithm)`.


Instantiates a GlideCertificateEncryption object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Scoped equivalent


**GlideCertificateEncryption - sign(String certificateID, String alias, String aliaspassword, String algorithm, String datatosign)**

Signs the data using the private key and the given algorithm.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias</td>
<td>String</td>
<td>Private key name.</td>
</tr>
<tr>
<td>aliaspassword</td>
<td>String</td>
<td>Password for the private key.</td>
</tr>
<tr>
<td>datatosign</td>
<td>String</td>
<td>Data to sign.</td>
</tr>
<tr>
<td>algorithm</td>
<td>String</td>
<td>SHA-1, SHA-256, and so on.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Signed data in base64 format.</td>
</tr>
</tbody>
</table>

```javascript
var gce = new GlideCertificateEncryption;
gce.sign("recordID", "alias", "password", "SHA-1", "sign this data");
```

Scoped equivalent
To use the `sign()` method in a scoped application, use the corresponding scoped method: `CertificateEncryption - sign(String certificateID, String alias, String aliaspassword, String algorithm, String datatosign)`.

GlideConversation - Global
The GlideConversation API provides access to information in a Connect message.

**Important:**
Starting with the Paris release, Connect Support no longer receives enhancements or non-priority bug fixes. Consider moving to ServiceNow® Advanced Work Assignment and Agent Chat in workspace to automatically assign chat requests and other work items to agents. For details, see Move from Connect Support to Advanced Work Assignment and Agent Chat.

GlideConversation properties are accessed through a global object (conversation) that is available only in Connect action conditions and scripts. Connect conversations are stored on the Live Group Profile [live_group_profile] table.

GlideConversation - description
The conversation's description.

```javascript
var c = conversation.description;
```

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>String</td>
<td>The conversation's description.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**GlideConversation - document**

The sysID of the document associated with the conversation.

This field is set for feed conversations, and contains the sysID of the record being discussed.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>document</td>
<td>GlideRecord</td>
<td>The document associated with the conversation.</td>
</tr>
</tbody>
</table>

**GlideConversation - name**

The name of the conversation.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the conversation.</td>
</tr>
</tbody>
</table>

```javascript
var c = conversation.name;
```

**GlideConversation - queueEntry**

The queue entry associated with the conversation.

The queue entry is a reference field pointing to the chat_queue_entry table. This field is only available on support conversations. The queue chat entry is used to track the state of the conversation. This property is not changed if an incident record is created from the support conversation using the `newRecord()` method.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queueEntry</td>
<td>GlideRecord</td>
<td>The queue entry associated with the conversation.</td>
</tr>
</tbody>
</table>

```javascript
var c = conversation.queueEntry;
```

**GlideConversation - sys_id**

The sys_id of the conversation.
Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The sys_id of the conversation.</td>
</tr>
</tbody>
</table>

```
var c = conversation.sys_id;
```

GlideConversation - table
The name of the table containing the record being discussed.
This field is set for feed conversations, and is set to the table holding the record.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>The name of the table holding the record being discussed.</td>
</tr>
</tbody>
</table>

```
var c = conversation.table;
```

GlideConversation - type
The conversation type.
This is not the message type.
Connect supports the following types.
- peer - direct conversations, which consist of messages between two users.
- connect - group conversations, which consist of messages between more than two users.
- support - Connect Support conversations.
- feed - record conversations, which consist of messages that correspond to comments and work notes on a specific record.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>The conversation type.</td>
</tr>
</tbody>
</table>

```
var c = conversation.type;
```
GlideCurrencyConfig - Scoped

The GlideCurrencyConfig API provides methods to retrieve various configuration information for FX Currency fields.

You can acquire information for a specific FX Currency field by calling the GlideCurrencyConfig - GlideCurrencyConfig(Object ed) constructor, which associates a specific FX Currency field to the instantiated object. When calling the various GlideCurrencyConfig() methods, if a record exists in the FX Currency Configuration [fx_configuration] table for the FX Currency field, the requested value is passed back. If a record or field is not available, the default value is passed back. You can also acquire the default FX Currency configuration values by calling the GlideCurrencyConfig - GlideCurrencyConfig() constructor.

For additional information on FX Currency configuration, see Setting up and operating FX Currency fields.

This class uses the sn_currency namespace.

GlideCurrencyConfig - getAggregationSource()

Returns the aggregation source value for an FX Currency field.

If called on an existing FX Currency field that has a corresponding FX Currency Configuration [fx_configuration] record, returns the value of the Aggregation Source (aggregation_source) field. Otherwise, if a new record, returns a default value.

For more information on aggregation source, see Specify the table field and its currency display parameters.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>by the GlideElementDescriptor passed in the constructor. If no fx_configuration record, follows the GlideCurrencyConfig() rules.</td>
</tr>
<tr>
<td>GlideCurrencyConfig()</td>
<td>If set, returns the value in the glide.currency2.aggregation_source field in system properties. Otherwise, returns a hard-coded default of reference.</td>
</tr>
</tbody>
</table>

Possible values:
- default
- as_entered
- reference

This code example returns the default aggregation source.

```javascript
var cur_config = new sn_currency.GlideCurrencyConfig();
var curr_agg_source = cur_config.getAggregationSource();
```

This code example returns the aggregation source for a specific FX Currency field.

```javascript
var grCurr = new GlideRecord('my_FXCurrency_table');
var curr_field = grCurr.getElement('currency');
var ed = curr_field.getED();
var cur_config = new sn_currency.GlideCurrencyConfig(ed);
var curr_agg_source = cur_config.getAggregationSource();
```

**GlideCurrencyConfig - getConversionDateSource()**

Returns the field used as the source of the date and time for the currency conversion.

If called on an existing FX Currency field that has a corresponding FX Currency Configuration [fx_configuration] record, returns the value of the Conversion Date Source (conversion_date_source) field. Otherwise, if a new record, returns "null".

For more information on the Conversion Date Source field, see Identify the rate table and date source for currency conversions.
### GlideCurrencyConfig - getConversionDateSource()

Returns the number of digits to display for an FX Currency field.

If called on an existing FX Currency field that has a corresponding FX Currency Configuration \([fx\_configuration]\) record, returns the value of the Display Digits (display_digits) field. Otherwise, if a new record, returns a default value.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var grCurr = new GlideRecord('my_FXCurrency_table');
var curr_field = grCurr.getElement('currency');
var ed = curr_field.getED();
var cur_config = new sn_currency.GlideCurrencyConfig(ed);
var curr_conv_date_src = cur_config.getConversionDateSource();
```
## GlideCurrencyConfig

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of digits to display for the FX Currency field. The returned value depends on the constructor used to instantiate the object.</td>
</tr>
<tr>
<td></td>
<td>• GlideCurrencyConfig(Object ed): If an fx_configuration record exists, returns the value of the display_digits field in the FX Currency Configuration [fx_configuration] record that is associated with the FX Currency field specified by the GlideElementDescriptor passed in the constructor. If no fx_configuration record, follows the GlideCurrencyConfig() rules.</td>
</tr>
<tr>
<td></td>
<td>• GlideCurrencyConfig(): If set, returns the value of the glide.currency2.display_digits property in system properties. Otherwise, returns -1.</td>
</tr>
</tbody>
</table>

This code example returns the default number of display digits.

```javascript
var cur_config = new sn_currency.GlideCurrencyConfig();
var curr_display_digits = cur_config.getDisplayDigits();
```

This code example returns the display digits for a specific FX Currency field.

```javascript
var grCurr = new GlideRecord('my_FXCurrency_table');
var curr_field = grCurr.getElement('currency');
var ed = curr_field.getED();
var cur_config = new sn_currency.GlideCurrencyConfig(ed);
var curr_display_digits = cur_config.getDisplayDigits();
```

### GlideCurrencyConfig - getDisplayValue()

Returns the currency code used to convert the FX Currency value when it appears in lists and reports.

If called on an existing FX Currency field that has a corresponding FX Currency Configuration [fx_configuration] record, returns the value of the Display Value Currency (display_value) field. Otherwise, if a new record, returns a default value.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Currency code to use to display the FX Currency field. The returned value depends on the constructor used to instantiate the object.</td>
</tr>
<tr>
<td></td>
<td>• GlideCurrencyConfig(Object ed): If an fx_configuration record exists, returns the display_value field, which contains the currency code, from the fx_configuration record that is associated with the FX Currency field specified by the GlideElementDescriptor passed in the constructor. If no fx_configuration record, follows the GlideCurrencyConfig() rules.</td>
</tr>
<tr>
<td></td>
<td>• GlideCurrencyConfig(): If set, returns the value of the glide.currency2.display_value property in system properties. Otherwise, returns as_entered.</td>
</tr>
</tbody>
</table>

Possible values:

• default: Global default. This is the value in the glide.currency2.display_digits property.
• as_entered: Currency value as entered by the user.
• in_session_currency: Currency value converted to the session currency. Session currency is based on the user's locale.
• in_reference_currency: Reference value if computed and stored. Otherwise, the currency value converted to the reference currency.

This code example returns the default currency code.

```javascript
var cur_config = new sn_currency.GlideCurrencyConfig();
var curr_display_value = cur_config.getDisplayValue();
```

This code example returns the display currency code for a specific FX Currency field.

```javascript
var grCurr = new GlideRecord('my_FXCurrency_table');
var curr_field = grCurr.getElement('currency');
var ed = curr_field.getED();
var cur_config = new sn_currency.GlideCurrencyConfig(ed);
var curr_display_value = cur_config.getDisplayValue();
```

### GlideCurrencyConfig - getRateFilterRateTableField()

Returns the rate table field used as a rate table filter for an FX Currency field.
If called on an existing FX Currency field that has a corresponding FX Currency Configuration \([fx\_configuration]\) record, returns the value of the Rate Table Field \(rate\_filter\_rate\_table\_field\). Otherwise, if a new record, returns a default value.

For more information on rate table filters, see Select the rate and target table fields used for filtering.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Field in the rate table used to filter the rate table records. The returned value depends on the constructor used to instantiate the object.</td>
</tr>
</tbody>
</table>

- GlideCurrencyConfig(Object ed): If an \(fx\_configuration\) record exists, returns the \(rate\_filter\_rate\_table\_field\) from the \(fx\_configuration\) record that is associated with the FX Currency field specified by the GlideElementDescriptor passed in the constructor. If no \(fx\_configuration\) record, follows the GlideCurrencyConfig() rules.

- GlideCurrencyConfig(): Returns null.

This code example returns the rate table field used to filter rate table records for a specific FX Currency field.

```javascript
var grCurr = new GlideRecord('my_FXurrency_table');
var curr_field = grCurr.getElement('currency');
var ed = curr_field.getED();
var cur_config = new sn_currency.GlideCurrencyConfig(ed);
var curr_reference_currency = cur_config.getRateFilterRateTableField();
```

**GlideCurrencyConfig - getRateFilterTargetTableField()**

Returns the target table field used to filter the rate table records for an FX Currency field.

If called on an existing FX Currency field that has a corresponding FX Currency Configuration \([fx\_configuration]\) record, returns the value of the Target Table Field \(rate\_filter\_target\_table\_field\). Otherwise, if a new record, returns a default value.
For more information on rate table filters, see Select the rate and target table fields used for filtering.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Field in the target table used/to use to filter the rate table. The returned value depends on the constructor used to instantiate the object.</td>
</tr>
<tr>
<td></td>
<td>• GlideCurrencyConfig(Object ed): If an fx_configuration record exists, returns the rate_filter_target_table_field from the fx_configuration record that is associated with the FX Currency field specified by the GlideElementDescriptor passed in the constructor. If no fx_configuration record, follows the GlideCurrencyConfig() rules.</td>
</tr>
<tr>
<td></td>
<td>• GlideCurrencyConfig(): Returns null.</td>
</tr>
</tbody>
</table>

This code example returns the default rate filter, target table field.

```javascript
var cur_config = new sn_currency.GlideCurrencyConfig();
var curr_rate_filter = cur_config.getRateFilterTargetTableField();
```

This code example returns the rate filter, target table field for a specific FX Currency field.

```javascript
var grCurr = new GlideRecord('my_FXCurrency_table');
var curr_field = grCurr.getElement('currency');
var ed = curr_field.getED();
var cur_config = new sn_currency.GlideCurrencyConfig(ed);
var curr_rate_filter = cur_config.getRateFilterTargetTableField();
```

**GlideCurrencyConfig - getRateTable()**

Returns the rate table used to convert currency for an FX Currency field.

If called on an existing FX Currency field that has a corresponding FX Currency Configuration [fx_configuration] record, returns the value of the Rate Table Field (rate_filter_rate_table_field). Otherwise, if a new record, returns a default value.
For more information on FX Currency rate tables, see Identify the rate table and date source for currency conversions.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the rate table used to perform the currency conversion. The returned value depends on the constructor used to instantiate the object.</td>
</tr>
<tr>
<td></td>
<td>• GlideCurrencyConfig(Object ed): If an fx_configuration record exists, returns the rate_table_field field from the fx_configuration record that is associated with the FX Currency field specified by the GlideElementDescriptor passed in the constructor. If no fx_configuration record, follows the GlideCurrencyConfig() rules.</td>
</tr>
<tr>
<td></td>
<td>• GlideCurrencyConfig(): If set, returns the value of the glide.currency2.system_rate_table property from system properties. Otherwise, returns fx_system_rate.</td>
</tr>
</tbody>
</table>

Returns the default rate table.

```
var cur_config = new sn_currency.GlideCurrencyConfig();
var curr_rate_table = cur_config.getRateTable();
```

Returns the rate table used for a specific FX Currency field.

```
var grCurr = new GlideRecord('my_FXCurrency_table');
var curr_field = grCurr.getElement('currency');
var ed = curr_field.getED();
var cur_config = new sn_currency.GlideCurrencyConfig(ed);
var curr_rate_table = cur_config.getRateTable();
```

**GlideCurrencyConfig - getReferenceCurrency()**

Returns the reference currency for an FX Currency field.

If called on an existing FX Currency field that has a corresponding FX Currency Configuration [fx_configuration] record, returns the value of the Reference
Currency (reference_currency) field. Otherwise, if a new record, returns a default value.

For more information on reference currency, see Set the reference currency.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Reference currency as an ISO currency code. The returned value depends on the constructor used to instantiate the object.</td>
</tr>
</tbody>
</table>

- GlideCurrencyConfig(Object ed): If an fx_configuration record exists, returns the reference_currency field from the fx_configuration record that is associated with the FX Currency field specified by the GlideElementDescriptor passed in the constructor. If no fx_configuration record, follows the GlideCurrencyConfig() rules.

- GlideCurrencyConfig(): If set, returns the value of the glide.currency2.default_reference_currency system property. Otherwise the method applies the following rules, in the specified order, to determine the reference currency code:
  - If the glide.system.reference_currency system property is set to true, uses the value in the glide.system.reference_currency.code system property.
  - The value in the glide.system.locale system property (which implies a currency).
  - Default locale of the Java Virtual Machine on which the instance is running.

This code example returns the default reference currency.

```javascript
var cur_config = new sn_currency.GlideCurrencyConfig();
var curr_reference_currency = cur_config.getReferenceCurrency();
```

This code example returns the reference currency for a specific FX Currency field.

```javascript
var grCurr = new GlideRecord('my_FXCurrency_table');
var curr_field = grCurr.getField('currency');
```
```javascript
var ed = curr_field.getED();
var cur_config = new sn_currency.GlideCurrencyConfig(ed);
var curr_reference_currency = cur_config.getReferenceCurrency();
```

**GlideCurrencyConfig - getReferenceCurrencySource()**

Returns the reference currency source for an FX Currency field.

If called on an existing FX Currency field that has a corresponding FX Currency Configuration [fx_configuration] record, returns the value (dot-walk path) of the Reference Currency Source (reference_currency_source) field. Otherwise, if a new record, returns a default value.

For more information on reference currency source, see Set the reference currency.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Dot-walk path of the field used/to use as the reference currency. The returned value depends on the constructor used to instantiate the object.</td>
</tr>
</tbody>
</table>

- GlideCurrencyConfig(Object ed): If an fx_configuration record exists, returns the reference_currency_source field from the fx_configuration record that is associated with the FX Currency field specified by the GlideElementDescriptor passed in the constructor. If no fx_configuration record, follows the GlideCurrencyConfig() rules.
- GlideCurrencyConfig(): Returns null.

This code example returns the default reference currency source.

```javascript
var cur_config = new sn_currency.GlideCurrencyConfig();
var curr_reference_currency = cur_config.getReferenceCurrency();
```

This code example returns the reference currency source for a specific FX Currency field.
var grCurr = new GlideRecord('my_FXCurrency_table');
var curr_field = grCurr.getElement('currency');
var ed = curr_field.getED();
var cur_config = new sn_currency.GlideCurrencyConfig(ed);
var curr_reference_currency = cur_config.getReferenceCurrencySource();

GlideCurrencyConfig - GlideCurrencyConfig()

Instantiates a GlideCurrencyConfig object that enables the retrieval of global default FX Currency configuration information.

When calling the available methods on this type of GlideCurrencyConfig object, the return values are the FX Currency global defaults from system properties. If a global default is not defined in system properties, a hard-coded default is returned (this can also be an empty value if not configured). This object does not provide configuration information for a specific FX Currency field. To retrieve specific FX Currency field information, use the GlideCurrencyConfig - GlideCurrencyConfig(Object ed) constructor.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
var cur_config = new sn_currency.GlideCurrencyConfig();
```

GlideCurrencyConfig - GlideCurrencyConfig(Object ed)

Instantiates a GlideCurrencyConfig object that enables the retrieval of the configuration information for a specific FX Currency field.

When calling the available methods on this type of GlideCurrencyConfig object, the return values are from the FX Currency Configuration [fx_configuration] record associated with the FX Currency field, if a record exists. If a record does not exist, default values are returned.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ed</td>
<td>GlideElementDescriptor</td>
<td>GlideElementDescriptor object of the FX Currency field to associate with the GlideCurrencyConfig object.</td>
</tr>
</tbody>
</table>

Example

```javascript
var grCurr = new GlideRecord('my_FXCurrency_table');
var curr_field = grCurr.getElement('currency');
var ed = curr_field.getED();
var cur_config = new sn_currency.GlideCurrencyConfig(ed);
```

**GlideCurrencyConverter - Scoped**

The GlideCurrencyConverter API provides methods to convert one currency value to another, such as converting US dollars into European Euro.

You can instantiate the GlideCurrencyConverter object and define the source and destination currencies during instantiation using GlideCurrencyConverter(from, to). You can also instantiate the object without these values and define them later using the setFromCurrency() and setToCurrency() methods. These values and the amount to convert must be set before calling the convert() method to perform the currency conversion. To set the amount to convert, use the setAmount() method.

The GlideCurrencyConverter() API also provides optional methods that enable you to:

- Set the date and time for which to perform the conversion, setDateTime(). By setting the date and time, the rate that is used in the conversion calculation is that for the specified date and time, instead of the default of the current date and time.

- Set the rate table to use in the conversion, setRateTable(). By default the conversion uses the fx_system_rate table however, you can define custom rate tables for your instance. For additional information on creating rate tables, see Add conversion rates using a custom rate table.

This class runs in the sn_currency namespace.

**Scoped GlideCurrencyConverter - convert()**

Executes the currency converter.
Call this method after calling other GlideCurrencyConverter methods that construct the currency conversion, such as setAmount(), setRateTable(), and setDate().

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>If the conversion is successful, returns CurrencyExchangeValue object. If unsuccessful, returns &quot;null&quot;.</td>
</tr>
</tbody>
</table>

```javascript
CurrencyExchangeValue {
  Rate: Number,
  OriginalAmount: Number,
  Amount: Number,
  fromCurrency: String,
  toCurrency: String,
  rateSysId: String
}
```

- **Rate**: Number. Exchange rate used in the conversion.
- **OriginalAmount**: Number. Source currency amount.
- **Amount**: Number. Converted amount (OriginalAmount * Rate).
- **fromCurrency**: String. Three-letter ISO 3166 country code of the source currency.
- **toCurrency**: String. Three-letter ISO 3166 country code of the converted currency.
- **rateSysId**: String. Sys_id of the rate table record used to calculate the conversion.

Example

```javascript
var conv = new sn_currency.GlideCurrencyConverter('EUR', 'USD');
conv.setAmount(100);
gs.info(conv.convert());
```

Output:
CurrencyExchangeValue{Rate = 1.0777, OriginalAmount = 100, Amount = 107.7700, fromCurrency = 'EUR', toCurrency = 'USD', rateSysId = '2ed537fcb271937adb'}

Scoped GlideCurrencyConverter - GlideCurrencyConverter()
Instantiates a GlideCurrencyConverter object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Example

```java
var conv = new sn_currency.GlideCurrencyConverter();
```

Scoped GlideCurrencyConverter - GlideCurrencyConverter(String from, String to)
Instantiates a GlideCurrencyConverter object and sets the source and destination country codes to use in the currency conversion.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>from</td>
<td>String</td>
<td>Three-letter ISO 3166 country code of the source currency.</td>
</tr>
<tr>
<td>to</td>
<td>String</td>
<td>Three-letter ISO 3166 country code of the converted currency.</td>
</tr>
</tbody>
</table>

Example

```java
var conv = new sn_currency.GlideCurrencyConverter('EUR', 'USD');
```

Scoped GlideCurrencyConverter - setAmount(String amount)
Sets the amount of currency to convert.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>amount</td>
<td>String</td>
<td>Currency amount to convert. This value must be unformatted except for a decimal point to denote fractional currency. For example, 1234.56 is valid, 1,234.56 is invalid.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
var conv = new sn_currency.GlideCurrencyConverter('EUR', 'USD');
conv.setAmount(100);
```

 Scoped GlideCurrencyConverter - setDateTime(Object date)

Sets the currency conversion date and time.

This date and time determines the conversion rate that is used to convert the currency. If this method is not called before the GlideCurrencyConverter.convert() method, the conversion is performed using the rate for the current date/time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>GlideDateTime</td>
<td>Date/time for which to calculate the currency conversion. This value determines the rate that is used in the conversion.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example
var conv = new sn_currency.GlideCurrencyConverter('EUR', 'USD');
conv.setAmount(100);
var gd = new GlideDateTime("2019-01-03 11:00:00");
conv.setDateTime(gd);
gs.info(conv.convert());

Output:

```
CurrencyExchangeValue{fOriginalAmount=100, fOriginalCurrency='EUR', fRate=1.061,
  fAmount=106.1, fCurrency='USD', fRateSysId='4555525f5553445f31303030313031'}
```

**Scoped GlideCurrencyConverter - setFromCurrency(String from)**

Sets the country code of the source currency.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>from</td>
<td>String</td>
<td>Three-letter ISO 3166 country code of the source currency.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Example**

```
var conv = new sn_currency.GlideCurrencyConverter().setFromCurrency('FRA');
```

**Scoped GlideCurrencyConverter - setRateTable(String rateTable)**

Defines the rate table to use in the currency conversion.

If this method is not called before the GlideCurrencyConverter.convert() method is called, the conversion is performed using the fx_system_rate table. All custom rate tables must extend the fx_conversion_rate table. For additional information on creating rate tables, see Add conversion rates using a custom rate table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rateTable</td>
<td>String</td>
<td>Name of the rate table to use in the currency conversion.</td>
</tr>
</tbody>
</table>
**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Example**

```javascript
var conv = new sn_currency.GlideCurrencyConverter('EUR', 'USD');
conv.setRateTable(custom_rate_table);
```

**Scoped GlideCurrencyConverter - setToCurrency(String to)**

Sets the country code of the destination currency.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>to</td>
<td>String</td>
<td>Three-letter ISO 3166 country code of the source currency.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Example**

```javascript
var conv = new sn_currency.GlideCurrencyConverter().setToCurrency('USA');
```

**GlideCurrencyExchangeValue - Scoped**

Provides access to information related to FX Currency conversions.

Refer to the GlideCurrencyConverter() API for methods that return a GlideCurrencyExchangeValue object.

**Note:** There is no constructor for this class.

**Scoped GlideCurrencyExchangeValue - getAmount()**

Returns the currency amount after the conversion.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td></td>
<td>Currency amount after the conversion.</td>
</tr>
</tbody>
</table>

This example assumes a conversion rate of 1 Euro = 0.92 US dollars.

```javascript
var conv = new sn_currency.GlideCurrencyConverter('EUR', 'USD');
conv.setAmount(100);
var convertValues = conv.convert();
gs.info('Exchanged Amount: ' + convertValues.getAmount());
```

Output:

```
Exchanged Amount: $92.00
```

### Scoped GlideCurrencyExchangeValue - `getCurrency()`

Returns the converted to currency type.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td></td>
<td>Three-letter ISO currency code of the converted to currency.</td>
</tr>
</tbody>
</table>

Example

```javascript
var convertValues = new sn_currency.GlideCurrencyConverter('USD', 'JPY');
convertValues.setAmount(100);
gs.info('Exchanged Currency (To): ' + convertValues.convert().getCurrency());
```

Output:
Exchanged Currency (To): JPY

Scoped GlideCurrencyExchangeValue - getOriginalCurrency()
Returns the original, pre-conversion currency type.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Three-letter ISO currency code of the currency converted from.</td>
</tr>
</tbody>
</table>

Example

```javascript
var converter = new sn_currency.GlideCurrencyConverter('USD', 'JPY');
converter.setAmount(999.999)
var convertValues = converter.convert();
gs.info('Original Amount: ' + convertValues.getOriginalCurrency());
```

Output:

Original Amount: USD

Scoped GlideCurrencyExchangeValue - getOriginalValue()
Returns the original, pre-conversion amount.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Original amount before conversion.</td>
</tr>
</tbody>
</table>
Example

```javascript
var converter = new sn_currency.GlideCurrencyConverter('USD', 'JPY');
converter.setAmount(999.999)
var convertValues = converter.convert();
gs.info('Original Amount: ' + convertValues.getOriginalAmount());
```

Output:

Original Amount: 999.999

Scoped GlideCurrencyExchangeValue - getRate()
Returns the exchange rate used during the conversion.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Exchange rate used in the currency conversion.</td>
</tr>
</tbody>
</table>

Example

```javascript
var converter = new sn_currency.GlideCurrencyConverter('EUR', 'USD');
converter.setAmount(999.999)
var convertValues = converter.convert();
gs.info('Original Amount: ' + convertValues.getRate());
```

Output:

Exchange rate: 1.0559

Scoped GlideCurrencyExchangeValue - getRateSysId()
Returns the sys_id of the record in the rate table used in the currency conversion.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the rate table used in the currency conversion.</td>
</tr>
</tbody>
</table>

Example

```javascript
var converter = new sn_currency.GlideCurrencyConverter('USD', 'JPY');
converter.setAmount(999.999);
var convertValues = converter.convert();
gs.info('Rate table sys_id: ' + convertValues.getRateSysId());
```

Output:

Original Amount: 99ebb4156fa831005be8883e6b3ee4b9

**GlideCurrencyFormatter - Scoped**

The GlideCurrencyFormatter API provides methods to format FX Currency strings.

When instantiating the GlideCurrencyFormatter object, you define the format of the output results for this object. Once set, the same format is used each time you call the GlideCurrencyFormatter.format() method of that object. The GlideCurrencyFormatter() and format() methods are the only two methods that you must call to format FX Currency.

The GlideCurrencyFormatter() API also provides optional methods that enable you to:

- Set the locale of the currency using setLocale(). By setting the locale, the formatter takes into consideration any formatting that is specific to that locale, such as whether a decimal point or decimal comma is used to denote fractions of currency amount. Default: System locale

- Set the minimum number of fractional digits to display using setMinFractionDigits(). Defines the minimum number of digits to the right of the decimal point/decimal comma to include. By default, all digits passed in are displayed. If the passed in amount has fewer digits than the specified minimum, the result is padded with "0".

- Set the maximum number of fractional digits to display using setMaxFractionDigits(). Defines the maximum number of digits to the right of the decimal point/decimal comma to include. By default, all digits passed in are displayed. If the passed in amount has a greater number of digits than the specified maximum, the additional digits are truncated during the process of rounding to the number of maximum fraction digits. Rounding is performed to the closest right-most digit. For example, if the maximum fraction digit is set to
2 and the passed in currency value is 123.456, the formatted value is 123.46. If the currency value is 23.122, the formatted value is 23.12. If the maximum is set to less than the minimum, the minimum value is used.

This class runs in the `sn_currency` namespace.

**Scoped GlideCurrencyFormatter - GlideCurrencyFormatter(String formatString)**

Instantiates a GlideCurrencyFormatter object and defines the format string to use to format FX Currency.

This method does not perform the actual string format. You must call the `GlideCurrencyFormatter.format()` method to format the currency.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>formatString</td>
<td>String</td>
<td>Optional. Format string to use to format any currency passed into the formatter. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• %s: Replaced by the currency symbol associated with the country code specified in the <code>format()</code> call.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• %v: Replaced by the currency amount, such as 123.45.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• %c: Replaced by the ISO currency code specified in the <code>format()</code> call, such as USD or EUR.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• %l: Replaced with the passed in value, no formatting performed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• %p: Replaced by the percent sign (%).</td>
</tr>
</tbody>
</table>

For example, if the format string is `%s%v%c` and the value to format is 123.45 in US dollars, the returned formatted string is $123.45 USD. If the format string is `%s%l%c` and the value string to format is '56M' in Euros, the returned formatted string is €56M EUR.

**Note:** The same format string is used each time the `format()` method is called on this object.
var formatString = '%s%v%c';
var exchangeValue = new sn_currency.GlideCurrencyFormatter(formatString);

Scoped GlideCurrencyFormatter - format(String value, String currencyCode)
Formats a specified currency value using the specified currency code.

The resulting format of the currency is also dependent on the values passed in when the GlideCurrencyFormatter object was initially instantiated.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>Currency amount to format. The amount can only contain a decimal point to denote the fractional amount, commas are not supported. Valid values: 123.45 or 2436.23 Invalid values 123,45 or 2,134.56</td>
</tr>
<tr>
<td>currencyCode</td>
<td>String</td>
<td>Three-letter ISO currency code to use when formatting the currency.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Formatted currency value.</td>
</tr>
</tbody>
</table>

Example

```javascript
var amount = '123456.123456789';
var currencyCode = 'USD';
var formatString = '%s%v%c';
var exchangeValue = new sn_currency.GlideCurrencyFormatter(formatString);
gs.info('Formatted currency: ' + exchangeValue.setMaxFractionDigits(2).format(amount, currencyCode));
```

Output:

Formatted currency: $123,456.12USD
Scoped GlideCurrencyFormatter - setLocale(String language, String country)

Sets the default language and country, which constitutes the locale, for the currency formatter.

If you do not call this method before calling the GlideCurrencyFormatter.format() method, the locale defaults to the current session locale.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>language</td>
<td>String</td>
<td>Two-letter ISO 639 language code, such as en, sp, and fr.</td>
</tr>
<tr>
<td>country</td>
<td>String</td>
<td>Two-letter ISO 3166 country code, such as US, ES, and FR.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows setting the currency formatter to France.

```javascript
var amount = '123456.123456789';
var currencyCode = 'EUR';
var formatString = '%s%v%c';
var exchangeValue = new sn_currency.GlideCurrencyFormatter(formatString);
exchangeValue.setLocale("fr", "FR"); // Language = fr Country = FR
gs.info('Formatted currency: ' + exchangeValue.setMaxFractionDigits(2).format(amount, currencyCode));
```

Output:

Formatted currency: €123 456,12EUR

Scoped GlideCurrencyFormatter - setMaxFractionDigits(Number maxFractionDigits)

Sets the maximum number of digits to the right of the decimal point/decimal comma to return.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxFractionDigits</td>
<td>Number</td>
<td>Maximum number of fraction digits to return. If this value is set to something less than the minFractionDigits value, it is overridden by minFractionDigits.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
var amount = '123456.123456789';
var currencyCode = 'USD';
var formatString = '%s%v%c';
var exchangeValue = new sn_currency.GlideCurrencyFormatter(formatString);
gs.info('Formatted currency: ' + exchangeValue.setMaxFractionDigits(2).format(amount, currencyCode));
```

Output:

Formatted currency: $123,456.12USD

Scoped GlideCurrencyFormatter - setMinFractionDigits(Number minFractionDigits)

Sets the minimum number of fraction digits (right of the decimal point/decimal comma) to return.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>minFractionDigits</td>
<td>Number</td>
<td>Minimum number of fraction digits to return.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var amount = '123456.12';
var currencyCode = 'USD';
var formatString = '%s%v%c';
var exchangeValue = new sn_currency.GlideCurrencyFormatter(formatString);
gs.info('Formatted currency: ' + exchangeValue.setMinFractionDigits(3).format(amount, currencyCode));

Output:
Formatted currency: $123,456.120USD

GlideCurrencyParser - Scoped
The GlideCurrencyParser API provides methods to parse FX Currency values to detect locale-based formatting.

The methods in this class parse a specified string into a GlideCurrencyValue object enabling you to obtain locale-based currency formatting. Before calling the GlideCurrencyParser.parse() method, which actually performs the parse, you must set the currency code to use during the parse. You can set the currency code by either passing it in the GlideCurrencyParser.parse() method call or by calling GlideCurrencyParser.setDefaultCurrencyCode().

Setting the locale is optional. If you do not set it through the GlideCurrencyParser.setLocale() method, the currency amount passed into GlideCurrencyParser.parse() must be unformatted.

This class runs in the sn_currency namespace.

Scoped GlideCurrencyParser - GlideCurrencyParser()
Instantiates a GlideCurrencyParser object that provides methods to parse a string into a GlideCurrencyValue object.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

Example
var parser = new sn_currency.GlideCurrencyParser();
Scoped GlideCurrencyParser - parse(String value)

Parses a specified string into a GlideCurrencyValue object to obtain locale-based currency formatting.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| value | String| String to parse. You can prepend this string with the ISO currency code to use when parsing the string (semicolon separated). If the currency code is not set prior to calling the parse() method, it defaults to the current session currency code. This method supports both decimal point and decimal comma notation. For example: "123.45", "USD;123.45", "123.45", "EUR;123.45"

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideCurrencyValue</td>
<td>Parsed value information. Data type: Object</td>
</tr>
</tbody>
</table>

Passes the currency code in value string.

```javascript
var gp = new sn_currency.GlideCurrencyParser();
gp.setLocale("fre", "FRA");
var cv = gp.parse("USD;1234.56");
gs.info(cv);
```

Output:

USD;1234.56

Uses the setDefaultCurrencyCode property to define currency code.

```javascript
var amount = "1234,56"
var gp = new sn_currency.GlideCurrencyParser();
gp.setDefaultCurrencyCode("USD");
var cv = gp.parse(amount);
gs.info(cv);
```

Output:

USD;1234.56
Scoped GlideCurrencyParser - setDefaultCurrencyCode(String currencyCode)
Sets the currency code for the associated GlideCurrencyParser object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>currencyCode</td>
<td>String</td>
<td>Three-letter ISO currency code to use as the default in the associated GlideCurrencyParser object.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
var amount = “123.45”
var gp = new sn_currency.GlideCurrencyParser();
    gp.setDefaultCurrencyCode("USD");
var cv = gp.parse(amount);
```

Scoped GlideCurrencyParser - setLocale(String language, String country)
Sets the default language and country codes, which constitute the locale, on the associated GlideCurrencyParser object.

If you do not call this method before calling the GlideCurrencyParser.parse() method, the locale defaults to the current session locale.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>language</td>
<td>String</td>
<td>Three-letter ISO 639 language code, such as eng, spa, and fre.</td>
</tr>
<tr>
<td>country</td>
<td>String</td>
<td>Three-letter ISO 3166 country code, such as USA, ESP, and FRA.</td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Example**

```javascript
var amount = "USD;123.45"
var gp = new sn_currency.GlideCurrencyParser();
gp.setLocale("spa", "ESP");
var cv = gp.parse(amount);
```

**GlideCurrencyValue - Scoped**

Provides methods that return the currency code and original currency from an associated GlideCurrencyParser.parse() call.

ℹ️ **Note:** There is no constructor for this class.

**Scoped GlideCurrencyValue - getAmount()**

Returns the currency amount originally passed into the associated GlideCurrencyParser.parse() call.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Original currency passed into the parse() method.</td>
</tr>
</tbody>
</table>

**Example**

```javascript
var gp = new sn_currency.GlideCurrencyParser();
gp.setLocale("eng", "USA");
var cv = gp.parse("USD;1234.56");
gs.info('Original currency amount: ' + cv.getAmount());
```

**Output:**
Original currency amount: 1234.56

 Scoped GlideCurrencyValue - getCurrencyCode()
 Returns the currency code used to parse the currency in the associated GlideCurrencyParser.parse() call.

 Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

 Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Three-letter ISO currency code used to parse the currency.</td>
</tr>
</tbody>
</table>

 Example

 ```javascript
 var gp = new sn_currency.GlideCurrencyParser();
gp.setLocale(eng, USA);
var cv = gp.parse("USD";"1234.56");
gs.info('Parsed currency code: ' + cv.getCurrencyCode());
```

 Output:  
 Parsed currency code: USD

 GlideDate - Scoped
 The scoped GlideDate class provides methods for performing operations on GlideDate objects, such as instantiating GlideDate objects or working with GlideDate fields.

 Scoped GlideDate - getByFormat(String format)
 Gets the date in the specified date format.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>format</td>
<td>String</td>
<td>Desired date format using Java <code>SimpleDateFormat</code>. For example, &quot;dd-MM-yyyy&quot; to get the day, month, and year, or &quot;EEEE&quot; to get the day of the week.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Date value for the GlideDate object in the specified format.</td>
</tr>
</tbody>
</table>

```javascript
var gd = new GlideDate();
gd.setValue('2021-04-21');
gs.info(gd.getByFormat("dd-MM-yyyy"));
```

Output:

```
21-04-2021
```

### Scoped GlideDate - getDayOfMonthNoTZ()

Gets the day of the month stored by the GlideDate object, expressed in the UTC time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The day of the month in the UTC time zone, from 1 to 31.</td>
</tr>
</tbody>
</table>

```javascript
// Today's date is 2016-05-13
var gd = new GlideDate();
gs.info(gd.getDayOfMonthNoTZ());
```

Output: 13
Scoped GlideDate - getDisplayValue()

 Gets the date in the current user's display format and time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Date value for the GlideDate object in the current user's display format and time zone. Keep in mind when designing business rules or script includes that this method may return values in different formats for different users.</td>
</tr>
</tbody>
</table>

```javascript
var gd = new GlideDate();
gd.setValue('2021-04-21');
gs.info(gd.getDisplayValue());
```

Output:

```
2021-04-21
```

Scoped GlideDate - getDisplayValueInternal()

 Gets the date in the internal format (yyyy-MM-dd) and current user's timezone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Date value for the GlideDate object in the current user's time zone. Format: yyyy-MM-dd</td>
</tr>
</tbody>
</table>
```javascript
var gd = new GlideDate();
gs.info(gd.getDisplayValueInternal());
```

Output:

2021-04-21

**Scoped GlideDate - getMonthNoTZ()**

Gets the month stored by the GlideDate object, expressed in the UTC time zone.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The numerical value of the month from 1 to 12.</td>
</tr>
</tbody>
</table>

```javascript
//Today's date is 2016-05-13
var gd = new GlideDate();
gs.info(gd.getMonthNoTZ());
```

Output: 5

**Scoped GlideDate - getValue()**

Gets the date in the internal format (yyyy-MM-dd) and the system time zone (UTC by default).

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The date value for the GlideDate object in the system time zone.</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format: yyyy-MM-dd</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gd = new GlideDate();
gd.setValue('2021-04-21');
gs.info(gd.getValue());
```

**Output:**

```
2021-04-21
```

### Scoped GlideDate - getYearNoTZ()

 Gets the year stored by the GlideDate object, expressed in the UTC time zone.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The numerical value of the year.</td>
</tr>
</tbody>
</table>

```javascript
//Today's date is 2016-05-13
var gd = new GlideDate();
gs.info(gd.getYearNoTZ());
```

**Output:** 2016

### Scoped GlideDate - GlideDate()

 Creates a GlideDate object with the current date time.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
//Today's date is April 21, 2021
var gd = new GlideDate();
gs.info(gd.getValue());

Output:
2021-04-21

**Scoped GlideDate - setDisplayValue(String asDisplayed)**
Sets a date value using the current user's display format and time zone.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>asDisplayed</td>
<td>String</td>
<td>Date in the current user's display format and time zone. Format: Must be formatted using the current user's preferred display format, such as yyyy-MM-dd.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gd = new GlideDate();
gd.setDisplayValue("2011-01-01");
gs.info(gd.getValue());

Output: 2011-01-01

**Scoped GlideDate - setValue(String o)**
Sets the date of the current GlideDate object in internal format (yyyy-MM-dd) and the system time zone (UTC by default).

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>o</td>
<td>String</td>
<td>Date and time to set in the current GlideDate object. Format: yyyy-MM-dd</td>
</tr>
</tbody>
</table>
 Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gd = new GlideDate();
gd.setValue('2015-01-01');
gs.info(gd.getValue());
```

Output: 2015-01-01

Scoped GlideDate - subtract(GlideDate start, GlideDate end)

Gets the duration difference between two GlideDate values.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>GlideDate</td>
<td>The start value.</td>
</tr>
<tr>
<td>end</td>
<td>GlideDate</td>
<td>The end value.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDuration</td>
<td>The duration between the two values.</td>
</tr>
</tbody>
</table>

```javascript
var sgd1 = new GlideDate();
sgd1.setDisplayValue('2014-07-18');
var sgd2 = new GlideDate();
sgd2.setDisplayValue('2014-07-19');

var duration = GlideDate.subtract(sgd1, sgd2);
gs.info(duration.getDisplayValue());
```

Output:

1 Day
GlideDateTime - Global

The GlideDateTime class provides methods for performing operations on GlideDateTime objects, such as instantiating GlideDateTime objects or working with glide_date_time fields.

Use the GlideDateTime methods to perform date-time operations, such as instantiating a GlideDateTime object, performing date-time calculations, formatting a date-time, or converting between date-time formats.

GlideDateTime - add(Number milliseconds)

Adds a specified number of milliseconds to the GlideDateTime object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>milliseconds</td>
<td>Number</td>
<td>The number of milliseconds to add</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.print(gdt.getNumericValue());
gdt.add(10);
gs.print(gdt.getNumericValue());
```

Output:

```
1314777600000
1314777600010
```

Scoped equivalent

To use the add() method in a scoped application, use the corresponding scoped method: Scoped GlideDateTime - add(Number milliseconds).

GlideDateTime - add(GlideTime time)

Adds a GlideTime object to the current GlideDateTime object.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>GlideTime</td>
<td>GlideTime object whose time value to add to the specified GlideDateTime object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to add 20 seconds to the time set in the gdt GlideDateTime object.

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
var gtime1 = new GlideTime();
gtime1.setValue("00:00:20");
gdt.add(gtime1);
gs.print(gdt.toString());
```

Output:

```
2011-08-31 08:00:20
```

### Scoped equivalent

To use the `add()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - add(GlideTime gd)**.

**GlideDateTime - addDays(Number days)**

Adds a specified number of days to the current GlideDateTime object. A negative parameter subtracts days.

Use `addDaysLocalTime()` and `addDaysUTC()` instead of this method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>days</td>
<td>Number</td>
<td>The number of days to add. Use a negative number to subtract.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addDays(-1);
gs.print(gdt.getDate());
```

Output: 2011-08-30

**GlideDateTime - addDaysLocalTime(Number days)**

Adds a specified number of days to the current GlideDateTime object. A negative parameter subtracts days.

The method determines the local date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts days using the local date and time values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>days</td>
<td>Number</td>
<td>The number of days to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addDaysLocalTime(-1);
gs.print(gdt.getLocalDate());
```

Output: 2011-08-30

**Scoped equivalent**

To use the `addDaysLocalTime()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - addDaysLocalTime(Number days)`. 
GlideDateTime - addDaysUTC(Number days)

Adds a specified number of days to the current GlideDateTime object. A negative parameter subtracts days.

The method determines the UTC date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts days using the UTC date and time values.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>days</td>
<td>Number</td>
<td>The number of days to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addDaysUTC(-1); gs.print(gdt.getDate());
```

Output: 2011-08-30

**Scoped equivalent**

To use the `addDaysUTC()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - addDaysUTC(Number days)**.

**GlideDateTime - addSeconds(Number seconds)**

Adds a specified number of seconds to the GlideDateTime object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>seconds</td>
<td>Number</td>
<td>The number of seconds to add</td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
var gdt = new GlideDateTime("2011-12-07 08:00:00");
gdt.addSeconds(1000);
gs.print(gdt.getValue());
```

Output: 2011-12-07 08:16:40

**Scoped equivalent**

To use the `addSeconds()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - addSeconds(Number seconds)`.

**GlideDateTime - addWeeks(Number weeks)**

Adds a specified number of weeks to the current GlideDateTime object. A negative parameter subtracts weeks.

Use `addWeeksLocalTime()` and `addWeeksUTC()` instead of this method.

```
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addWeeks(-1);
gs.print(gdt.getDate());
```

Output: 2011-08-24
GlideDateTime - addWeeksLocalTime(Number weeks)

Adds a specified number of weeks to the current GlideDateTime object. A negative parameter subtracts weeks.

The method determines the local date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts weeks using the local date and time values.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>weeks</td>
<td>Number</td>
<td>The number of weeks to add. Use a negative number to subtract.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addWeeksLocalTime(-1);
gs.print(gdt.getDate());
```

Output: 2011-08-24

**Scoped equivalent**

To use the `addWeeksLocalTime()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - addWeeksLocalTime(Number weeks)**.

GlideDateTime - addWeeksUTC(Number weeks)

Adds a specified number of weeks to the current GlideDateTime object. A negative parameter subtracts weeks.

The method determines the UTC date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts weeks using the UTC date and time values.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>weeks</td>
<td>Number</td>
<td>The number of weeks to add. Use a negative number to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addWeeksUTC(-1);
gs.print(gdt.getDate());
```

Output: 2011-08-24

Scoped equivalent

To use the `addWeeksUTC()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - addWeeksUTC(Number weeks)**.

GlideDateTime - addMonths(Number months)

Adds a specified number of months to the current GlideDateTime object. A negative parameter subtracts months.

Use `addMonthsLocalTime()` or `addMonthsUTC()` instead of this method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>months</td>
<td>Number</td>
<td>The number of months to add. Use a negative number to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addMonths(2);
gs.print(gdt.getDate());

Output: 2011-10-31

GlideDateTime - addMonthsLocalTime(Number months)

Adds a specified number of months to the current GlideDateTime object. A negative parameter subtracts months.

The method determines the local date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts months using the local date and time values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>months</td>
<td>Number</td>
<td>The number of months to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addMonthsLocalTime(2);
gs.print(gdt.getDate());

Output: 2011-10-31

Scoped equivalent

To use the `addMonthsLocalTime()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - addMonthsLocalTime(Number months)**.

GlideDateTime - addMonthsUTC(Number months)

Adds a specified number of months to the current GlideDateTime object. A negative parameter subtracts months.
The method determines the UTC date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts months using the UTC date and time values.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>months</td>
<td>Number</td>
<td>The number of months to add. Use a negative number to subtract.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addMonthsUTC(2);
gs.print(gdt.getDate());
```

Output: 2011-10-31

**Scoped equivalent**

To use the `addMonthsUTC()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - addMonthsUTC(Number months)**.

**GlideDateTime - addYears(Number years)**

Adds a specified number of years to the current GlideDateTime object. A negative parameter subtracts years.

Use `addYearsLocalTime()` or `addYearsUTC()` instead of this method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>years</td>
<td>Number</td>
<td>The number of years to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2010-08-31 08:00:00");
gdt.addYears(1);
gs.print(gdt.getDate());
```

Output: 2011-08-31

**GlideDateTime - addYearsLocalTime(Number years)**

Adds a specified number of years to the current GlideDateTime object. A negative parameter subtracts years.

The method determines the local date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts years using the local date and time values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>years</td>
<td>Number</td>
<td>The number of years to add. To subtract use a negative value.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2010-08-31 08:00:00");
gdt.addYearsLocalTime(1);
gs.print(gdt.getDate());
```

Output: 2011-08-31

**Scoped equivalent**

To use the `addYearsLocalTime()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - addYearsLocalTime(Number years)`. 

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
GlideDateTime - addYearsUTC(Number years)

Adds a specified number of years to the current GlideDateTime object. A negative parameter subtracts years.

The date and time value stored by GlideDateTime object is interpreted as being in the UTC time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>years</td>
<td>Number</td>
<td>The number of years to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2010-08-31 08:00:00");
gdt.addYearsUTC(1);
gs.print(gdt.getDate());
```

Output: 2011-08-31

Scoped equivalent

To use the addYearsUTC() method in a scoped application, use the corresponding scoped method: Scoped GlideDateTime - addYearsUTC(Number years).

GlideDateTime - compareTo(Object dateTime)

Compares two date and time objects to determine whether one occurs before the other or if they are equivalent.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>Object</td>
<td>Date time in a GlideDateTime object</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>• 0 = Dates are equal</td>
</tr>
<tr>
<td></td>
<td>• 1 = The object's date is after the date specified in the parameter</td>
</tr>
<tr>
<td></td>
<td>• -1 = The object's date is before the date specified in the parameter</td>
</tr>
</tbody>
</table>

```javascript
var initDate = new GlideDateTime("2011-08-01 12:00:00");
var compDate1 = new GlideDateTime("2011-08-01 12:00:00");
var compDate2 = new GlideDateTime("2011-07-31 12:00:00");
var compDate3 = new GlideDateTime("2011-08-04 16:00:00");

gs.info(initDate.compareTo(compDate1)); // Equals (0)
gs.info(initDate.compareTo(compDate2)); // initDate is after compDate2 (1)
gs.info(initDate.compareTo(compDate3)); // initDate is before compDate3 (-1)
```

### Scoped equivalent

To use the `compareTo()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - compareTo(Object o).**

### GlideDateTime - equals(Object GDT)

Compares an object with an existing value for equality.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GDT</td>
<td>Object</td>
<td>The object to compare. Can be a GlideDateTime object or a valid date time string.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if they are equal, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 00:00:00");
gs.print(gdt.equals("2011-09-30 00:12:01"));
```

Output: false
Scoped equivalent

To use the `equals()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - equals(Object dateTime)`.

GlideDateTime - `getDate()`

Gets the date stored by the GlideDateTime object, expressed in the standard format, yyyy-MM-dd, and the system time zone, UTC by default.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDate</td>
<td>The date in the system time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.print(gdt.getDate());
```

Output: 2011-08-31

Scoped equivalent

To use the `getDate()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getDate()`.

GlideDateTime - `getDayOfMonth()`

Gets the current day of the month in the UTC time zone.

Use `getDayOfMonthLocalTime()` and `getDayOfMonthUTC()` instead of this method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Number</td>
<td>The day of the month in the UTC time zone, from 1 to 31.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-02 12:00:00");
gs.print(gdt.getDayOfMonth());
```

Output: 2

### GlideDateTime - getDayOfMonthLocalTime()

Gets the day of the month stored by the GlideDateTime object, expressed in the current user's time zone.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The day of the month in the user's time zone, from 1 to 31.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-02 12:00:00");
gs.print(gdt.getDayOfMonthLocalTime());
```

Output: 2

**Scoped equivalent**

To use the `getDayOfMonthLocalTime()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getDayOfMonthLocalTime()`.

### GlideDateTime - getDayOfMonthUTC()

Gets the day of the month stored by the GlideDateTime object, expressed in the UTC time zone.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The day of the month in the UTC time zone, from 1 to 31.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-02 12:00:00");
gs.print(gdt.getDayOfMonthUTC());
```

Output: 2

**Scoped equivalent**

To use the `getDayOfMonthUTC()` method in a scoped application, use the corresponding scoped method: Scoped GlideDateTime - `getDayOfMonthUTC()`.

**GlideDateTime - getDayOfWeek()**

Retrieves the day of the week stored by the GlideDateTime object, expressed in the user's time zone.

Use `getDayOfWeekLocalTime()` and `getDayOfWeekUTC()` instead of this method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The day of the week value - Monday = 1, ... Sunday = 7.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-01 12:00:00");
gs.print(gdt.getDayOfWeek());
```

Output: 4
**GlideDateTime - getDayOfWeekLocalTime()**

Gets the day of the week stored by the GlideDateTime object, expressed in the user's time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The day of the week value - Monday = 1, ... Sunday = 7</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-01 12:00:00");
gs.print(gdt.getDayOfWeekLocalTime());
```

Output: 4

**Scoped equivalent**

To use the `getDayOfWeekLocalTime()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getDayOfWeekLocalTime()`.

**GlideDateTime - getDayOfWeekUTC()**

Gets the day of the week stored by the GlideDateTime object, expressed in the UTC time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The day of the week value - Monday = 1, ... Sunday = 7</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
var gdt = new GlideDateTime("2011-12-01 12:00:00");
gs.print(gdt.getDayOfWeekUTC());

Output: 4

**Scoped equivalent**

To use the `getDayOfWeekUTC()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getDayOfWeekUTC()`.

**GlideDateTime - getDaysInMonth()**

Gets the number of days in the month stored by the GlideDateTime object, expressed in the Java Virtual Machine time zone.

Use `getDaysInMonthLocalTime()` and `getDaysInMonthUTC()` instead of this method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of days in the current month in the Java Virtual Machine time zone.</td>
</tr>
</tbody>
</table>

```java
var gdt = new GlideDateTime(); //December
gs.print(gdt.getDaysInMonth());
```

Output: 31

**GlideDateTime - getDaysInMonthLocalTime()**

Gets the number of days in the month stored by the GlideDateTime object, expressed in the current user’s time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of days in the current month in the user's time zone.</td>
</tr>
</tbody>
</table>

```
var gdt = new GlideDateTime(); //December
gs.print(gdt.getDaysInMonthLocalTime());
```

Output: 31

Scoped equivalent

To use the `getDaysInMonthLocalTime()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - getDaysInMonthLocalTime().**

GlideDateTime - `getDaysInMonthUTC()`

Gets the number of days in the month stored by the GlideDateTime object, expressed in the UTC time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of days in the month stored by the GlideDateTime object, expressed in the UTC time zone.</td>
</tr>
</tbody>
</table>

```
var gdt = new GlideDateTime(); //December
gs.print(gdt.getDaysInMonthUTC());
```

Output: 31

Scoped equivalent

To use the `getDaysInMonthUTC()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - getDaysInMonthUTC().**
GlideDateTime - getDisplayValue()

Gets the date and time value in the current user's display format and time zone.

⚠️ Note: Referring to the GlideDateTime object directly returns the date and time value in the GMT time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Date and time in the user's format and time zone. Keep in mind when designing business rules or script includes that this method may return values in different formats for different users.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.print(gdt.getDisplayValue());
```

Output:

```
2011-08-31 01:00:00
```

Scoped equivalent

To use the `getDisplayValue()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - getDisplayValue()**.

GlideDateTime - getDisplayValueInternal()

Returns the display value in the internal format (yyyy-MM-dd HH:mm:ss). This method is useful for date/time fields, but not for date fields.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### GlideDateTime - getDisplayValueInternal()

The date and time values for the GlideDateTime object in the current user's time zone and the internal date and time format of yyyy-MM-dd HH:mm:ss.

```javascript
// Wednesday
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.print(gdt.getDisplayValueInternal());
```

**Output:**

```
2011-08-31 01:00:00
```

### Scoped equivalent

To use the `getDisplayValueInternal()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - getDisplayValueInternal().**

### GlideDateTime - getDSTOffset()

Gets the amount of time that daylight saving time is offset.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Amount of time, in milliseconds, that daylight saving is offset. Returns 0 if there is no offset or if the time is not during daylight saving time.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.print(gdt.getDSTOffset());
```

**Output:**

```
3600000
```
Scoped equivalent

To use the `getDSTOffset()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getDSTOffset()`.

GlideDateTime - `getErrorMsg()`

Gets the current error message.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The error message</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-aa-31 aa:00:00"); //bad
gs.print(gdt.isValid()); //false
gs.print(gdt.getErrorMsg());  //reason
```

Output:

false

Could not parse DateTime: 2011-aa-31 aa:00:00

Scoped equivalent

To use the `getErrorMsg()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getErrorMsg()`.

GlideDateTime - `getInternalFormattedLocalTime()`

Returns the object’s time in the local time zone and in the internal format.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The object's time in the local time zone and the internal format.</td>
</tr>
</tbody>
</table>

Scoped equivalent

To use the `getInternalFormattedLocalTime()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getInternalFormattedLocalTime()`.

**GlideDateTime - getInternalMidnight(Number dayOfTheWeek)**

Returns a date and time object set to midnight of a specified day using UTC.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>dayOfTheWeek</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>GlideDateTime</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.print(gdt.getInternalMidnight(2));
```

Output: 2011-08-30 00:00:01

**GlideDateTime - getLocalDate()**

Gets the date stored by the GlideDateTime object, expressed in the standard format, yyyy-MM-dd, and the current user's time zone.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDate</td>
<td>The date in the user's time zone.</td>
</tr>
</tbody>
</table>

```
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.print(gdt.getLocalDate());
```

Output: 2011-08-31

**Scoped equivalent**

To use the `getLocalDate()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getLocalDate()`.

**GlideDateTime - getLocalTime()**

Gets the time in the user's time zone.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

```
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.print(gdt.getLocalTime());
```

Output: 1970-01-01 01:00:00

**Scoped equivalent**

To use the `getLocalTime()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getLocalTime()`.
**GlideDateTime - getMonth()**

Retrieves the month stored by the GlideDateTime object, expressed in Java Virtual Machine time zone.

Use `getMonthLocalTime()` and `getMonthUTC()` instead of this method.

---

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The numerical value of the month, Jan=1, Dec=12.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime(); //December
gs.print(gdt.getMonth());

Output: 12
```

---

**GlideDateTime - getMonthLocalTime()**

Gets the month stored by the GlideDateTime object, expressed in the current user's time zone.

---

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The numerical value of the month, Jan=1, Dec=12.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime(); //December
gs.print(gdt.getMonthLocalTime());

Output: 12
```
Scoped equivalent

To use the `getMonthLocalTime()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getMonthLocalTime()`.

**GlideDateTime - getMonthUTC()**

Gets the month stored by the GlideDateTime object, expressed in the UTC time zone.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Number</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime(); //December
gs.print(gdt.getMonthUTC());
```

Output: 12

Scoped equivalent

To use the `getMonthUTC()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getMonthUTC()`.

**GlideDateTime - getNumericValue()**

 Gets the number of milliseconds since January 1, 1970, 00:00:00 GMT.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of milliseconds since January 1, 1970, 00:00:00 GMT.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.print(gdt.getNumericValue());
```

Output: 1314777600000

**GlideDateTime - getSpanTime(Number dayOfWeek)**

Retrieves the amount of time elapsed since the midnight of a specified day to the current time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dayOfWeek</td>
<td>Number</td>
<td>Day of week value from 1 to 7. 1 = Monday, 7 = Sunday.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideTime</td>
<td>The amount of time elapsed since midnight of the specified day. To display the result in user-friendly terms, set the value to GlideDuration.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00"); //Wednesday
var dur = new GlideDuration();
var span = gdt.getSpanTime(1); //how much time since Monday 00:00:00
dur.setValue(span);
gs.print(dur.getDisplayValue());
```

Output: 2 Days 8 Hours

**GlideDateTime - getTime()**

Gets the Unix duration stamp.
### GlideTime

The Unix duration stamp in system format based on GMT time.

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.print(gdt.getTime());
```

Output: 1970-01-01 08:00:00

#### Scoped equivalent

To use the `getTime()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getTime()`.

### GlideDateTime - getTZOffset()

Gets the time zone offset in milliseconds.

```javascript
var gdt = new GlideDateTime();
gdt.getLocalTime(); // PST local time
gs.print(gdt.getTZOffset());
```

Output: -28800000
Scoped equivalent
To use the `getTZOffset()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getTZOffset()`.

GlideDateTime - `getUseUserFormattedLocalTime()`
Returns the object's time in local time zone in the user's format.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

Reached by using the `getUserFormattedLocalTime()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getUserFormattedLocalTime()`.

GlideDateTime - `getUserTimeZone()`
Retrieves the time zone for the current user session. This method is equivalent to `gs.getSession().getTimeZone()`.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

Scoped equivalent
To use the `getUserFormattedLocalTime()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getUserFormattedLocalTime()`.

GlideDateTime - `getUserTimeZone()`
Retrieves the time zone for the current user session. This method is equivalent to `gs.getSession().getTimeZone()`.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>TimeZone</td>
</tr>
</tbody>
</table>
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.print(gdt.getUserTimeZone());

Output:

sun.util.calendar.ZoneInfo[id=\"America/Los_Angeles\",offset=-28800000,\ndstSavings=3600000,useDaylight=true,transitions=185,
lastRule=java.util.SimpleTimeZone[id=\"America/Los_Angeles\",offset=-28800000,\ndstSavings=3600000,\n0,useDaylight=true,startYear=0,startMode=3,
startMonth=2,startDay=8,startDayOfWeek=1,startTime=7200000,startTimeMode=0,endMode=3,endMonth=10,endDay=1,endDayOfWeek=1,
endTime=7200000,endTimeMode=0]]

**GlideDateTime - getUTCMidnight(Number dayOfTheWeek)**

Retrieves a GlideDateTime object with the time set to midnight using the UTC time zone.

This method sets the date of the new GlideDateTime object as the specified day of the week within the week of the original GlideDateTime object.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>dayOfTheWeek</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>GlideDateTime</td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime("2011-08-31 08:00:00"); // Wednesday, 3rd day of the week. gs.print(gdt.getUTCMidnight(5)); //Friday, 5th day of the week.

Output: 2011-09-02 00:00:00

**GlideDateTime - getValue()**

Gets the date and time value stored by the GlideDateTime object in the internal format, yyyy-MM-dd HH:mm:ss, and the system time zone, UTC by default.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The date and time in the internal format and system time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.print(gdt.getValue());
```

Output:

2011-08-31 08:00:00

Scoped equivalent

To use the `getValue()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - getValue()`.

GlideDateTime - `getWeekOfYearLocalTime()`

Gets the number of the week stored by the GlideDateTime object, expressed in the current user's time zone.

All weeks begin on Sunday. The first week of the year is the week that contains at least one day of the new year. The week beginning Sunday 2015-12-27 is considered the first week of 2016 as that week contains January 1 and 2.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of the current week. The highest week number in a year is either 52 or 53.</td>
</tr>
</tbody>
</table>
var gdt = new GlideDateTime("2011-12-01 12:00:00");
gs.print(gdt.getWeekOfYearUTC());

Output: 49

Scoped equivalent
To use the getWeekOfYearLocalTime() method in a scoped application, use the corresponding scoped method: Scoped GlideDateTime - getWeekOfYearLocalTime().

GlideDateTime - getWeekOfYearUTC()

Gets the number of the week stored by the GlideDateTime object, expressed in the UTC time zone.

All weeks begin on Sunday. The first week of the year is the week that contains at least one day of the new year. The week beginning Sunday 2015-12-27 is considered the first week of 2016 as that week contains January 1 and 2.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of the current week in UTC time. The highest week number in a year is either 52 or 53.</td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime("2011-12-01 12:00:00");
gs.print(gdt.getWeekOfYearUTC());

Output: 49

Scoped equivalent
To use the getWeekOfYearUTC() method in a scoped application, use the corresponding scoped method: Scoped GlideDateTime - getWeekOfYearUTC().
GlideDateTime - `getYear()`

Retrieves the year stored by the GlideDateTime object, expressed in the Java Virtual Machine time zone.

Use `getYearLocalTime()` and `getYearUTC()` instead of this method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The 4-digit year value in the Java Virtual Machine time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime(); //2011
gs.print(gdt.getYear());
```

Output: 2011

GlideDateTime - `getYearLocalTime()`

Gets the year stored by the GlideDateTime object, expressed in the current user's time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The 4-digit year value in the user's time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime(); //2011
gs.print(gdt.getYearLocalTime());
```

Output: 2011
Scoped equivalent

To use the `getYearLocalTime()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - getYearLocalTime()**.

**GlideDateTime - getYearUTC()**

Gets the year stored by the GlideDateTime object, expressed in the UTC time zone.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>number</td>
</tr>
</tbody>
</table>

```
var gdt = new GlideDateTime(); //2011
gs.print(gdt.getYearUTC());
```

Output: 2011

Scoped equivalent

To use the `getYearUTC()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - getYearUTC()**.

**GlideDateTime - GlideDateTime()**

Instantiates a new GlideDateTime object with the current date and time in GMT format.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

```
var gdt = new GlideDateTime();
```
**GlideDateTime - GlideDateTime(String dateTime)**

Instantiates a new GlideDateTime object from a date and time value in the UTC time zone specified with the format yyyy-MM-dd HH:mm:ss.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>String</td>
<td>A UTC date and time using the format yyyy-MM-dd HH:mm:ss.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-01-01 12:00:00");
```

**GlideDateTime - GlideDateTime(GlideDateTime gdt)**

Instantiates a new GlideDateTime object set to the time of a specified GlideDateTime object in GMT format.

The system attempts to match the argument passed to the gdt parameter with the specified internal system format. If the argument does not match the system format, the system attempts to match it to one of the following formats in this order:

- yyyy-MM-dd'T'HH:mm:ss.Z
- MM/dd/yyyy HH:mm:ss
- MM-dd-yyyy HH:mm:ss
- MM-dd-yyyy HH:mm
- MM-dd-yyyy
- MM/dd/yyyy
- dd-MM-yyyy HH:mm:ss
- dd-MM-yyyy HH.mm:ss
- dd-MM-yyyy HH:mm
- dd-MM-yyyy
- yyyy-MM-dd HH:mm
• yyyy-MM-dd
• dd.MM.yyyy HH:mm:ss
• dd.MM.yyyy HH.mm.ss
• dd.MM.yyyy hh:mm:ss a
• dd.MM.yyyy hh.mm.ss a
• dd.MM.yyyy

If one of these formats results in a valid date/time, GlideDateTime.isValid() returns true.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gdt</td>
<td>GlideDateTime</td>
<td>Object used to set the time of the new object.</td>
</tr>
</tbody>
</table>

This example instantiates a GlideDateTime object using the internal system format.

```javascript
var start = new GlideDateTime("2011-01-01 12:00:00");
var end = new GlideDateTime(start);
gs.print(end);
```

Output:

```
2011-01-01 12:00:00
```

This example instantiates a GlideDateTime object using the `yyyy-MM-dd'T'HH:mm:ss.SSSZ` format.

```javascript
var start = new GlideDateTime("2014-07-04T12:08:56.235-0700");
var end = new GlideDateTime(start);
gs.print(end);
```

Output:

```
2014-07-04 19:08:56
```

**GlideDateTime - hasDate()**

Determines if an object's date is set.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the object's date is set, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.print(gdt.hasDate());
```

**Output:**

```
true
```

### Scoped equivalent

To use the `hasDate()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - hasDate()**.

### GlideDateTime - isDST()

Determines if the object's time uses a daylight saving offset

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the time is daylight saving time, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 00:00:00");
gs.print(gdt.isDST()); //true
```

**Output:**
Scoped equivalent
To use the `isDST()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - isDST()`.

**GlideDateTime - isValid()**
Determines if a value is a valid date and time.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if value is valid, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-aa-31 aa:00:00"); //bad
gs.print(gdt.isValid()); //false
gs.print(gdt.getErrorMsg()); //reason
```

Output: false

Scoped equivalent
To use the `isValid()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - isValid()`.

**GlideDateTime - setDayOfMonth(Number day)**
Sets the day of the month to a specified value.

Use `setDayOfMonthLocalTime(day)` and `setDayOfMonthUTC(day)` instead of this method.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>day</td>
<td>Number</td>
<td>Day of the month, from 1 to 31.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime();
gdt.setDayOfMonth(9);
gs.print(gdt.getDayOfMonth());
```

Output: 9

**GlideDateTime - setDayOfMonthLocalTime(Number day)**

Sets the day of the month to a specified value in the current user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>day</td>
<td>Number</td>
<td>The day of month to change to, from 1 to 31. If this value is greater than the maximum number of days in the month, the value is set to the last day of the month.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime();
gdt.setDayOfMonthLocalTime(9);
gs.print(gdt.getDayOfMonthLocalTime());
```

Output: 9
Scoped equivalent

To use the `setDayOfMonthLocalTime()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - setDayOfMonthLocalTime(Number day)`.

**GlideDateTime - setDayOfMonthUTC(Number day)**

Sets the day of the month to a specified value in the UTC time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>day</td>
<td>Number</td>
<td>The day of month to change to, from 1 to 31. If this value is greater than</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the maximum number of days in the month, the value is set to the last day</td>
</tr>
<tr>
<td></td>
<td></td>
<td>of the month.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime();
gdt.setDayOfMonthUTC(9);
gs.print(gdt.getDayOfMonthUTC());
```

Output: 9

Scoped equivalent

To use the `setDayOfMonthUTC()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - setDayOfMonthUTC(Number day)`.

**GlideDateTime - setDisplayValue(String asDisplayed)**

Sets a date and time value using the current user's display format and time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>asDisplayed</td>
<td>String</td>
<td>The date and time in the current user's display format and time zone. The</td>
</tr>
<tr>
<td></td>
<td></td>
<td>parameter must be formatted</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>using the current user’s preferred display format, such as MM-dd-yyyy HH:mm:ss. To assign the current date and time to a variable in a workflow script, use <code>&lt;variable&gt;.setDisplayValue(gs.nowDateTime);</code>.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-02-02 12:00:00");
gdt.setDisplayValue("2011-01-01 12:00:00");
gs.print(gdt.getValue());
```

Output:

```
2011-01-01 20:00:00
```

Scoped equivalent

To use the `setDisplayValue()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - setDisplayValue(String asDisplayed)**.

**GlideDateTime - setDisplayValue(String dateTime, String format)**

Sets a date and time value using the current user's time zone and the specified date and time format.

This method throws a runtime exception if the date and time format used in the `dateTime` parameter does not match the `format` parameter. You can retrieve the error message by calling `getErrorMsg()` on the GlideDateTime object after the exception is caught.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>String</td>
<td>The date and time in the current user’s time zone.</td>
</tr>
<tr>
<td>format</td>
<td>String</td>
<td>The format to use to parse the dateTime parameter.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-02-02 12:00:00");
gdt.setDisplayValue("20-5-2011 12:00:00", "dd-MM-yyyy HH:mm:ss");
gs.print(gdt.getValue());
```

Output:

```
2011-05-20 19:00:00
```

Scoped equivalent

To use the `setDisplayValue()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - setDisplayValue(String value, String format)`.

**GlideDateTime - setDisplayValueInternal(String dateTime)**

Sets a date and time value using the internal format (yyyy-MM-dd HH:mm:ss) and the current user's time zone.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>String</td>
<td>The date and time in internal format</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-02-02 12:00:00");
gdt.setDisplayValueInternal("2011-01-01 12:00:00");
gs.print(gdt.getValue());
```

Output:

```
2011-01-01 20:00:00
```
Scoped equivalent

To use the `setDisplayValueInternal()` method in a scoped application, use the corresponding scoped method: 

```
Scoped GlideDateTime - setDisplayValueInternal(String value).
```

**GlideDateTime - setDisplayValueInternalWithAlternates(String dateTime)**

Sets a date and time value using the internal format (yyyy-MM-dd HH:mm:ss) and the current user's time zone.

This method attempts to parse incomplete date and time values.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>String</td>
<td>The date and time in internal format.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideDateTime - setGlideDateTime(GlideDateTime gDT)**

Sets the date and time of the current object using an existing GlideDateTime object.

This method is equivalent to instantiating a new object with a GlideDateTime parameter.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gDT</td>
<td>GlideDateTime</td>
<td>A GlideDateTime object</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var dt1 = new GlideDateTime("2011-01-01 12:00:00");
var dt2 = new GlideDateTime("2011-02-02 08:00:00");
dt1.setGlideDateTime(dt2);
gs.print(dt1.getValue());

Output:
2011-02-02 08:00:00

Scoped equivalent
To use the setGlideDateTime() method in a scoped application, use the corresponding scoped method: Scoped GlideDateTime - setGlideDateTime(GlideDateTime g).

GlideDateTime - setInitialValue(String dateTime)
Sets the date and time.
This method is equivalent to setValue(Object).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>String</td>
<td>The date and time to use. Accepts either a string in the GMT time zone in the internal format, or a GlideDateTime object.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime();
gdt.setInitialValue("2011-01-01 12:00:00");
gs.print(gdt.getValue());

Output: 2011-01-01 12:00:00

GlideDateTime - setMonth(Number month)
Sets the month stored by the GlideDateTime object to a specified value using the Java Virtual Machine time zone.

Use setMonthLocalTime() or setMonthUTC() instead of this method.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>month</td>
<td>Number</td>
<td>The month to change to.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime();
gdt.setMonth(1);
gs.print(gdt.getMonth());
```

Output: 1

**GlideDateTime - setMonthLocalTime(Number month)**

Sets the month stored by the GlideDateTime object to a specified value using the current user's time zone.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>month</td>
<td>Number</td>
<td>The month to change to.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime();
gdt.setMonthLocalTime(1);
gs.print(gdt.getMonthLocalTime());
```

Output: 1
Scoped equivalent
To use the `setMonthLocalTime()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - setMonthLocalTime(Number month)`. 

GlideDateTime - `setMonthUTC(Number month)`
Sets the month stored by the GlideDateTime object to a specified value using the UTC time zone.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>month</td>
<td>Number</td>
<td>The month to change to.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime();
gdt.setMonthUTC(1);
gs.print(gdt.getMonthUTC());
```

Output: 1

Scoped equivalent
To use the `setMonthUTC()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - setMonthUTC(Number month)`. 

GlideDateTime - `setNumericValue(Number milliseconds)`
Sets the date and time to the number of milliseconds since January 1, 1970 00:00:00 GMT.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>milliseconds</td>
<td>Number</td>
<td>Number of milliseconds</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime();
gdt.setNumericValue(1314777600000);
gs.print(gdt.getValue());
```

Output: 2011-08-31 08:00:00

**GlideDateTime - setTZ(TimeZone timeZone)**

Sets the time zone of the GlideDateTime object to be the specified time zone.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeZone</td>
<td>TimeZone</td>
<td>A time zone object</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var tz = gs.getSession().getTimeZone();
var gdt = new GlideDateTime();
gdt.setTZ(tz);
```

**GlideDateTime - setValue(Object dateTime)**

Sets the date and time of the GlideDateTime object.

This method is equivalent to `setInitialValue()`.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>Object</td>
<td>The date and time to use. This parameter may be one of several types.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>- A string in the UTC time zone and the internal format of yyyy-MM-dd HH:mm:ss: Sets the value of the object to the specified date and time. Using the method this way is equivalent to instantiating a new GlideDateTime object using the GlideDateTime(value) constructor. If the date and time format used does not match the internal format, the method attempts to set the date and time using other available formats. Resolving the date and time this way can lead to inaccurate data due to ambiguity in the day and month values. When using a non-standard date and time format, use setValueUTC(dateTime, format) instead.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- A GlideDateTime object: Sets the value of the object to the date and time stored by the GlideDateTime passed in the parameter. Using the method this way is equivalent to instantiating a new GlideDateTime object using the GlideDateTime(g) Constructor.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- A Java Date object: Sets the value of the object using the value stored by the Java Date object. Using the method this way is equivalent to passing the value returned by the Java Date object getTime() to the GlideDateTime setNumericValue() method. This method does not accept JavaScript Date objects.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- A JavaScript Number: Sets the value of the object using the Number value as milliseconds past January 1, 1970 00:00:00 GMT. Using the method this way is equivalent to the setNumericValue(milliseconds) method.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- A Java Integer or Long: Sets the value of the object using the Integer or Long value as milliseconds past January 1, 1970 00:00:00 GMT. Using the method this way is equivalent to the setNumericValue(milliseconds) method.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
Set the date and time using a string in yyyy-MM-dd HH:mm:ss format.

```javascript
var gdt = new GlideDateTime("2011-01-01 12:00:00");
gdt.setValue("2011-02-02 08:00:00");
gs.print(gdt.getValue());
```

Output:

```
2011-02-02 08:00:00
```

Set the date and time using an existing GlideDateTime object.

```javascript
var gdtSource = new GlideDateTime("2017-05-23 11:37:41");
var gdtDest = new GlideDateTime();
gdtDest.setValue(gdtSource);
gs.print(gdtDest.getValue());
```

Output:

```
2017-05-23 11:37:41
```

Set the date and time using a number indicating milliseconds past January 1, 1970 00:00:00 GMT.

```javascript
var gdt = new GlideDateTime("2011-01-01 12:00:00");
gdt.setValue(1617634522375);
gs.print(gdt.getValue());
```

Output:

```
2021-04-05 14:55:22
```

**Scoped equivalent**

To use the `setValue()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - setValue(String o).**

**GlideDateTime - setValueUTC(String dateTime, String format)**

Sets a date and time value using the UTC time zone and the specified date and time format.

This method throws a runtime exception if the date and time format used in the `dateTime` parameter does not match the format parameter. You can retrieve the error message by calling `getErrorMsg()` on the GlideDateTime object after the exception is caught.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>String</td>
<td>The date and time to use.</td>
</tr>
<tr>
<td>format</td>
<td>String</td>
<td>The format to use.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-01-01 12:00:00");
gdt.setValueUTC("15-02-2011 08:00:00", "dd-MM-yyyy HH:mm:ss");
gs.print(gdt.getValue());
```

Output:

2011-02-15 08:00:00

Scoped equivalent

To use the `setValueUTC()` method in a scoped application, use the corresponding scoped method: **Scoped GlideDateTime - setValueUTC(String dt, String format)**.

GlideDateTime - setYear(Number year)

Sets the year stored by the GlideDateTime object to a specified value using the Java Virtual Machine time zone.

Use `setYearLocalTime()` or `setYearUTC()` instead of this method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>year</td>
<td>Number</td>
<td>The year to change to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var gdt = new GlideDateTime();
gdt.setYear(2013);
gs.print(gdt.getYear());

Output: 2013

GlideDateTime - setYearLocalTime(Number year)
Sets the year stored by the GlideDateTime object to a specified value using the current user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>year</td>
<td>Number</td>
<td>The year to change to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime();
gdt.setYearLocalTime(2013);
gs.print(gdt.getYearLocalTime());

Output: 2013

Scoped equivalent
To use the setYearLocalTime() method in a scoped application, use the corresponding scoped method: Scoped GlideDateTime - setYearLocalTime(Number year).

GlideDateTime - setYearUTC(Number year)
Sets the year stored by the GlideDateTime object to a specified value using the UTC time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>year</td>
<td>Number</td>
<td>The year to change to.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime();
gdt.setYearUTC(2013);
gs.print(gdt.getYearUTC());
```

Output: 2013

**Scoped equivalent**

To use the `setYearUTC()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - setYearUTC(Number year)`.

**GlideDateTime - subtract(GlideTime time)**

Subtracts a specified amount of time.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>GlideTime</td>
<td>The time to subtract</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
var gtime1 = new GlideTime();
gtime1.setValue("00:00:20");
gdt.subtract(gtime1);
gs.print(gdt.getTime());
```

Output: 1970-01-01 07:59:40

**Scoped equivalent**

To use the `subtract()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - subtract(GlideTime time)`. 
**GlideDateTime - subtract(Number milliseconds)**

Subtracts a specified number of milliseconds from the GlideDateTime object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>milliseconds</td>
<td>Number</td>
<td>The number of milliseconds to subtract</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-07 08:00:00");
gdt.subtract(1000);
gs.print(gdt.getValue());
```

Output: 2011-12-07 07:59:59

**Scoped equivalent**

To use the `subtract()` method in a scoped application, use the corresponding scoped method: `Scoped GlideDateTime - subtract(Number milliseconds)`.

**GlideDateTime - subtract(GlideDateTime start, GlideDateTime end)**

Gets the duration difference between two GlideDateTime values.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>GlideDateTime</td>
<td>The start value</td>
</tr>
<tr>
<td>end</td>
<td>GlideDateTime</td>
<td>The end value</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDuration</td>
<td>The time between the two values</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Output: 2 Days 23 Hours

Scoped equivalent
To use the `subtract()` method in a scoped application, use the corresponding scoped method: Scoped GlideDateTime - `subtract(GlideDateTime start, GlideDateTime end)`.

GlideDateTime - toString()
Returns the date and time value stored by the GlideDateTime object in the internal format, `yyyy-MM-dd HH:mm:ss`, and the system time zone, UTC by default.

This method is equivalent to `getValue()`.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The date and time stored by the GlideDateTime object in the system time zone and format.</td>
</tr>
</tbody>
</table>

Output: 2011-08-31 08:00:00

Scoped equivalent
To use the `toString()` method in a scoped application, use the corresponding scoped method: Scoped GlideDateTime - `toString()`.
GlideDateTime - Scoped

The scoped GlideDateTime class provides methods for performing operations on GlideDateTime objects, such as instantiating GlideDateTime objects or working with glide_date_time fields.

Use the GlideDateTime methods to perform date-time operations, such as instantiating a GlideDateTime object, performing date-time calculations, formatting a date-time, or converting between date-time formats.

Scoped GlideDateTime - add(GlideTime gd)

Adds a GlideTime object to the current GlideDateTime object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gd</td>
<td>GlideTime</td>
<td>GlideTime object whose time value to add to the specified GlideDateTime object.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
var gtime1 = new GlideTime();
gtime1.setValue("00:00:20");
gdt.add(gtime1);
var gtime2 = gdt.getTime();
gs.info(gtime2.getByFormat('hh:mm:ss'));
```

**Output:**

08:00:20

Scoped GlideDateTime - add(Number milliseconds)

Adds the specified number of milliseconds to the current GlideDateTime object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>milliseconds</td>
<td>Number</td>
<td>The number of milliseconds to add.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.info(gdt.getNumericValue());
gdt.add(10);
gs.info(gdt.getNumericValue());
```

Output:

1314777600000 1314777600010

**Scoped GlideDateTime - addDaysLocalTime(Number days)**

Adds a specified number of days to the current GlideDateTime object. A negative parameter subtracts days. The method determines the local date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts days using the local date and time values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>days</td>
<td>Number</td>
<td>The number of days to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addDaysLocalTime(-1);
gs.info(gdt.getLocalDate());
gs.info(gdt.getNumericValue());
```

Output:

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Scoped GlideDateTime - addDaysUTC(Number days)

Adds a specified number of days to the current GlideDateTime object. A negative parameter subtracts days. The method determines the UTC date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts days using the UTC date and time values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>days</td>
<td>Number</td>
<td>The number of days to add. Use a negative number to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addDaysUTC(-1);
gs.info(gdt.getDate());
```

Output:

2011-08-30

Scoped GlideDateTime - addMonthsLocalTime(Number months)

Adds a specified number of months to the current GlideDateTime object. A negative parameter subtracts months. The method determines the local date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts months using the local date and time values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>months</td>
<td>Number</td>
<td>The number of months to add. use a negative value to subtract.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addMonthsLocalTime(2);
gs.info(gdt.getDate());
```

Output: 2011-10-31

**Scoped GlideDateTime - addMonthsUTC(Number months)**

Adds a specified number of months to the current GlideDateTime object. A negative parameter subtracts months. The method determines the UTC date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts months using the UTC date and time values.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>months</td>
<td>Number</td>
<td>The number of months to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addMonthsUTC(2);
gs.info(gdt.getDate());
```

**Scoped GlideDateTime - addSeconds(Number seconds)**

Adds the specified number of seconds to the current GlideDateTime object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>seconds</td>
<td>Number</td>
<td>The number of seconds to add.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-07 08:00:00");
gdt.addSeconds(1000);
gs.info(gdt.getValue());
```

Output: 2011-12-07 08:16:40

**Scoped GlideDateTime - addWeeksLocalTime(Number weeks)**

Adds a specified number of weeks to the current GlideDateTime object. A negative parameter subtracts weeks. The method determines the local date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts weeks using the local date and time values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>weeks</td>
<td>Number</td>
<td>The number of weeks to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addWeeksLocalTime(-1);
gs.info(gdt.getDate());
```

Output: 2011-08-24
Scoped GlideDateTime - addWeeksUTC(Number weeks)

Adds a specified number of weeks to the current GlideDateTime object. A negative parameter subtracts weeks. The method determines the UTC date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts weeks using the UTC date and time values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>weeks</td>
<td>Number</td>
<td>The number of weeks to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime("2011-08-31 08:00:00");
gdt.addWeeksUTC(-1);
gdt.addWeeksUTC(-1);
gs.info(gdt.getDate());

Output: 2011-08-24

Scoped GlideDateTime - addYearsLocalTime(Number years)

Adds a specified number of years to the current GlideDateTime object. A negative parameter subtracts years. The method determines the local date and time equivalent to the value stored by the GlideDateTime object, then adds or subtracts years using the local date and time values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>years</td>
<td>Number</td>
<td>The number of years to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
```javascript
var gdt = new GlideDateTime("2010-08-31 08:00:00");
gdt.addYearsLocalTime(1);
gs.info(gdt.getDate());
```

Output: 2011-08-31

**Scoped GlideDateTime - addYearsUTC(Number years)**

Adds a specified number of years to the current GlideDateTime object. A negative parameter subtracts years. The date and time value stored by GlideDateTime object is interpreted as being in the UTC time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>years</td>
<td>Number</td>
<td>The number of years to add. Use a negative value to subtract.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2010-08-31 08:00:00");
gdt.addYearsUTC(1);
gs.info(gdt.getDate());
```

Output: 2011-08-31

**Scoped GlideDateTime - after(GlideDateTime gdt)**

Determines if the GlideDateTime object's date and time occurs after the specified object's date and time.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gdt</td>
<td>GlideDateTime</td>
<td>Date and time to check against.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean| Flag that indicates whether the GlideDateTime object's date and time is after the date and time specified by the parameter. Possible values:  
  • true: GlideDateTime object date and time is after the specified object's date and time.  
  • false: GlideDateTime object date and time is before or equal to the specified object's date and time. |

var gdt1 = new GlideDateTime("2016-05-09 10:11:12");
var gdt2 = new GlideDateTime("2017-06-12 15:11:12");
gs.info(gdt1.after(gdt2));

Output: False

Scoped GlideDateTime - before(GlideDateTime gdt)

Determines if the GlideDateTime object's date and time occurs before the specified GlideDateTime object's date and time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gdt</td>
<td>GlideDateTime</td>
<td>Date and time to check against.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean| Flag that indicates whether the GlideDateTime object's date and time is before the date and time specified by the parameter. Possible values:  
  • true: GlideDateTime object date and time is before the specified date and time.  
  • false: GlideDateTime object date and time is after or equal to the specified date and time. |
var gdt1 = new GlideDateTime("2016-05-09 10:11:12");
var gdt2 = new GlideDateTime("2017-06-12 15:11:12");
gs.info(gdt1.before(gdt2));

Output
true

Scoped GlideDateTime - compareTo(Object o)

Compares two date and time objects to determine whether they are equivalent or one occurs before or after the other.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>o</td>
<td>Object</td>
<td>Date and time object in GlideDateTime format</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Number | 0 = Dates are equal  
|      | 1 = The object's date is after the date specified in the parameter  
|      | -1 = The object's date is before the date specified in the parameter  |

var initDate = new GlideDateTime("2011-08-01 12:00:00");
var compDate1 = new GlideDateTime("2011-08-01 12:00:00");
var compDate2 = new GlideDateTime("2011-07-31 12:00:00");
var compDate3 = new GlideDateTime("2011-08-04 16:00:00");
gs.info(initDate.compareTo(compDate1)); // Equals (0)
gs.info(initDate.compareTo(compDate2)); // initDate is after compDate2 (1)
gs.info(initDate.compareTo(compDate3)); // initDate is before compDate3 (-1)

Scoped GlideDateTime - equals(Object dateTime)

Compares a datetime with an existing value for equality.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTime</td>
<td>GlideDateTime object or String</td>
<td>The datetime to compare.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if they are equal; otherwise, false.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 00:00:00");
gs.info(gdt.equals("2011-09-30 00:12:01"));
```

Output: false

**Scoped GlideDateTime - getDate()**

Gets the date stored by the GlideDateTime object, expressed in the standard format, yyyy-MM-dd, and the system time zone, UTC by default.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDate</td>
<td>The date in the system time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.info(gdt.getDate());
```

**Scoped GlideDateTime - getDayOfMonthLocalTime()**

Gets the day of the month stored by the GlideDateTime object, expressed in the current user's time zone.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The day of the month in the user's time zone, from 1 to 31.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-02 12:00:00");
gs.info(gdt.getDayOfMonthLocalTime());
```

### Scoped GlideDateTime - getDayOfMonthUTC()

Gets the day of the month stored by the GlideDateTime object, expressed in the UTC time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The day of the month in the UTC time zone, from 1 to 31.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-02 12:00:00");
gs.info(gdt.getDayOfMonthUTC());
```

Output: 02

### Scoped GlideDateTime - getDayOfWeekLocalTime()

Gets the day of the week stored by the GlideDateTime object, expressed in the user's time zone.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The day of week value, in the user's time zone, from 1 to 7. Monday equals 1, Sunday equals 7.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-01 12:00:00"); //Thursday
gs.info(gdt.getDayOfWeekLocalTime());
```

### Scoped GlideDateTime - getDayOfWeekUTC()

Gets the day of the week stored by the GlideDateTime object, expressed in the UTC time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The day of week value from 1 to 7. Monday equals 1, Sunday equals 7.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-01 12:00:00"); //Thursday
gs.info(gdt.getDayOfWeekLocalTime());
```

### Scoped GlideDateTime - getDaysInMonthLocalTime()

Gets the number of days in the month stored by the GlideDateTime object, expressed in the current user's time zone.

```javascript
var gdt = new GlideDateTime("2011-12-01 12:00:00"); //Thursday
gs.info(gdt.getDayOfWeekLocalTime());
```
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of days in the current month in the user's time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-02 12:00:00"); //December
gs.info(gdt.getDaysInMonthLocalTime());
```

**Scoped GlideDateTime - getDaysInMonthUTC()**

Gets the number of days in the month stored by the GlideDateTime object, expressed in the UTC time zone.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of days in the month stored by the GlideDateTime object, expressed in the UTC time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-11-02 12:00:00"); //November
gs.info(gdt.getDaysInMonthUTC());
```

**Scoped GlideDateTime - getDisplayValue()**

Gets the date and time value in the current user's display format and time zone.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The date and time in the user's format and time zone. Keep in mind when designing business rules or script includes that this method may return values in different formats for different users.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.info(gdt.getDisplayValue()); //uses current user session time zone (US/Pacific)
```

**Scoped GlideDateTime - getDisplayValueInternal()**

Gets the display value in the internal format (yyyy-MM-dd HH:mm:ss).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The date and time values for the GlideDateTime object in the current user's time zone and the internal date and time format of yyyy-MM-dd HH:mm:ss.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.info(gdt.getDisplayValueInternal()); //uses current user session time zone (US/Pacific)
```

**Scoped GlideDateTime - getDSTOffset()**

Gets the amount of time that daylight saving time is offset.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Amount of time, in milliseconds, that daylight saving is offset. Returns 0 if there is no offset or if the time is not during daylight saving time.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2014-08-31 08:00:00");
gs.info(gdt.getDSTOffset()); //uses current user session time zone (US/Pacific)
```

Scoped GlideDateTime - getErrorMsg()

Gets the current error message.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The error message.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime();
gdt.setDisplayValue("2011-aa-01 00:00:00");
gs.info(gdt.getErrorMsg());
```

Output: Could not parse DateTime: 2011-aa-01 00:00:00

Scoped GlideDateTime - getInternalFormattedLocalTime()

Returns the object's time in the local time zone and in the internal format.

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The object's time in the local time zone and the internal format.</td>
</tr>
</tbody>
</table>

Scoped GlideDateTime - getLocalDate()

Gets the date stored by the GlideDateTime object, expressed in the standard format, yyyy-MM-dd, and the current user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDate</td>
<td>The date in the user's time zone.</td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.info(gdt.getLocalDate());

Scoped GlideDateTime - getLocalTime()

Returns a GlideTime object that represents the time portion of the GlideDateTime object in the user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideTime</td>
<td>The time in the user's time zone.</td>
</tr>
</tbody>
</table>

```
var gdt = new GlideDateTime("2014-08-31 08:00:00");
gt = gdt.getLocalTime();
gs.info("local time is " + gt.getByFormat('hh:mm:ss'));
```

Scoped GlideDateTime - getMonthLocalTime()

Gets the month stored by the GlideDateTime object, expressed in the current user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The numerical value of the month.</td>
</tr>
</tbody>
</table>

```
var gdt = new GlideDateTime("2011-11-02 12:00:00"); //November
gs.info(gdt.getMonthLocalTime());
```

Scoped GlideDateTime - getMonthUTC()

Gets the month stored by the GlideDateTime object, expressed in the UTC time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The numerical value of the month.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-11-02 12:00:00"); //November
gs.info(gdt.getMonthUTC());
```

**Scoped GlideDateTime - getNumericValue()**

Gets the number of milliseconds since January 1, 1970, 00:00:00 GMT.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of milliseconds since January 1, 1970, 00:00:00 GMT.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.info(gdt.getNumericValue());
```

**Scoped GlideDateTime - getTime()**

Returns a GlideTime object that represents the time portion of the GlideDateTime object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideTime</td>
<td>The Unix duration stamp in system format based on GMT time.</td>
</tr>
</tbody>
</table>
var gdt = new GlideDateTime("2014-08-31 08:00:00");
gt = gdt.getTime();
gs.info(gt.getByFormat('hh:mm:ss'));

Scoped GlideDateTime - getTZOffset()
Gets the time zone offset in milliseconds.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of milliseconds of time zone offset.</td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime();
gdt.getLocalTime(); // PST local time
gs.info(gdt.getTZOffset());

Output: -25200000

Scoped GlideDateTime - getUserFormattedLocalTime()
Returns the object's time in the local time zone and in the user's format.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The object's time in the local time zone and in the user's format.</td>
</tr>
</tbody>
</table>
Scoped GlideDateTime - getValue()

Gets the date and time value stored by the GlideDateTime object in the internal format, yyyy-MM-dd HH:mm:ss, and the system time zone, UTC by default.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String - The date and time value in the internal format and system time zone.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2014-08-31 08:00:00");
gs.info(gdt.getValue());
```

Scoped GlideDateTime - getWeekOfYearLocalTime()

Gets the number of the week stored by the GlideDateTime object, expressed in the current user's time zone. All weeks begin on Sunday. The first week of the year is the week that contains at least one day of the new year. The week beginning Sunday 2015-12-27 is considered the first week of 2016 as that week contains January 1 and 2.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number - The number of the current week in local time. The highest week number in a year is either 52 or 53.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-01 12:00:00"); // 49th week, 1st week in december
gs.info(gdt.getWeekOfYearLocalTime());
```
Scoped GlideDateTime - getWeekOfYearUTC()

Gets the number of the week stored by the GlideDateTime object, expressed in the UTC time zone. All weeks begin on Sunday. The first week of the year is the week that contains at least one day of the new year. The week beginning Sunday 2015-12-27 is considered the first week of 2016 as that week contains January 1 and 2.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number - The number of the current week in UTC time. The highest week number in a year is either 52 or 53.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-12-01 12:00:00"); // 49th week, 1st week in december
gs.info(gdt.getWeekOfYearUTC());
```

Scoped GlideDateTime - getYearLocalTime()

Gets the year stored by the GlideDateTime object, expressed in the current user's time zone.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number - Four-digit year value in the user's time zone.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-11-02 12:00:00");
gs.info(gdt.getYearLocalTime());
```
Scoped GlideDateTime - getYearUTC()

Gets the year stored by the GlideDateTime object, expressed in the UTC time zone.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number</td>
<td>4-digit year value in the UTC time zone.</td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime("2011-11-02 12:00:00");
gs.info(gdt.getYearUTC());

Scoped GlideDateTime - GlideDateTime()

Instantiates a new GlideDateTime object with the current date and time in Greenwich Mean Time (GMT).

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime();

Scoped GlideDateTime - GlideDateTime(GlideDateTime g)

Instantiates a new GlideDateTime object set to the time of the GlideDateTime object passed in the parameter.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>g</td>
<td>GlideDateTime</td>
<td>The GlideDateTime object to use for setting the time of the new object.</td>
</tr>
</tbody>
</table>
var start = new GlideDateTime("2011-01-01 12:00:00");
var end = new GlideDateTime(start);
gs.info(end);

Scoped GlideDateTime - GlideDateTime(String value)

Instantiates a new GlideDateTime object from a date and time value in the UTC time zone specified with the format yyyy-MM-dd HH:mm:ss.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>A UTC date and time using the internal format yyyy-MM-dd HH:mm:ss.</td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime("2011-01-01 12:00:00");

Scoped GlideDateTime - hasDate()

Determines if an object's date is set.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the object date is set; otherwise, returns false.</td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.info(gdt.hasDate());

Scoped GlideDateTime - isDST()

Determines if an object's time uses a daylight saving offset.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the time is daylight saving; otherwise, returns false.</td>
</tr>
</tbody>
</table>

```java
var gdt = new GlideDateTime("2014-08-31 00:00:00");
gs.info(gdt.isDST()); //true
```

Output:

**Scoped GlideDateTime - isValid()**

Determines if a value is a valid date and time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if value is valid; otherwise, returns false.</td>
</tr>
</tbody>
</table>

```java
var gdt = new GlideDateTime();
gdt.setDisplayValue("2011-aa-01 00:00:00");
gs.info(gdt.isValid());
```

Output: false

**Scoped GlideDateTime - onOrAfter(GlideDateTime gdt)**

Determines if the GlideDateTime object's data and time occurs on or after the specified GlideDateTime object's date and time.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gdt</td>
<td>GlideDateTime</td>
<td>Date and time to check against.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the GlideDateTime object's date and time is on or after the date and time specified by the parameter. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: GlideDateTime object date and time is on or after the specified object's date and time.</td>
</tr>
<tr>
<td></td>
<td>• false: GlideDateTime object date and time is before the specified object's date and time.</td>
</tr>
</tbody>
</table>

```
var gdt1 = new GlideDateTime("2016-05-09 10:11:12");
var gdt2 = new GlideDateTime("2017-06-12 15:11:12");
gs.info(gdt1.onOrAfter(gdt2));
```

Output

false

Scoped GlideDateTime - onOrBefore(GlideDateTime gdt)

Determines if the GlideDateTime object's data and time occurs on or before the specified GlideDateTime object's date and time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gdt</td>
<td>GlideDateTime</td>
<td>Date and time to check against.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the GlideDateTime object's date and time is on or before the date and time specified by the parameter. Possible values:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• true: GlideDateTime object date and time is on or before the specified object's date and time.</td>
<td></td>
</tr>
<tr>
<td>• false: GlideDateTime object date and time is after the specified object's date and time.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt1 = new GlideDateTime("2016-05-09 10:11:12");
var gdt2 = new GlideDateTime("2017-06-12 15:11:12");
gs.info(gdt1.onOrBefore(gdt2));
```

Output

```
true
```

**Scoped GlideDateTime - setDayOfMonthLocalTime(Number day)**

Sets the day of the month to a specified value in the current user's time zone.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>day</td>
<td>Number</td>
<td>The day of month to change to, from 1 to 31. If this value is greater than the maximum number of days in the month, the value is set to the last day of the month.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime();
gdt.setDayOfMonthLocalTime(9);
gs.info(gdt.getDayOfMonthLocalTime());
```

**Scoped GlideDateTime - setDayOfMonthUTC(Number day)**

Sets the day of the month to a specified value in the UTC time zone.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>day</td>
<td>Number</td>
<td>The day of month to change to, from 1 to 31. If this value is greater than</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the maximum number of days in the month, the value is set to the last day</td>
</tr>
<tr>
<td></td>
<td></td>
<td>of the month.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime();
gdt.setDayOfMonthUTC(9);
gs.info(gdt.getDayOfMonthUTC());
```

### Scoped GlideDateTime - setDisplayValue(String asDisplayed)

Sets a date and time value using the current user's display format and time zone.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>asDisplayed</td>
<td>String</td>
<td>The date and time in the current user's display format and time zone. The</td>
</tr>
<tr>
<td></td>
<td></td>
<td>parameter must be formatted using the current user's preferred display</td>
</tr>
<tr>
<td></td>
<td></td>
<td>format, such as MM-dd-yyyy HH:mm:ss. To assign the current date and time to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>a variable in a workflow script, use variable.setDisplayValue(gs.nowDateTime);.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2014-02-02 12:00:00");
gdt.setDisplayValue("2014-01-01 12:00:00"); //uses current user session time zone (US/Pacific)
gs.info(gdt.getValue());
```
Scoped GlideDateTime - setDisplayValue(String value, String format)

Sets a date and time value using the current user’s time zone and the specified
date and time format.

This method throws a runtime exception if the date and time format used in the
value parameter does not match the format parameter. You can retrieve the
error message by calling getErrorMsg() on the GlideDateTime object after the
exception is caught.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>Date and time in the current user’s time zone.</td>
</tr>
</tbody>
</table>
| format| String | Date and time format to use to parse the value parameter. Use the following values to describe the value parameter:  
• dd: Day of the month  
• MM: Month of the year  
• yyyy: Year  
• HH: Hour  
• mm: Minutes  
• ss: Seconds

For example: "dd-MM-yyyy HH:mm:ss" or "MM-dd-yyyy HH:mm".

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-02-02 12:00:00");
gdt.setDisplayValue("20-5-2011 12:00:00", "dd-MM-yyyy HH:mm:ss"); //uses current user session time zone
gs.info(gdt.getValue());
```

**Output**

"20-05-2011 12:00:00"
Scoped GlideDateTime - setDisplayValueInternal(String value)
Sets a date and time value using the internal format (yyyy-MM-dd HH:mm:ss) and the current user's time zone.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>The date and time in internal format.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var gdt = new GlideDateTime("2014-02-02 12:00:00");
gdt.setDisplayValueInternal("2014-01-01 12:00:00"); //uses current user session time zone (US/Pacific)
gs.info(gdt.getValue());
```

Scoped GlideDateTime - setGlideDateTime(GlideDateTime g)
Sets the date and time of the current object using an existing GlideDateTime object. This method is equivalent to instantiating a new object with a GlideDateTime parameter.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>g</td>
<td>GlideDateTime</td>
<td>The object to use for setting the datetime value.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var dt1 = new GlideDateTime("2011-01-01 12:00:00");
var dt2 = new GlideDateTime("2011-02-02 08:00:00");
dt1.setGlideDateTime(dt2);
gs.info(dt1.getValue());
```
Scoped GlideDateTime - setMonthLocalTime(Number month)

Sets the month stored by the GlideDateTime object to the specified value using the current user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>month</td>
<td>Number</td>
<td>The month to change to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime();
gdt.setMonthLocalTime(1);
gs.info(gdt.getMonthLocalTime());

Output: 1

Scoped GlideDateTime - setMonthUTC(Number month)

Sets the month stored by the GlideDateTime object to the specified value using the UTC time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>month</td>
<td>Number</td>
<td>The month to change to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime();
gdt.setMonthUTC(1);
gs.info(gdt.getMonthUTC());

Output: 1
Scoped GlideDateTime - setValue(String o)

Sets the date and time of the GlideDateTime object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>o</td>
<td>String</td>
<td>The date and time to use. This parameter may be one of several types:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• A string in the UTC time zone and the internal format of yyyy-MM-dd HH:mm:ss. Sets the value of the object to the specified date and time. Using the method this way is equivalent to instantiating a new GlideDateTime object using the GlideDateTime(String value) constructor. If the date and time format used does not match the internal format, the method attempts to set the date and time using other available formats. Resolving the date and time this way can lead to inaccurate data due to ambiguity in the day and month values. When using a non-standard date and time format, use setValueUTC(String dt, String format) instead.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• A GlideDateTime object. Sets the value of the object to the date and time stored by the GlideDateTime passed in the parameter. Using the method this way is equivalent to instantiating a new GlideDateTime object using the GlideDateTime(GlideDateTime g) constructor.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• A JavaScript Number. Sets the value of the object using the Number value as milliseconds past January 1, 1970 00:00:00 GMT.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-01-01 12:00:00");
gdt.setValue("2011-02-02 08:00:00"); // value set = 2011-02-02 08:00:00
gs.info(gdt.getValue());
```
Scoped GlideDateTime - setValueUTC(String dt, String format)

Sets a date and time value using the UTC time zone and the specified date and time format. This method throws a runtime exception if the date and time format used in the dt parameter does not match the format parameter. You can retrieve the error message by calling getErrorMsg() on the GlideDateTime object after the exception is caught.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dt</td>
<td>String</td>
<td>The date and time to use.</td>
</tr>
<tr>
<td>format</td>
<td>String</td>
<td>The date and time format to use.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-01-01 12:00:00");
gdt.setValueUTC("15-02-2011 08:00:00", "dd-MM-yyyy HH:mm:ss");
gs.info(gdt.getValue());
```

Scoped GlideDateTime - setYearLocalTime(Number year)

Sets the year stored by the GlideDateTime object to the specified value using the current user's time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>year</td>
<td>Number</td>
<td>The year to change to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var gdt = new GlideDateTime();
gdt.setYearLocalTime(2013);
gs.info(gdt.getYearLocalTime());

Output: 2013

Scoped GlideDateTime - setYearUTC(Number year)
Sets the year stored by the GlideDateTime object to the specified value using the UTC time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>year</td>
<td>Number</td>
<td>The year to change to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime();
gdt.setYearUTC(2013);
gs.info(gdt.getYearUTC());

Output: 2013

Scoped GlideDateTime - subtract(GlideTime time)
Subtracts a specified amount of time from the current GlideDateTime object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>GlideTime</td>
<td>The time value to subtract.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var gdt = new GlideDateTime("2011-08-31 08:00:00");
var gtime1 = new GlideTime();
gtime1.setValue("00:00:20");
gdt.subtract(gtime1);
var gtime2 = gdt.getTime();
gs.info(gtime2.getByFormat('hh:mm:ss'));

Output: 07:59:40

**Scoped GlideDateTime - subtract(Number milliseconds)**
Subtracts the specified number of milliseconds from the GlideDateTime object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>milliseconds</td>
<td>Number</td>
<td>Number of milliseconds to subtract.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gdt = new GlideDateTime("2011-12-07 08:00:00");
gdt.subtract(1000);
gs.info(gdt.getValue());

Output: 2011-12-07 07:59:59

**Scoped GlideDateTime - subtract(GlideDateTime start, GlideDateTime end)**
Gets the duration difference between two GlideDateTime values.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td>GlideDateTime</td>
<td>The start value.</td>
</tr>
<tr>
<td>End</td>
<td>GlideDateTime</td>
<td>The end value.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDuration</td>
<td>The duration between the two values.</td>
</tr>
</tbody>
</table>

```javascript
var gdt1 = new GlideDateTime("2011-08-28 09:00:00");
var gdt2 = new GlideDateTime("2011-08-31 08:00:00");

var dur = GlideDateTime.subtract(gdt1, gdt2); //the difference between gdt1 and gdt2
gs.info(dur.getDisplayValue());
```

Output: 2 Days 23 Hours

**Scoped GlideDateTime - toString()**

Gets the date and time value stored by the GlideDateTime object in the internal format, yyyy-MM-dd HH:mm:ss, and the system time zone, UTC by default. This method is equivalent to getValue().

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The date and time stored by the GlideDateTime object in the system time zone and format.</td>
</tr>
</tbody>
</table>

```javascript
var gdt = new GlideDateTime("2011-08-31 08:00:00");
gs.info(gdt.toString());
```

**GlideDBFunctionBuilder - Scoped, Global**

Build functions to perform SQL operations in the database.

The GlideDBFunctionBuilder methods provide a way to build Relational Database Management System (RDBMS) functions to perform SQL operations on record data. These methods can be used in both scoped and global server scripts.

To use platform functions:
• Construct a function using the GlideDBFunctionBuilder constructor and associated methods.

• After building a function, you can apply the function to the current record using the addFunction() method of the GlideRecord class.

• Add the function to a query using the addQuery() method of the GlideRecord class.

• Retrieve the results of the function using the existing GlideRecord API methods such as getValue() and getElement().

Scoped GlideDBFunctionBuilder - add()
Adds the values of two or more integer fields.

Use the field(String field) method to define fields on which the operation is performed.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var functionBuilder = new GlideDBFunctionBuilder();
var myAddingFunction = functionBuilder.add();
myAddingFunction = functionBuilder.field('order');
myAddingFunction = functionBuilder.field('priority');
myAddingFunction = functionBuilder.build();

Scoped GlideDBFunctionBuilder - build()
Builds the database function defined by the GlideDBFunctionBuilder object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var functionBuilder = new GlideDBFunctionBuilder();
var myAddingFunction = functionBuilder.add();
myAddingFunction = functionBuilder.field('order');
myAddingFunction = functionBuilder.field('priority');
myAddingFunction = functionBuilder.build();
gs.info(myAddingFunction);
```

Output:

```plaintext
*** Script: glidefunction:add(order,priority)
```

Scoped GlideDBFunctionBuilder - concat()

Concatenates the values of two or more fields.

Use the `field(String field)` method to define fields on which the operation is performed.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var functionBuilder = new GlideDBFunctionBuilder();
var myConcatFunction = functionBuilder.concat();
myConcatFunction = functionBuilder.field('short_description');
```
```javascript
myConcatFunction = functionBuilder.field('caller_id.name');
myConcatFunction = functionBuilder.build();
```

**Scoped GlideDBFunctionBuilder - constant(String constant)**

Defines a constant value to use in the function. If used with the `dayofweek()` method, the string defines whether to use Sunday or Monday as the first day of the week.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>constant</td>
<td>String</td>
<td>A constant value used in a function.</td>
</tr>
</tbody>
</table>

When used with the `dayofweek()` method, the value defines whether the week starts on a Sunday or Monday.

- **1**: Week begins on Sunday.
- **2**: Week begins on Monday.

This definition enables the `dayofweek()` method to return the correct day of the week from a given date. If a value other than 1 or 2 is provided, the `dayofweek()` method uses Sunday as the first day of the week.

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Scoped GlideDBFunctionBuilder - datediff()**

Determines the duration using a given start date/time and end date/time.

Use the `field(String field)` method to define start and end date/time fields.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var functionBuilder = new GlideDBFunctionBuilder();
var myDateDiffFunction = functionBuilder.datediff();
myDateDiffFunction = functionBuilder.field('sys_updated_on');
myDateDiffFunction = functionBuilder.field('opened_at');
myDateDiffFunction = functionBuilder.build();
```

Scoped GlideDBFunctionBuilder - dayofweek()

Returns an integer representing the day of the week for a given date.

Use the `field(String field)` method to define the given date/time. Use the `constant(String constant)` method to define whether the week starts on a Sunday or Monday.

This method can be used with MySQL, Oracle, and Microsoft SQL Server databases only. If using an Oracle database, the NLS_TERRITORY setting must be set to a territory with Sunday as the first day of the week.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td></td>
</tr>
</tbody>
</table>

If the first day of the week is set to Sunday in the `constant(String constant)` method, return values are associated with the following days of the week:

- 1: Sunday
- 2: Monday
- 3: Tuesday
- 4: Wednesday
- 5: Thursday
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• 6</td>
<td>Friday</td>
</tr>
<tr>
<td>• 7</td>
<td>Saturday</td>
</tr>
</tbody>
</table>

If the first day of the week is set to Monday:

| • 1  | Monday      |
| • 2  | Tuesday     |
| • 3  | Wednesday   |
| • 4  | Thursday    |
| • 5  | Friday      |
| • 6  | Saturday    |
| • 7  | Sunday      |

If a value other than 1 or 2 is provided in the `constant(String constant)` method, the `dayOfWeek()` method uses Sunday as the first day of the week.

```javascript
var functionBuilder = new GlideDBFunctionBuilder();
var dayOfWeekFunction = functionBuilder.dayOfWeek();
dayOfWeekFunction = functionBuilder.field('opened_at');
dayOfWeekFunction = functionBuilder.constant('2');
dayOfWeekFunction = functionBuilder.build();

var now_GR = new GlideRecord('incident');
now_GR.addFunction(dayOfWeekFunction);
now_GR.query();
while(now_GR.next())
gs.log(now_GR.getValue(dayOfWeekFunction));
```

**Scoped GlideDBFunctionBuilder - divide()**

Divides the value of one integer field by another.

Use the `field(String field)` method to define fields on which the operation is performed.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var functionBuilder = new GlideDBFunctionBuilder();
var myDivideFunction = functionBuilder.divide();
myDivideFunction = functionBuilder.field('order');
myDivideFunction = functionBuilder.field('priority');
myDivideFunction = functionBuilder.build();
```

**Scoped GlideDBFunctionBuilder - field(String field)**

Defines a field on which a SQL operation is performed.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>The field on which you are performing the SQL operation.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var functionBuilder = new GlideDBFunctionBuilder();
var myAddingFunction = functionBuilder.add();
myAddingFunction = functionBuilder.field('order');
myAddingFunction = functionBuilder.field('priority');
myAddingFunction = functionBuilder.build();
```

**Scoped GlideDBFunctionBuilder - GlideDBFunctionBuilder()**

Instantiates a GlideDBFunctionBuilder object.
### Scoped GlideDBFunctionBuilder - length()

Determines the number of code units in a field.

Use the `field(String field)` method to define fields on which the operation is performed.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

```javascript
define GlideDBFunctionBuilder()

Scoped GlideDBFunctionBuilder - length()

void functionBuilder = GlideDBFunctionBuilder();
definition myLengthFunction = functionBuilder.length();
definition myLengthFunction = functionBuilder.field('short_description');
definition myLengthFunction = functionBuilder.build();
```

### Scoped GlideDBFunctionBuilder - multiply()

Multiplies the values of two integer fields.

Use the `field(String field)` method to define fields on which the operation is performed.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
<tr>
<td>Returns</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

```javascript
var functionBuilder = new GlideDBFunctionBuilder();
var myMultiplyFunction = functionBuilder.multiply();
myMultiplyFunction = functionBuilder.field('order');
myMultiplyFunction = functionBuilder.field('priority');
myMultiplyFunction = functionBuilder.build();
```

**Scoped GlideDBFunctionBuilder - subtract()**

Subtracts the value of one integer field from another.

Use the `field(String field)` method to define fields on which the operation is performed.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

```javascript
var functionBuilder = new GlideDBFunctionBuilder();
var mySubtractFunction = functionBuilder.subtract();
mySubtractFunction = functionBuilder.field('order');
mySubtractFunction = functionBuilder.field('priority');
mySubtractFunction = functionBuilder.build();
```

**GlideDialogWindow - Client**

The GlideDialogWindow API provides methods for displaying a dialog in the current window and frame.
Use these methods in scripts anywhere that you can use a client-side JavaScript. These methods are most often called from a UI action with the Client check box selected.

💡 Note: This API has been deprecated, use the GlideModalV3 API instead.

GlideDialogWindow - adjustBodySize()

Adjusts the body height of a dialog window to be the window height minus the header height.

You typically call this method after calling GlideDialogWindow - setSize().

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

```javascript
var gdw = new GlideDialogWindow('show_list');
gdw.setTitle('Test');
gdw.setSize(750,300);
gdw.adjustBodySize();
gdw.render();
```

GlideDialogWindow - destroy()

Closes the dialog window.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>
GlideDialogWindow - GlideDialogWindow(String id, Boolean readOnly, Number width, Number height)

Provides methods for displaying a dialog in the current window and frame.

Use these methods in scripts anywhere that you can use a client-side JavaScript. These methods are most often called from a UI Action with the **Client** check box selected.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Name of the UI page to load into the dialog window.</td>
</tr>
<tr>
<td>readOnly</td>
<td>Boolean</td>
<td>Optional. Flag that indicates whether the dialog window is read only (true) or read/write (false). Default: false</td>
</tr>
<tr>
<td>width</td>
<td>Number</td>
<td>Optional. Size (in pixels) to set the width of the dialog window.</td>
</tr>
<tr>
<td>height</td>
<td>Number</td>
<td>Optional. Size (in pixels) to set the height of the dialog window.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
//Destroy the current dialog window.
GlideDialogWindow.get().destroy();
```

```java
// Creates a dialog window
var gdw = new GlideDialogWindow('show_list');
```

```java
// Creates a read-only dialog window
var gdw = new GlideDialogWindow('show_list', true);
```
// Creates a dialog window that is 400 pixels wide
var gdw = new GlideDialogWindow('show_list', false, 400);

// Creates a dialog window that is 400 pixels wide and 200 pixels tall
var gdw = new GlideDialogWindow('show_list', false, 400, 200);

**GlideDialogWindow - render()**
Renders the dialog window.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
var gdw = new GlideDialogWindow('show_list');
gdw.setTitle('Test');
gdw.setSize(750,300);
gdw.setPreference('table', 'u_test_list');
gdw.setPreference('title', 'A New Title');
gdw.render();
```

**GlideDialogWindow - removeCloseDecoration()**
Removes the close decoration in the upper right corner of the window.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
// Remove the close decoration from the current dialog window.
GlideDialogWindow.get().removeCloseDecoration();

GlideDialogWindow - setPreference(String name, String value)
Sets a given window property to a specified value.
Any window property can be set using this method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The window property to set.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The value for the window property.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdw = new GlideDialogWindow('show_list');
gdw.setTitle('Test');
gdw.setSize(750,300);
gdw.setPreference('table', 'u_test_list');
gdw.setPreference('title', 'A New Title');
```

GlideDialogWindow - setSize(Number width, Number height)
Sets the size of the dialog window.
If you do not pass width and height parameters, a default size is used.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>The width of the dialog window.</td>
</tr>
<tr>
<td>height</td>
<td>Number</td>
<td>The height of the dialog window.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gdw = new GlideDialogWindow('show_list');
gdw.setSize(750, 300);
```

**GlideDialogWindow - setTitle(String title)**

Sets the title of the dialog window.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>String</td>
<td>The title for the current window.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
//var gdw = new GlideDialogWindow('show_list');
gdw.setTitle('test');
```

**GlideDigest - Scoped**

The scoped GlideDigest class provides methods for creating a message digest from strings or input streams using MD5, SHA1, or SHA256 hash algorithms.

**Scoped GlideDigest - getMD5Base64(String source)**

Create a message digest from a string using the MD5 algorithm. The output string is in Base64.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>String</td>
<td>The source string.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>---------------------------</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>The message digest.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var inputString = "Her molasses flowed slowly down the hill."
var digest = new GlideDigest();
gs.info(digest.getMD5Base64(inputString));
```

**Scoped GlideDigest - getMD5Base64FromInputStream( GlideScriptableInputStream inputStream)**

Create a message digest from an input stream using the MD5 algorithm. The output string is in Base64.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputStream</td>
<td>GlideScriptableInputStream</td>
<td>The source input stream.</td>
</tr>
</tbody>
</table>

```javascript
var inputStream = new GlideSysAttachment().getContentStream(attachmentSysID);
var digest = new GlideDigest();
gs.info(digest.getMD5Base64FromInputStream(inputStream));
```

**Scoped GlideDigest - getMD5Hex(String source)**

Create a message digest from a string using the MD5 algorithm. The output string is in hexadecimal.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>String</td>
<td>The source string.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The message digest.</td>
</tr>
</tbody>
</table>

```javascript
var inputString = "Her molasses flowed slowly down the hill.";
var digest = new GlideDigest();
gs.info(digest.getMD5Hex(inputString));
```

**Scoped GlideDigest - getMD5HexFromInputStream( GlideScriptableInputStream inputStream)**

Create a message digest from an input stream using the MD5 algorithm. The output string is in hexadecimal.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputStream</td>
<td>GlideScriptableInputStream</td>
<td>The source input stream.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The message digest.</td>
</tr>
</tbody>
</table>

```javascript
var inputStream = new GlideSysAttachment().getContentStream(attachmentSysID);
var digest = new GlideDigest();
gs.info(digest.getMD5HexFromInputStream(inputStream));
```

**Scoped GlideDigest - getSHA1Base64(String source)**

Create a message digest from a string using the SHA1 algorithm. The output string is in Base64.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>String</td>
<td>The source string.</td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>The message digest.</td>
<td></td>
</tr>
</tbody>
</table>

```
var inputString = "Her molasses flowed slowly down the hill.";
var digest = new GlideDigest();
gs.info(digest.getSHA1Base64(inputString));
```

**Scoped GlideDigest - getSHA1Base64FromInputStream( GlideScriptableInputStream inputStream)***

Create a message digest from an input stream using the SHA1 algorithm. The output string is in Base64.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>inputStream</td>
<td>GlideScriptableInputStream</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>String</td>
<td>The message digest.</td>
</tr>
</tbody>
</table>

```
var inputStream = new GlideSysAttachment().getContentStream(attachmentSysID);
var digest = new GlideDigest();
gs.info(digest.getSHA1Base64FromInputStream(inputStream));
```

**Scoped GlideDigest - getSHA1Hex(String source)**

Create a message digest from a string using the SHA1 algorithm. The output string is in hexadecimal.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>source</td>
<td>String</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The message digest.</td>
</tr>
</tbody>
</table>

var inputString = "Her molasses flowed slowly down the hill.";
var digest = new GlideDigest();
gs.info(digest.getSHA1Hex(inputString));

Scoped GlideDigest - getSHA1HexFromInputStream(GlideScriptableInputStream inputStream)

Create a message digest from an input stream using the SHA1 algorithm. The output string is in hexadecimal.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputStream</td>
<td>GlideScriptableInputStream</td>
<td>The source input stream.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The message digest.</td>
</tr>
</tbody>
</table>

var inputStream = new GlideSysAttachment().getContentStream(attachmentSysID);
var digest = new GlideDigest();
gs.info(digest.getSHA1HexFromInputStream(inputStream));

Scoped GlideDigest - getSHA256Base64(String source)

Create a message digest from a string using the SHA256 algorithm. The output string is in Base64.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>String</td>
<td>The source string.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The message digest.</td>
</tr>
</tbody>
</table>

```
var inputString = "Her molasses flowed slowly down the hill.";
var digest = new GlideDigest();
gs.info(digest.getSHA256Base64(inputString));
```

**Scoped GlideDigest - getSHA256Base64FromInputStream( GlideScriptableInputStream inputStream)**

Create a message digest from an input stream using the SHA256 algorithm. The output string is in Base64.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputStream</td>
<td>GlideScriptableInputStream</td>
<td>The source input stream.</td>
</tr>
</tbody>
</table>

```
var inputStream = new GlideSysAttachment().getContentStream(attachmentSysID);
var digest = new GlideDigest();
gs.info(digest.getSHA256Base64FromInputStream(inputStream));
```

**Scoped GlideDigest - getSHA256Hex(String source)**

Create a message digest from a string using the SHA256 algorithm. The output string is in hexadecimal.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>String</td>
<td>The source string.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The message digest.</td>
</tr>
</tbody>
</table>

var inputString = "Her molasses flowed slowly down the hill.";
var digest = new GlideDigest();
gs.info(digest.getSHA256Hex(inputString));

Scoped GlideDigest -
getSHA256HexFromInputStream(GlideScriptableInputStream inputStream)
Create a message digest from an input stream using the SHA256 algorithm. The output string is in hexadecimal.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputStream</td>
<td>GlideScriptableInputStream</td>
<td>The source input stream.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The message digest.</td>
</tr>
</tbody>
</table>

var inputStream = new GlideSysAttachment().getContentStream(attachmentSysID);
var digest = new GlideDigest();
gs.info(digest.getSHA256HexFromInputStream(inputStream));

Scoped GlideDigest - GlideDigest()
Creates an instance of scoped GlideDigest.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
GlideDocumentV3 - Client

The GlideDocument class provides the ability to search a DOM element, a document, or a JQuery element.

The GlideDocumentV3 API can be used in client-side scripts using ListV2 and ListV3 APIs. The GlideDocument APIs are accessed using the g_document global object.

GlideDocumentV3 - getElement(String selector, Element context)

Returns a node found in the specified DOM based context or created from the HTML context.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selector</td>
<td>String or Object</td>
<td>Selector expression</td>
</tr>
<tr>
<td>context</td>
<td>String or Object</td>
<td>Optional. DOM Element, document, or JQuery object to search.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>node</td>
<td>Node that matches the selector.</td>
</tr>
</tbody>
</table>

This example shows how to get the list view name value from a UI page.

```javascript
// HTML entry
<input type="hidden" id="list_view" name="list_view" value="${sysparm_list_view}" />

// Client script
var listView = g_document.getElementById('#list_view').value;
```

GlideDocumentV3 - getElements(String selector, Element context)

Returns a node list found in the specified DOM based context or created if an HTML context is specified.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selector</td>
<td>String or Object</td>
<td>The selector expression</td>
</tr>
<tr>
<td>context</td>
<td>String or Object</td>
<td>(Optional) A DOM Element, document, or JQuery object to be searched.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>node list</td>
<td>A list of nodes that matches the selector.</td>
</tr>
</tbody>
</table>

### GlideDuration - Scoped

The GlideDuration class provides methods for working with spans of time known as durations.

GlideDuration objects store the duration as the number of days and time from January 1, 1970, 00:00:00. As a result, `setValue()` and `getValue()` use the scoped GlideDateTime object for parameters and return values.

#### Scoped GlideDuration - add(GlideDuration duration)

Adds the duration of the specified GlideDuration object to the current GlideDuration object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>duration</td>
<td>GlideDuration</td>
<td>GlideDuration object that contains the duration value to add to the current GlideDuration object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDuration</td>
<td>New GlideDuration object whose duration is the sum of the durations of the two GlideDuration objects.</td>
</tr>
</tbody>
</table>

```javascript
var duration = new GlideDuration('3 12:00:00');
var duration2 = new GlideDuration('3:00:00');
```
```javascript
var answer = duration.add(duration2);
gs.info(answer.getDisplayValue());
```

Output

3 Days 15 Hours

**Scoped GlideDuration - getByFormat(String format)**

Returns the duration value in the specified format.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>format</td>
<td>String</td>
<td>Duration format. Format: <a href="https://docs.servicenow.com/t%D0%BC%D0%B5%D0%B6%D0%B4%D1%83a/8.2/content/ux-sdk-development-guide.html">Global date and time field format</a></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Current duration in the specified format.</td>
</tr>
</tbody>
</table>

```javascript
var dur = new GlideDuration('3 22:00:00');
gs.info(dur.getByFormat('HH:mm'));
```

Output

22:00

**Scoped GlideDuration - getDayPart()**

Returns the number of days.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>Number</td>
<td>Number of days in the duration.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var dur = new GlideDuration('3 12:00:00');
gs.info(dur.getDayPart());
```

**Output**

| 3 |

**Scoped GlideDuration - getDisplayValue()**

Returns the display value of the duration in number of days, hours, and minutes.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Number of days, hours, and minutes, such as 2 Days 10 Hours 36 Minutes.</td>
</tr>
<tr>
<td></td>
<td>Format: Display value: &quot;n&quot; Days &quot;n&quot; Hours &quot;n&quot; Minutes</td>
</tr>
</tbody>
</table>

```javascript
var dur = new GlideDuration('3 12:00:00');
gs.info(dur.getDisplayValue());
```

**Output**

| 3 Days 12 Hours |

**Scoped GlideDuration - GlideDuration()**

Instantiates a GlideDuration object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Scoped GlideDuration - getDurationValue()

Returns the duration value in "d HH:mm:ss" format.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Duration value. Format: d HH:mm:ss where &quot;d&quot; is number of days.</td>
</tr>
</tbody>
</table>

```javascript
var dur = new GlideDuration(60000);
gs.info(dur.getDurationValue());
```

Output

00:01:00

Scoped GlideDuration - getRoundedDayPart()

Returns the rounded number of days. If the time part is more than 12 hours, the return value is rounded up. Otherwise, it is rounded down.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Day value of the display value rounded.</td>
</tr>
</tbody>
</table>

```javascript
var dur = new GlideDuration('3 14:00:00');
gs.info(dur.getRoundedDayPart());
```

Output

4

Scoped GlideDuration - getValue()

Returns the internal date/time value of the current GlideDuration object.

GlideDuration objects store the duration as a date and time from January 1, 1970, 00:00:00.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Current duration within the GlideDuration object.</td>
</tr>
<tr>
<td></td>
<td>Format: YYYY-MM-DD HH:mm:ss</td>
</tr>
</tbody>
</table>

```javascript
var dur = new GlideDuration('3 12:00:00');
gs.info(dur.getValue());
```

Output:

1970-01-04 12:00:00

Scoped GlideDuration - GlideDuration(GlideDuration another)

Instantiates a GlideDuration object by cloning the values of the passed in GlideDuration object.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>another</td>
<td>GlideDuration</td>
<td>GlideDuration object.</td>
</tr>
</tbody>
</table>

```javascript
var duration = new GlideDuration('3 12:00:00');
var duration2 = new GlideDuration(duration);
gs.info(duration2.getDisplayValue());
```

**Output**

3 Days 12 Hours

---

### Scoped GlideDuration - GlideDuration(Number milliseconds)

Instantiates a GlideDuration object with the specified duration in milliseconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>milliseconds</td>
<td>Number</td>
<td>Duration value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unit: Milliseconds</td>
</tr>
</tbody>
</table>

```javascript
var dur = new GlideDuration(60000);
gs.info(dur.getDurationValue());
```

**Output**

00:01:00

---

### Scoped GlideDuration - GlideDuration(String displayValue)

Instantiates a GlideDuration object with the specified duration display value.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>displayValue</td>
<td>String</td>
<td>Duration value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Format: d HH:mm:ss where &quot;d&quot; is number of days.</td>
</tr>
</tbody>
</table>

```javascript
var duration = new GlideDuration('3 12:00:00');
var duration2 = new GlideDuration('3:00:00');
```
```javascript
var answer = duration.add(duration2);
gs.info(answer.getDisplayValue());
```

**Output**

```
3 Days 15 Hours
```

### Scoped GlideDuration - setDisplayValue(String asDisplayed)

Sets the duration display value.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>asDisplayed</td>
<td>String</td>
<td>Display duration value to set. Format: d HH:mm:ss where &quot;d&quot; is number of days</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var dur = new GlideDuration();
dur.setDisplayValue('3 08:00:00');
gs.info(dur.getDisplayValue());
```

**Output**

```
3 Days 8 Hours
```

### Scoped GlideDuration - setValue(Object o)

Sets the internal date/time value of the GlideDuration object.

The method sets the duration value to the difference of the passed in date/time the base date/time value of January 1, 1970, 00:00:00. The passed in date/time object (string) is parsed into a GlideDateTime object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>o</td>
<td>Object</td>
<td>Date and time to use as the endpoint for the calculated duration time.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Format: YYYY-MM-DD HH:mm:ss</td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var dur = new GlideDuration();
dur.setValue('1970-01-05 08:00:00'); // sets internal DateTime value. The String is parsed into a GlideDateTime object.
gs.info(dur.getDisplayValue());
```

Output:

```
4 Days 8 Hours
```

**Scoped GlideDuration - subtract(GlideDuration duration)**

Subtracts the duration of the specified GlideDuration object to the current GlideDuration object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>duration</td>
<td>GlideDuration</td>
<td>GlideDuration object that contains the duration value to subtract from the current GlideDuration object.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDuration</td>
<td>New GlideDuration object whose duration contains the result of the subtraction of the duration of the two GlideDuration objects.</td>
</tr>
</tbody>
</table>

```javascript
var duration = new GlideDuration('3 12:00:00');
var duration2 = new GlideDuration('3:00:00');
var answer = duration.subtract(duration2);
gs.info(answer.getDisplayValue());
```
Output:

3 Days 9 Hours

**GlideElement - Global**

The GlideElement API provides a number of convenient script methods for dealing with fields and their values. GlideElement methods are available for the fields of the current GlideRecord.

**GlideElement - canCreate()**

Determines if the user's role permits the creation of new records in this field.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Boolean</td>
</tr>
</tbody>
</table>

**Scoped equivalent**

To use the canCreate() method in a scoped application, use the corresponding scoped method: **Scoped GlideElement - canCreate()**.

**GlideElement - canRead()**

Determines whether the user’s role permits them to read the associated GlideRecord.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the field can be read, false otherwise.</td>
</tr>
</tbody>
</table>

### Scoped equivalent

To use the `canRead()` method in a scoped application, use the corresponding scoped method: `Scoped GlideElement - canRead()`.

### GlideElement - canWrite()

Determines whether the user's role permits them to write to the associated GlideRecord.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the user can write to the field, false otherwise.</td>
</tr>
</tbody>
</table>

### Scoped equivalent

To use the `canWrite()` method in a scoped application, use the corresponding scoped method: `Scoped GlideElement - canWrite()`.

### GlideElement - changes()

Determines if the current field has been modified. This functionality is available for all available data types, except Journal fields.

> **Note:** The `changes()` method is not supported within ACL scripts.

> **Note:** If the GlideRecord on which you are performing this method has only been initialized and read, and has not been written, the underlying before-and-after values are the same. In this case, the method returns "false", as there has been no change to the data store.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the field has changed, false otherwise.</td>
</tr>
</tbody>
</table>

**Scoped equivalent**

To use the `changes()` method in a scoped application, use the corresponding scoped method: **Scoped GlideElement - changes()**.

**GlideElement - changesFrom(Object value)**

Determines if the previous value of the current field matches the specified object.

⚠️ **Note:** If the GlideRecord on which you are performing this method has only been initialized and read, and has not been written, the underlying before-and-after values are the same. In this case, the method returns "false", as there has been no change to the data store.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Object</td>
<td>An object value to check against the previous value of the current field.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the previous value matches the parameter, false if it does not.</td>
</tr>
</tbody>
</table>

```java
if (theState.changesTo(resolvedState)) {
    operation = 4; //Resolved
}
else if (theState.changesTo(closedState)) {
```
operation = 11; //Resolution Accepted
} else if (theState.changesFrom(resolvedState) || theState.changesFrom(closedState)) {
    operation = 10; //Re-open
} else {
    operation = 6; //Update
}

### Scoped equivalent

To use the `changesFrom()` method in a scoped application, use the corresponding scoped method: `Scoped GlideElement - changesFrom(Object o)`.

### GlideElement - changesTo(Object value)

Determines if the new value of a field, after a change, matches the specified object.

> **Note:** The `changesTo()` method is not supported within ACL scripts.

> **Note:** If the GlideRecord on which you are performing this method has only been initialized and read, and has not been written, the underlying before-and-after values are the same. In this case, the method returns “false”, as there has been no change to the data store.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Object</td>
<td>An object value to check against the new value of the current field.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the new value matches the parameter, false if it does not.</td>
</tr>
</tbody>
</table>

if (theState.changesTo(resolvedState)) {
    operation = 4; //Resolved
} else if (theState.changesTo(closedState)) {

operation = 11; //Resolution Accepted
}

else if (theState.changesFrom(resolvedState) || theState.changesFrom(closedState)) {
    operation = 10; //Re-open
}
else {
    operation = 6; //Update
}

Scoped equivalent

To use the changesTo() method in a scoped application, use the corresponding scoped method: Scoped GlideElement - changesTo(Object o).

GlideElement - dateNumericValue()

Returns the number of milliseconds since January 1, 1970, 00:00:00 GMT for a duration field. Does not require the creation of a GlideDateTime object because the duration field is already a GlideDateTime object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of milliseconds since January 1, 1970, 00:00:00 GMT.</td>
</tr>
</tbody>
</table>

var inc = new GlideRecord('incident');
inc.get('17c90efb13418700cc36b1422244b05d');
gs.info(inc.calendar_duration.dateNumericValue());

Output: 98000

Scoped equivalent

To use the dateNumericValue() method in a scoped application, use the corresponding scoped method: Scoped GlideElement - dateNumericValue().
**GlideElement - debug(Object o)**

Debugs the object and adds debug messages using `setError(String)`.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>o</td>
<td>Object</td>
<td>An object to debug.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideElement - getAttribute(String attributeName)**

Returns the value of the specified attribute from the dictionary.

If the attribute is a boolean attribute, use `getBooleanAttribute(String)` to get the value as a boolean rather than as a string.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributeName</td>
<td>String</td>
<td>Attribute name</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Attribute value</td>
</tr>
</tbody>
</table>

```javascript
function doit() {
  var now_GR = new GlideRecord('sys_user');
  now_GR.query("user_name","admin");
  if (now_GR.next()) {
    gs.print("we got one");
    gs.print(now_GR.location.getAttribute("tree_picker"));
  }
}
```
Scoped equivalent

To use the `getAttribute()` method in a scoped application, use the corresponding scoped method: `Scoped GlideElement - getAttribute(String attributeName)`.

`GlideElement - getBaseTableName()`

Gets the base table of the field.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

`GlideElement - getBooleanAttribute(String attributeName)`

Returns the Boolean value of the specified attribute from the dictionary.

To get the value as a string, use `getAttribute(string)`.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>attributeName</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Boolean</td>
</tr>
</tbody>
</table>
Scoped equivalent

To use the `getBooleanAttribute()` method in a scoped application, use the corresponding scoped method: `Scoped GlideElement - getBooleanAttribute(String attributeName)`.

GlideElement - `getChoices(String value)`

Generates a choice list for a field. Returns the choice values from the base table only, not from the extended table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>An optional dependent value.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>array list</td>
<td>The choice values for the field.</td>
</tr>
</tbody>
</table>

Scoped equivalent

To use the `getChoices()` method in a scoped application, use the corresponding scoped method: `Scoped GlideElement - getChoices(String dependent)`.

GlideElement - `getChoiceValue()`

Gets the choice label for the current choice value.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The choice label.</td>
</tr>
</tbody>
</table>
Scoped equivalent

To use the `getChoiceValue()` method in a scoped application, use the corresponding scoped method: `Scoped GlideElement - getChoiceValue()`.

GlideElement - `getDebugCount()`

Gets the number of debug messages logged by `debug()`.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Number</td>
</tr>
</tbody>
</table>

GlideElement - `getDependent()`

Checks whether or not the field is dependent on another field.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

GlideElement - `getDependentTable()`

Gets the table that the current table depends on.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the table.</td>
</tr>
</tbody>
</table>

**GlideElement - getDisplayValue(Number maxChar)**

Gets the formatted display value of the field.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxChar</td>
<td>Number</td>
<td>Optional, maximum number of characters to return.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Display value of the field.</td>
</tr>
</tbody>
</table>

```javascript
var fields = current.getFields();
for (var i = 0; i < fields.size(); i++) {
    var field = fields.get(i);
    var name = field.getName();
    var value = field.getDisplayValue();
    gs.print(i + ". " + name + "=" + value);
}
```

**Scoped equivalent**

To use the `getDisplayValue()` method in a scoped application, use the corresponding scoped method: **Scoped GlideElement - getDisplayValue(Number maxCharacters)**.
GlideElement - getDisplayValueExt(Number maxChar, String nullSub)

Gets the formatted display value of a field, or a specified substitute value if the display value is null or empty.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxChar</td>
<td>Number</td>
<td>Optional, the maximum number of characters to be returned.</td>
</tr>
<tr>
<td>nullSub</td>
<td>String</td>
<td>The value to return if the display value is null or empty.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The formatted display value of the field, or the specified substitute value.</td>
</tr>
</tbody>
</table>

GlideElement - getED()

Returns an element descriptor, which provides information about specific fields, rather than the data inside of those fields.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ElementDescriptor</td>
<td>The field's element descriptor.</td>
</tr>
</tbody>
</table>

This example gets the fields and field descriptors for the current record.

```javascript
var fields = current.getFields();
for (i=0; i<fields.size(); i++) {
    var field = fields.get(i);
    var descriptor = field.getED();
    gs.print("type=" + descriptor.getType() + " name=" + descriptor.getName());
```
Scoped equivalent

To use the `getED()` method in a scoped application, use the corresponding scoped method: **Scoped GlideElement - getED().**

GlideElement - `getElementValue(String value)`

Returns the value for the specified element.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>Element whose value you want returned.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the element.</td>
</tr>
</tbody>
</table>

```javascript
var fields = current.getFields();
for (var i = 0; i < fields.size(); i++) {
    var field = fields.get(i);
    var name = field.getName();
    // Returns the unformatted value of the element
    var value = field.getElementValue(name);
    var disValue = field.getDisplayValue();
    gs.print(i + " . " + name + " = " + value + ' display value = ' + disValue);
}
```

**Output**

```
1. cmdb_ci = 109562a3c611227500a7b7ff98cc0dc7 display value = Storage Area Network 001
2. impact = 2 display value = 2 - Medium
```

GlideElement - `getError()`

Gets error debug messages.
### GlideElement - getEscapedValue()

Gets the escaped value for the current element.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A string of debug messages</td>
</tr>
</tbody>
</table>

### GlideElement - getFieldStyle()

Gets the CSS style for the field.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The CSS style for the field.</td>
</tr>
</tbody>
</table>
```javascript
var fields = current.getFields();
for (var i = 0; i < fields.size(); i++) {
    var field = fields.get(i);
    var css_style = field.getFieldStyle();
    gs.print("CSS style" + "=" + css_style);
}
```

**GlideElement - getGlideObject()**

Gets a glide object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>A Glide object.</td>
</tr>
</tbody>
</table>

```javascript
function calcDateDelta(start, end, calendar) {
    var cal = GlideCalendar.getCalendar(calendar);
    if (!cal.isValid())
        return null;
    var realStart = start.getGlideObject();
    var realEnd = end.getGlideObject();
    var duration = cal.subtract(realStart, realEnd);
    return duration;
}
```

**GlideElement - getGlideRecord()**

Gets a glide record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
var grInc = new GlideRecord('incident');
grInc.get('sys_id','ef43c6d40a0a0b5700c77f9bf387afe3');
gs.info("Initial grInc - " + grInc.getDisplayValue());

var caller = grInc.getElementById("caller_id");
doit(caller);

function doit(caller) {
    var now_GR = caller.getGlideRecord();
    gs.info("doit gr is - " + now_GR.getDisplayValue());
}

Output

*** Script: Initial grInc - INC0000050
*** Script: doit gr is - INC0000050

GlideElement - getHTMLValue(Number maxChars)

Returns the HTML value of a field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxChars</td>
<td>Number</td>
<td>Optional. Maximum number of characters to return.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>HTML value for the field.</td>
</tr>
</tbody>
</table>

var inccause = new GlideRecord("incident");
inccause.short_description = current.short_description;
inccause.comments = current.comments.getHTMLValue();
inccause.insert();
Scoped equivalent

To use the `getHTMLValue()` method in a scoped application, use the corresponding scoped method: `Scoped GlideElement - getHTMLValue(Number maxChars)`.

**GlideElement - getHTMLValueExt(Number maxChar, String nullSub)**
Returns the HTML value of a field, or a specified substitute value if the HTML value is null or empty.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxChar</td>
<td>Number</td>
<td>The maximum number of characters to return.</td>
</tr>
<tr>
<td>nullSub</td>
<td>String</td>
<td>The value to return if the HTML value is null or empty.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The HTML value or the specified substitute value.</td>
</tr>
</tbody>
</table>

**GlideElement - getJournalEntry(Number mostRecent)**
Returns either the most recent journal entry or all journal entries.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mostRecent</td>
<td>Number</td>
<td>If 1, returns the most recent entry. If -1, returns all journal entries.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>For the most recent entry, returns a string that contains the field label, timestamp, and user display name of the journal entry. For all journal entries, returns the same information for all journal entries ever entered as a single string with each entry delimited by &quot;\n\n&quot;.</td>
</tr>
</tbody>
</table>
// gets all journal entries as a string where each entry is delimited by '\n\n'
var notes = current.work_notes.getJournalEntry(-1);
// stores each entry into an array of strings
var na = notes.split("\n\n");

for (var i = 0; i < na.length; i++)
    gs.print(na[i]);

**Scoped equivalent**

To use the `getJournalEntry()` method in a scoped application, use the corresponding scoped method: **Scoped GlideElement - getJournalEntry(Number mostRecent).**

**GlideElement - getLabel()**

Returns the object's label.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Object's label</td>
</tr>
</tbody>
</table>

### Example

```javascript
var now_GR = new GlideRecord("sc_req_item");
now_GR.addQuery("request", current.sysapproval);
now_GR.query();
while(now_GR.next()) {
    var nicePrice = now_GR.price.toString();
    if (nicePrice != ) {
        nicePrice = parseFloat(nicePrice);
        nicePrice = nicePrice.toFixed(2);
    }
    template.print(now_GR.number +": " + now_GR.quantity + " X " +
now_GR.cat_item.getDisplayValue() + " at $" + nicePrice + " each \n");
    template.print(" Options:
");
    var variables = now_GR.variables.getElements();
```
for (var key in variables) {
    var now_V = variables[key];
    if (now_V.getQuestion().getLabel() != ) {
        template.space(4);
        template.print('     ' +  now_V.getQuestion().getLabel() + " = " +
        now_V.getDisplayValue() + "\n");
    }
}

Scoped equivalent
To use the `getLabel()` method in a scoped application, use the corresponding scoped method: **Scoped GlideElement - `getLabel()`**.

**GlideElement - `getName()`**
Returns the name of the field.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

Scoped equivalent
To use the `getName()` method in a scoped application, use the corresponding scoped method: **Scoped GlideElement - `getName()`**.

**GlideElement - `getRefRecord()`**
Returns a GlideRecord object for a given reference element.

⚠️ **Warning:** If the reference element does not contain a value, it returns an empty GlideRecord object, not a NULL object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>A GlideRecord object</td>
</tr>
</tbody>
</table>

```javascript
var grINC = new GlideRecord('incident');
grINC.notNullQuery('caller_id');
grINC.query();
if (grINC.next()) {

  // Get a GlideRecord object for the referenced sys_user record
  var grUSER = grINC.caller_id.getRefRecord();
  if (grUSER.isValidRecord())
    gs.print( grUSER.getValue('name') );
}
```

Scoped equivalent

To use the `getRefRecord()` method in a scoped application, use the corresponding scoped method: **Scoped GlideElement - getRefRecord()**.

**GlideElement - getStyle()**

Get the CSS style for the value.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The CSS style for the value.</td>
</tr>
</tbody>
</table>
// Get string of style field from Field Style record
var cssStyle = now_GR.state.getStyle();

**GlideElement - getTableName()**

Returns the name of the field's table.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the table. This may be different from the table Class that the record is in. See Tables and Classes in the product documentation.</td>
</tr>
</tbody>
</table>

```javascript
if (current.approver.getTableName() == "sysapproval_approver") {
    if (current.approver == email.from_sys_id) {
        current.comments = "reply from: " + email.from + "\n\n" + email.body_text;

        // if it's been cancelled, it's cancelled.
        var doit = true;
        if (current.state=='cancelled')
            doit = false;

        if (email.body.state != undefined)
            current.state= email.body.state;

        if (doit)
            current.update();
    } else {
        gs.log("Approval for task ("+current.sysapproval.getDisplayValue()+") rejected because user sending email( "+email.from+") does not match the approver ("+current.approver.getDisplayValue()+")");
    }
}
```
Scoped equivalent

To use the `getTableName()` method in a scoped application, use the corresponding scoped method: `Scoped GlideElement - getTableName()`.

**GlideElement - getTextAreaDisplayValue()**

Retrieves the value and escapes the HTML.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

**GlideElement - getXHTMLValue()**

Retrieves the XHTML value of a field.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

**GlideElement - getXMLValue()**

Gets the XML value of a field as a string.
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td></td>
<td>The XML value</td>
</tr>
</tbody>
</table>

**GlideElement - hasAttribute(String attributeName)**

Determines whether a field has a particular attribute.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>attributeName</td>
<td>String</td>
<td>The attribute to check for</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td></td>
<td>True if the field has the attribute, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var totalCritical = 0;

var filledCritical = 0; var fields = current.getFields(); gs.print(fields); for (var num = 0; num < fields.size(); num++) {
    gs.print("RUNNING ARRAY VALUE " + num);
    var ed = fields.get(num).getED();
    if(ed.hasAttribute("tiaa_critical")) {
        gs.print("CRITICAL FIELD FOUND");
        totalCritical ++;
        if (!fields.get(num).isNil()) {
            filledCritical ++;
        }
    }
}
```
var answer = 0; gs.print("TOTAL = " + totalCritical); gs.print("FILLED = " + filledCritical); if (filledCritical &gt; 0 &amp;&amp; totalCritical &gt; 0) {
    var pcnt = (filledCritical/totalCritical)*100;
    answer = pcnt.toFixed(2);
}

GlideElement - hasRightsTo(String operationName)
Determines if the user has the right to perform a particular operation.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operationName</td>
<td>String</td>
<td>Name of the operation to check for</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the user has permission to perform the operation, false otherwise.</td>
</tr>
</tbody>
</table>

GlideElement - hasValue()
Determines if the field has a value.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the field has a value, false otherwise.</td>
</tr>
</tbody>
</table>

GlideElement - nil()
Determines whether the field is null.
### GlideElement - `setValue(String value)`

Sets the duration field to a value in the duration format. Does not require the creation of a GlideDateTime object because the duration field is already a GlideDateTime object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>Value to set to the duration field</td>
</tr>
</tbody>
</table>

**Returns**

- `void`

```java
if (current.start_date.changes() || current.end_date.changes() ||
    current.assigned_to.changes()) {
    if (!current.start_date.nil() && !current.end_date.nil() && !current.assigned_to.nil()) {
        gs.eventQueue("change.calendar.notify", current, current.assigned_to,
                      previous.assigned_to);
    }
}
```

### Scoped equivalent

To use the `nil()` method in a scoped application, use the corresponding scoped method: `Scoped GlideElement - nil()`.

### GlideElement - `setDateNumericValue(Number milliseconds)`

Sets the duration field to a number of milliseconds since January 1, 1970, 00:00:00 GMT for a duration field. Does not require the creation of a GlideDateTime object because the duration field is already a GlideDateTime object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>milliseconds</td>
<td>Number</td>
<td>Number of milliseconds spanned by the duration.</td>
</tr>
</tbody>
</table>

**Returns**

- `void`
var inc = new GlideRecord('incident');
inc.get('17c90efb13418700cc36b1422244b05d');
var timems = inc.calendar_duration.dateNumericValue();
timems = timems + 11*1000;
inc.calendar_duration.setDateNumericValue(timems)
gs.info(inc.calendar_duration.getValue());

Output: 1970-01-01 00:01:38

Scoped equivalent

To use the `setDateNumericValue()` method in a scoped application, use the corresponding scoped method: `Scoped GlideElement - setDateNumericValue(Number milliseconds)`.

**GlideElement - setDisplayValue(Object displayValue)**

Sets the display value of the field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>displayValue</td>
<td>Object</td>
<td>Value to be displayed.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Scoped equivalent

To use the `setDisplayValue()` method in a scoped application, use the corresponding scoped method: `Scoped GlideElement - setDisplayValue(Object value)`.

**GlideElement - setError(String message)**

Adds an error message.

Can be retrieved using `getError()`.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```java
if (!current.u_date1.nil() && !current.u_date2.nil()) {
    var start = current.u_date1.getGlideObject().getNumericValue();
    var end = current.u_date2.getGlideObject().getNumericValue();
    if (start > end) {
        gs.addInfoMessage('start must be before end');
        current.setAbortAction(true);
        current.u_date1.setError('start must be before end');
    }
}
```

### Scoped equivalent

To use the `setError()` method in a scoped application, use the corresponding scoped method: **Scoped GlideElement - setError(String errorMessage)**.

### GlideElement - setInitialValue(Object value)

Sets the initial value of a field.

This method had been deprecated. Use **GlideElement - setValue(Object value)** for this functionality.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Object</td>
<td>Initial value for the field.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
GlideElement - setJournalEntry(String entry, String userName)

Adds a journal entry and author as a work note or comment field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entry</td>
<td>String</td>
<td>Content of the journal entry.</td>
</tr>
<tr>
<td>userName</td>
<td>String</td>
<td>Optional. The user to attribute the journal entry to. Does not set the journal entry's created by field.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add a work note and its author to a record.

```java
var now_GR = new GlideRecord("incident");

now_GR.addQuery("sys_id", "<sys_id_value>");
now_GR.query();

if(now_GR.next()){
    now_GR.work_notes.setJournalEntry("Content of the journal entry.", "abel.tuter");
    now_GR.update();
}
```

GlideElement - setValue(Object value)

Sets the value of a field.

⚠️ **Note:** Before calling this method, the element must already exist by querying an existing record or by using the `now_GR.initialize()` method to initialize a new record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Object</td>
<td>The value the field is to be set to.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>----------------------</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Set the value passing a string.

```javascript
var glideRecord = new GlideRecord('incident');
glideRecord.query('priority','1');
glideRecord.next();
glideRecord.short_description.setValue('Network failure');
```

Set the value passing an object.

```javascript
var now_GR  = new GlideRecord('student');
now_GR.initialize();
now_GR.setValue('first_name', 'Joe');
now_GR.setValue('last_name', 'Smith');
now_GR.insert();
```

**Scoped equivalent**

To use the `setValue()` method in a scoped application, use the corresponding scoped method: **Scoped GlideElement - setValue(Object value)**.

**GlideElement - toString()**

Converts the field's value to a string.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The field's value as a string.</td>
</tr>
</tbody>
</table>

```javascript
doit();
function doit() {
```
var now_GR = new GlideRecord('sys_user');
now_GR.query();
while (now_GR.next()) {
    if ((now_GR().length != now_GR.first_name.toString().trim().length) ||
        (now_GR.last_name.toString().length != now_GR.last_name.toString().trim().length)) {
        now_GR.first_name = now_GR.first_name.toString().trim();
        now_GR.last_name = now_GR.last_name.toString().trim();
        now_GR.autoSysFields(false);
        now_GR.update();
    }
}

Scoped equivalent

To use the toString() method in a scoped application, use the corresponding scoped method: Scoped GlideElement - toString().

GlideElement - Scoped

The Scoped GlideElement API provides a number of convenient script methods for dealing with fields and their values. Scoped GlideElement methods are available for the fields of the current GlideRecord.

Scoped GlideElement - canCreate()

Determines if the user's role permits the creation of new records in this field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the field can be created, false otherwise.</td>
</tr>
</tbody>
</table>
Scoped GlideElement - canRead()
Indicates whether the user's role permits them to read the associated GlideRecord.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the field can be read, false otherwise.</td>
</tr>
</tbody>
</table>

Scoped GlideElement - canWrite()
Determines whether the user's role permits them to write to the associated GlideRecord.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the user can write to the field, false otherwise.</td>
</tr>
</tbody>
</table>

Scoped GlideElement - changes()
Determines if the current field has been modified. This functionality is available for all available data types, except Journal fields.

Note: The `changes()` method is not supported within ACL scripts.

Note: If the GlideRecord on which you are performing this method has only been initialized and read, and has not been written, the underlying before-and-after values are the same. In this case, the method returns "false", as there has been no change to the data store.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the fields have been changed, false if the field has not.</td>
</tr>
</tbody>
</table>

```java
// This method is often used in business rules. The following example shows is from a business rule,
// if "assigned_to" field value is changed, create a event in the EventQueue.
if (!current.assigned_to.nil() && current.assigned_to.changes()) {
    gs.eventQueue('incident.assigned', current, current.assigned_to.getDisplayValue(),
                  previous.assigned_to.getDisplayValue());
}
```

**Scoped GlideElement - changesFrom(Object o)**

Determines if the previous value of the current field matches the specified object.

**Note:** If the GlideRecord on which you are performing this method has only been initialized and read, and has not been written, the underlying before-and-after values are the same. In this case, the method returns "false", as there has been no change to the data store.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>o</td>
<td>Object</td>
<td>An object value to check against the previous value of the current field.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the previous value matches, false if it does not.</td>
</tr>
</tbody>
</table>
// The following example shows that in a business rule, if "active" field is changed from
// true,
// insert a event in the EventQueue.
if (current.active.changesFrom(true)) {
    gs.eventQueue("incident.inactive", current, current.incident_state,
        previous.incident_state);
}

Scoped GlideElement - changesTo(Object o)

Determines if the new value of a field, after a change, matches the specified
object.

⚠️ Note: The changesTo() method is not supported within ACL scripts.

⚠️ Note: If the GlideRecord on which you are performing this method has only
been initialized and read, and has not been written, the underlying before-
and-after values are the same. In this case, the method returns "false", as
there has been no change to the data store.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>o</td>
<td>Object</td>
<td>An object value to check against the new value of the current field.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the previous value matches, false if it does not.</td>
</tr>
</tbody>
</table>

// The following example shows that in a business rule, if "active" field is changed to
false,
// insert a event in the EventQueue.
if (current.active.changesTo(false)) {
    gs.eventQueue("incident.inactive", current, current.incident_state,
        previous.incident_state);
}
Scoped GlideElement - dateNumericValue()

Returns the number of milliseconds since January 1, 1970, 00:00:00 GMT for a duration field. Does not require the creation of a GlideDateTime object because the duration field is already a GlideDateTime object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of milliseconds since January 1, 1970, 00:00:00 GMT.</td>
</tr>
</tbody>
</table>

```javascript
var inc = new GlideRecord('incident');
inc.get('17c90efb13418700cc36b1422244b05d');
gs.info(inc.calendar_duration.dateNumericValue());
```

Output: 98000

Scoped GlideElement - getAttribute(String attributeName)

Returns the value of the specified attribute from the dictionary.

If the attribute is a boolean attribute, use getBooleanAttribute(String) to get the value as a boolean rather than as a string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributeName</td>
<td>String</td>
<td>Attribute name</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Attribute value</td>
</tr>
</tbody>
</table>

```javascript
doit();
function doit() {
```
```javascript
var now_GR = new GlideRecord('sys_user');
now_GR.query("user_name","admin");
if (now_GR.next()) {
    gs.info("we got one");
    gs.info(now_GR.location.getAttribute("tree_picker"));
}
```
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of possible values for the choice list, which are the values in the Choice [sys_choice] table. If the dependent parameter is passed, the return results reflect only those choices available for the specified dependent field.</td>
</tr>
</tbody>
</table>

```javascript
var glideRecord = new GlideRecord('incident');
glideRecord.query('priority','1');
glideRecord.next();

// urgency has choice list: 1 - High, 2 - Medium, 3 - Low, with value: 1, 2, 3
var choices = glideRecord.urgency.getChoices();
```

**Scoped GlideElement - getChoiceValue()**

Returns the choice label for the current choice.

A choice has a value (number) and a label (string). This method returns the label.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The selected choice's label.</td>
</tr>
</tbody>
</table>

```javascript
var glideRecord = new GlideRecord('incident');
glideRecord.query('priority','1');
glideRecord.next();

// urgency has choice list: 1 - High, 2 - Medium, 3 - Low, with value: 1, 2, 3
var choiceLabel = glideRecord.urgency.getChoiceValue();
gs.info(choiceLabel);
```
**Scoped GlideElement - getDecryptedValue()**

Returns the clear text value for Password (2 way encrypted) fields in scoped applications.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The clear text password.</td>
</tr>
</tbody>
</table>

```javascript
var tablename = 'xScopedApp_table';
var CI = new GlideRecord(tablename);
CI.addQuery('number', '0001002');
CI.query();
CI.next();

var password = CI.password_field
var decrypted = password.getDecryptedValue();
gs.info(decrypted);
```

**Output:**

`xScopedApp: cleartextpassword`

**Scoped GlideElement - getDisplayValue( Number maxCharacters)**

Gets the formatted display value of the field.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxCharacters</td>
<td>Number</td>
<td>Optional: Maximum characters desired</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The display value of the field</td>
</tr>
</tbody>
</table>

```javascript
var glideRecord = new GlideRecord('incident');
glideRecord.query('priority','1');
glideRecord.next();
gs.info(glideRecord.priority.getDisplayValue());
```

**Scoped GlideElement - getED()**

Returns the field's element descriptor.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scoped GlideElementDescriptor</td>
<td>The field's element descriptor.</td>
</tr>
</tbody>
</table>

```javascript
var grInc = new GlideRecord('incident');
grInc.query('priority', '1');

var field = grInc.getElement('priority');
var ed = field.getED();
```

**Scoped GlideElement - getGlobalDisplayValue()**

Returns the phone number in international format.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>--------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>The phone number in international format.</td>
<td></td>
</tr>
</tbody>
</table>

**Scoped GlideElement - getHTMLValue(Number maxChars)**

Returns the HTML value of a field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxChars</td>
<td>Number</td>
<td>Optional. Maximum number of characters to return.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>HTML value for the field.</td>
</tr>
</tbody>
</table>

```javascript
var inccause = new GlideRecord("incident");
inccause.short_description = current.short_description;
inccause.comments = current.comments.getHTMLValue();
inccause.insert();
```

**Scoped GlideElement - getJournalEntry(Number mostRecent)**

Returns either the most recent journal entry or all journal entries.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mostRecent</td>
<td>Number</td>
<td>If 1, returns the most recent entry. If -1, returns all journal entries.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>For the most recent entry, returns a string that contains the field label, timestamp, and user display name of the journal entry.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>For all journal entries, returns the same information for all journal entries ever entered as a single string with each entry delimited by &quot;\n\n&quot;.</td>
</tr>
</tbody>
</table>

```javascript
//gets all journal entries as a string where each entry is delimited by '\n\n'
var notes = current.work_notes.getJournalEntry(-1);
//stores each entry into an array of strings
var na = notes.split("\n\n");

for (var i = 0; i < na.length; i++)
  gs.info(na[i]);
```

 Scoped GlideElement - getLabel()

Returns the object label.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Object label</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord("sc_req_item");
now_GR.addQuery("request", current.sysapproval);
now_GR.query();
while(now_GR.next()) {
  var nicePrice = now_GR.price.toString();
  if (nicePrice !== ) {
    nicePrice = parseFloat(nicePrice);
    nicePrice = nicePrice.toFixed(2);
  }
  template.print(now_GR.number + ":  " + now_GR.quantity + " X " +
    now_GR.cat_item.getDisplayValue() + " at $" + nicePrice + " each \n\n");
  template.print("    Options:\n")
```
var variables = now_GR.variables.getElements();
for (var key in variables) {
    var now_V = variables[key];
    if (now_V.getQuestion().getLabel() != ) {
        template.space(4);
        template.print('     ' + now_V.getQuestion().getLabel() + " = " + now_V.getDisplayValue() + "\n");
    }
}

**Scoped GlideElement - getName()**

Returns the name of the field.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Scoped GlideElement - getReferenceTable()**

Gets the table name for a reference element.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>The table name of the reference</td>
</tr>
</tbody>
</table>
var grINC = new GlideRecord('incident');
grINC.query('number','INC0010041'); // record assignment group assigned to "CAB Approval"
if (grINC.next()) {
    // Get the table name
    var tableName = grINC.assignment_group.getReferenceTable();
    gs.info(tableName);
}

Scoped GlideElement - getRefRecord()
Returns a GlideRecord object for a given reference element.

⚠️ Warning: If the reference element does not contain a value, it returns an empty GlideRecord object, not a NULL object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>A GlideRecord object</td>
</tr>
</tbody>
</table>

var grINC = new GlideRecord('incident');
grINC.addNotNullQuery('caller_id');
grINC.query();
if (grINC.next()) {
    // Get a GlideRecord object for the referenced sys_user record
    var grUSER = grINC.caller_id.getRefRecord();
    if (grUSER.isValidRecord())
        gs.info(grUSER.getValue('name'));
}

Scoped GlideElement - getTableName()
Returns the name of the table on which the field resides.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the table. The returned value may be different from the table Class that the record is in. See Tables and Classes in the product documentation.</td>
</tr>
</tbody>
</table>

```javascript
if (current.approver.getTableName() == "sysapproval_approver") {
    if (current.approver == email.from_sys_id) {
        current.comments = "reply from: " + email.from + "\n\n" + email.body_text;
        // if it's been cancelled, it's cancelled.
        var doit = true;
        if (current.state=='cancelled')
            doit = false;
    
    if (email.body.state != undefined)
        current.state= email.body.state;
    if (doit)
        current.update();
    } else {
        gs.log("Approval for task ("+current.sysapproval.getDisplayValue()+") rejected because user sending email ("+email.from+") does not match the approver ("+current.approver.getDisplayValue()+");
    }
}
```

Scoped GlideElement - nil()

Determines if a field is null.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the field is null or an empty string, false if not.</td>
</tr>
</tbody>
</table>

```javascript
var glideRecord = new GlideRecord('incident');
glideRecord.query('priority','1');
glideRecord.next();
gs.info(glideRecord.state.nil());
```

### Scoped GlideElement - `setDateNumericValue(Number milliseconds)`

Sets the value of a date/time element to the specified number of milliseconds since January 1, 1970 00:00:00 GMT.

When called, `setDateNumericValue()` automatically creates the necessary GlideDateTime/GlideDate/GlideDuration object, and then sets the element to the specified value.

**Note:** Before calling this method, the element must already exist by querying an existing record or by using the `now_GR.initialize()` method to initialize a new record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>milliseconds</td>
<td>Number</td>
<td>Number of milliseconds since 1/1/1970</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord("incident");
now_GR.initialize();
now_GR.opened_at.setDateNumericValue(10000);
```
Scoped GlideElement - setDisplayValue(Object value)
Sets the display value of the field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Object</td>
<td>The value to set for the field.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var glideRecord = new GlideRecord('incident');
glideRecord.query('priority','1');
glideRecord.next();

//change the urgency to 3
glideRecord.urgency.setDisplayValue('3 - Low');
gs.info(glideRecord.urgency);
```

Scoped GlideElement - setError(String errorMessage)
Adds an error message. Available in Fuji patch 3.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errorMessage</td>
<td>String</td>
<td>The error message.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var glideRecord = new GlideRecord('incident');
glideRecord.query('priority','1');
glideRecord.next();
```
**Scoped GlideElement - setPhoneNumber(Object phoneNumber, Boolean strict)**

Sets the field to the specified phone number.

This method is only available on a phone number GlideElement.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phoneNumber</td>
<td>Object</td>
<td>The phone number to set. This can be in either the international or local format.</td>
</tr>
<tr>
<td>strict</td>
<td>Boolean</td>
<td>When true, specifies that the number specified must match the correct format. When false, the system attempts to correct an improperly formatted phone number.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the value was set.</td>
</tr>
</tbody>
</table>

**Scoped GlideElement - setValue(Object value)**

Sets the value of a field.

⚠️ **Note:** Before calling this method, the element must already exist by querying an existing record or by using the `now_GR.initialize()` method to initialize a new record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Object</td>
<td>Object value to set the field to.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Set the value passing a string.

```javascript
var glideRecord = new GlideRecord('incident');
glideRecord.query('priority','1');
glideRecord.next();
glideRecord.short_description.setValue('Network failure');
```

Set the value passing an object.

```javascript
var now_GR  = new GlideRecord('student');
now_GR.initialize();
now_GR.setValue('first_name', 'Joe');
now_GR.setValue('last_name', 'Smith');
now_GR.insert();
```

**Scoped GlideElement - toString()**

Converts the value of a GlideRecord field to a string.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value as a string.</td>
</tr>
</tbody>
</table>

```javascript
var glideRecord = new GlideRecord('incident');
glideRecord.query('priority','1');
glideRecord.next();
gs.info(glideRecord.opened_at.toString());
```

**Output**

```
2019-08-31 23:09:51
```
GlideElementCurrency2 - Scoped

The GlideElementCurrency2 API provides methods to perform operations on Foreign Exchange (FX) Currency fields (also known as Currency2) within the current GlideRecord.

This API provides methods that enable you to perform the following:

• Obtain the FX Currency field display string, as entered by the user.
• Obtain the display value of an FX Currency field.
• Obtain the reference currency value of an FX Currency field.
• Obtain the session currency value of an FX Currency field.
• Set the display value of an FX Currency field.

The GlideElementCurrency2 class has no constructor.

Scoped GlideElementCurrency2 - getAsEnteredDisplayValue()

Returns an FX Currency field display string, as entered by the user, from the associated GlideRecord.

For additional information on FX Currency fields, see Setting up and operating FX Currency fields.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Display currency string as entered by the user.</td>
</tr>
</tbody>
</table>

Example

```javascript
var tablename = 'my_currency2_table';
var setcur2 = new GlideRecord(tablename);
setcur2.initialize();
setcur2.currency_2_field.setDisplayValue('JPY;999999.1234');
setcur2.insert();
gs.info('As entered display value: ' + setcur2.currency_2_field.getAsEnteredDisplayValue());
```
Output:

As entered display value: ¥999,999.12

Scoped GlideElementCurrency2 - getDisplayValue()

Returns the display value of an FX Currency field within the associated GlideRecord.

Depending on how the display_value is set in the FX Currency Configuration [fx_configuration] table, the returned value may be the:

- Currency value as entered by the user
- Currency value converted to the session currency (based on the user's locale)
- Currency value converted to the reference currency

For additional information on FX Currency and the configuration table, see Configure FX Currency global settings.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Display currency value</td>
</tr>
</tbody>
</table>

Example

```javascript
var tablename = 'my_FXcurrency_table';
var setcur2 = new GlideRecord(tablename);
setcur2.initialize();
setcur2.currency_2_field.setDisplayValue('JPY;999999.1234');
setcur2.insert();
gs.info('Display Value: ' + setcur2.currency_2_field.getDisplayValue());
```

Output:

Display Value: ¥999,999.12
Scoped GlideElementCurrency2 - getReferenceDisplayValue()

Returns the reference currency value of an FX Currency field within the associated GlideRecord.

Reference currency is the common currency into which all currency values in a field are converted. The currency code that the `getReferenceDisplayValue()` method uses to derive the reference currency is based on the following:

- If an instance record exists, then the method uses the value in the reference_currency field of the FX Currency [fx_currency2_instance] table.
- If an instance record does not exist, then the method uses the reference currency that would be set on the instance record when the instance record is eventually created (in the following order):
  - If set, the `reference_currency` in the FX Currency Configuration [fx_configuration] table.
  - If set, the `reference_currency_source` in the FX Currency Configuration [fx_configuration] table.
  - System default, system reference currency.

For additional information on the values in the FX Currency Configuration [fx_configuration] table, see Setting up and operating FX Currency fields.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Reference currency amount for the associated FX Currency display value.</td>
</tr>
</tbody>
</table>

This example assumes the reference currency code is USD and uses the default conversion rate specified in the FX Currency Configuration [fx_configuration] table. The output will change based on locale.

```javascript
var tablename = 'my_currency2_table';
var setcur2 = new GlideRecord(tablename);
setcur2.initialize();
setcur2.currency_2_field.setDisplayValue('JPY;999999.1234');
```
Scoped GlideElementCurrency2 - getSessionDisplayValue()

Returns the session currency value of the associated FX Currency field within the current GlideRecord.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Currency value converted to the session currency, which is based on the current user's locale.</td>
</tr>
</tbody>
</table>

This example assumes the session locale is language code = eng and country code = USA, and uses the default conversion rate specified in the FX Currency Configuration [fx_configuration] table. The output will change based on locale.

```javascript
var tablename = 'my_currency2_table';
var setcur2 = new GlideRecord(tablename);
setcur2.initialize();
setcur2.currency_2_field.setDisplayValue('JPY;999999.1234');
setcur2.insert();
gs.info('Session display value: ' + setcur2.currency_2_field.getSessionDisplayValue());
```

Output:

Session display value: $8,677.99

Scoped GlideElementCurrency2 - setDisplayValue(String displayValue)

Sets the display value of an FX Currency field with the specified currency value.

For additional information on FX Currency fields, see Setting up and operating FX Currency fields.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>displayValue</td>
<td>String</td>
<td>Value to set in the FX Currency field in the format <code>&lt;currency code&gt;;&lt;currency amount&gt;</code>. You must format the <code>&lt;currency amount&gt;</code> field in the user's locale. For example, if the user's locale is USA/eng, the passed in <code>&lt;currency amount&gt;</code> would be 123.45. If the user's local is FRA/fre, the passed in <code>&lt;currency amount&gt;</code> would be 123,45.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
var tablename = 'my_FXcurrency_table';
var setcur2 = new GlideRecord(tablename);
setcur2.initialize();
setcur2.currency_2_field.setDisplayValue('JPY;999,999.1234');
setcur2.insert();
```

**GlideElementDescriptor - Scoped, Global**

The GlideElementDescriptor API provides information about individual fields in a record.

There is no constructor for this class. Use the GlideElement or GlideRecord getED() method to obtain a GlideElementDescriptor object.

**Scoped GlideElementDescriptor - getAttachmentEncryptionType()**

Returns the encryption type used for attachments on the element's table.

This method is for use with the Edge Encryption plugin.

Parameters
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The encryption type used on attachments. Returns null if attachments on the element's table are not being encrypted.</td>
</tr>
</tbody>
</table>

```javascript
var grInc = new GlideRecord('incident');
grInc.query('priority', '1');

var field = grInc.getElementById('priority');
var ed = field.getED();

var isEdge = ed.getAttachmentEncryptionType();
gs.info(isEdge);
```

Output: null

**Scoped GlideElementDescriptor - getEncryptionType()**

Returns the element's encryption type.

This method is for use with the Edge Encryption plugin.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The element's encryption type. Returns null if the element is not encrypted.</td>
</tr>
</tbody>
</table>

```javascript
var grInc = new GlideRecord('incident');
grInc.query('priority', '1');

var field = grInc.getElementById('priority');
var ed = field.getED();
```
sEdge = ed.getEncryptionType();
gs.info(isEdge);

Output: null

Scoped GlideElementDescriptor - getInternalType()
Returns the element's internal data type.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String    | The element's internal data type. Possible values:  
|           | • boolean  
|           | • char     
|           | • collection  
|           | • conditions  
|           | • date     
|           | • decimal  
|           | • documentation_field  
|           | • domain_id  
|           | • due_date  
|           | • email    
|           | • field_name  
|           | • file_attachment  
|           | • float    
|           | • glide_date  
|           | • glide_date_time  
|           | • glide_duration  
<p>|           | • glide_list  |</p>
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GUID</td>
<td></td>
</tr>
<tr>
<td>html</td>
<td></td>
</tr>
<tr>
<td>image</td>
<td></td>
</tr>
<tr>
<td>integer</td>
<td></td>
</tr>
<tr>
<td>long</td>
<td></td>
</tr>
<tr>
<td>longint</td>
<td></td>
</tr>
<tr>
<td>multi_two_lines</td>
<td></td>
</tr>
<tr>
<td>journal</td>
<td></td>
</tr>
<tr>
<td>journal_input</td>
<td></td>
</tr>
<tr>
<td>numeric</td>
<td></td>
</tr>
<tr>
<td>order_index</td>
<td></td>
</tr>
<tr>
<td>password</td>
<td></td>
</tr>
<tr>
<td>ph_number</td>
<td></td>
</tr>
<tr>
<td>reference</td>
<td></td>
</tr>
<tr>
<td>script</td>
<td></td>
</tr>
<tr>
<td>script_plain</td>
<td></td>
</tr>
<tr>
<td>string</td>
<td></td>
</tr>
<tr>
<td>sys_class_name</td>
<td></td>
</tr>
<tr>
<td>table_name</td>
<td></td>
</tr>
<tr>
<td>template_value</td>
<td></td>
</tr>
<tr>
<td>timer</td>
<td></td>
</tr>
<tr>
<td>translated_field</td>
<td></td>
</tr>
<tr>
<td>url</td>
<td></td>
</tr>
<tr>
<td>user_image</td>
<td></td>
</tr>
<tr>
<td>user_input</td>
<td></td>
</tr>
<tr>
<td>user_roles</td>
<td></td>
</tr>
<tr>
<td>video</td>
<td></td>
</tr>
<tr>
<td>workflow</td>
<td></td>
</tr>
</tbody>
</table>
var grInc = new GlideRecord('incident');
grInc.query('priority', '1');

var field = grInc.getElement('priority');
var ed = field.getED();

var isEdge = ed.getInternalType();
gs.info(isEdge);

Output:

integer

Scoped GlideElementDescriptor - getLabel()

Returns the element's label.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The element's label.</td>
</tr>
</tbody>
</table>

var grInc = new GlideRecord('incident');
grInc.query('priority', '1');

var field = grInc.getElement('priority');
var ed = field.getED();

var isEdge = ed.getLabel();
gs.info(isEdge);

Output: Priority

Scoped GlideElementDescriptor - getLength()

Returns the element's length.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The element's size.</td>
</tr>
</tbody>
</table>

```javascript
var grInc = new GlideRecord('incident');
grInc.query('priority', '1');

var field = grInc.getElement('priority');
var ed = field.getED();

var isEdge = ed.getLength();
gs.info(isEdge);
```

Output: 40

Scoped GlideElementDescriptor - getName()

Returns the element's name.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The element's name.</td>
</tr>
</tbody>
</table>

```javascript
var grInc = new GlideRecord('incident');
grInc.query('priority', '1');

var field = grInc.getElement('priority');
var ed = field.getED();
```
var isEdge = ed.getName();
gs.info(isEdge);

Output: priority

**Scoped GlideElementDescriptor - getPlural()**

Returns the element's plural label.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The element's plural label.</td>
</tr>
</tbody>
</table>

var now_GR = new GlideRecord('incident');
now_GR.query();
var ed = now_GR.getED();
gs.info(ed.getPlural());

Output: Incidents

**Scoped GlideElementDescriptor - hasAttachmentsEncrypted()**

Returns true if an encrypted attachment has been added to the table.

This method is for use with the Edge Encryption plugin.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if an encrypted attachment has been added to the table.</td>
</tr>
</tbody>
</table>

```javascript
var grInc = new GlideRecord('incident');
grInc.query('priority', '1');

var field = grInc.getElement('priority');
var ed = field.getED();

var isEdge = ed.hasAttachmentsEncrypted();
gs.info(isEdge);
```

Output: false

**Scoped GlideElementDescriptor - isAutoOrSysID()**

Returns true if the element is an automatically generated or system field.

Automatically generated and system fields cannot be encrypted. This method is for use with the Edge Encryption plugin.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the element is automatically generated or a system field.</td>
</tr>
</tbody>
</table>

```javascript
var grInc = new GlideRecord('incident');
grInc.query('priority', '1');
var field = grInc.getElement('priority');
var ed = field.getED();

isEdge = ed.isAutoOrSysID();
gs.info(isEdge);
```
Output: false

Scoped GlideElementDescriptor - isChoiceTable()

Returns true if the element is defined as a dropdown choice in its dictionary definition.

Choice fields cannot be encrypted.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the element is defined as a dropdown choice. Returns true even if there are no entries defined in the choice table. The last choice type, suggestion, does not return true.</td>
</tr>
</tbody>
</table>

```javascript
var grInc = new GlideRecord('incident');
grInc.query('priority', '1');
var field = grInc.getElementById('priority');
var ed = field.getED();
var isChoiceTable = ed.isChoiceTable();
gs.info(isChoiceTable);
```

Output: true

Scoped GlideElementDescriptor - isEdgeEncrypted()

Returns true if an element is encrypted.

This method is for use with the Edge Encryption plugin.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the element is encrypted, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var grInc = new GlideRecord('incident');
grInc.query('priority', '1');

var field = grInc.getElement('priority');
var ed = field.getED();

var isEdge = ed.isEdgeEncrypted();
gs.info(isEdge)
```

Output: false

**Scoped GlideElementDescriptor - isMandatory()**

Determines whether the element is mandatory and must contain a value before the record can be saved.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean   | Flag that indicates whether the associated element is mandatory and must contain a value before the record containing the element can be saved. Valid values:  
• true: element must contain a value; mandatory field  
• false: element does not have to contain a value |

This example shows how to check whether the name field is mandatory.

```javascript
var isRecordValid;
var now_GR = new GlideRecord('my_table');
```
```javascript
var field = now_GR.getElement('name');
var elementDescriptor = field.getED();
now_GR.query();
while (now_GR.next()) {
  if (elementDescriptor.isMandatory() && !now_GR.name) {
    isRecordValid = false;
  }
}
```

### Scoped GlideElementDescriptor - isVirtual()

Returns true if the element is a virtual element.

A virtual element is a calculated field as set by the dictionary definition of the field. Virtual fields cannot be encrypted.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the element is a virtual element.</td>
</tr>
</tbody>
</table>

```javascript
var grInc = new GlideRecord('incident');
grInc.query('priority', '1');

var field = grInc.getElement('priority');
var ed = field.getED();

var isVirtual = ed.isVirtual();
gs.info(isVirtual);
```

Output: false

### GlideEmailOutbound - Scoped

The scoped GlideEmailOutbound class implements the email object for scoped applications. You can use the GlideEmailOutbound methods with the email global object available in mail scripts. The email object behaves identically for global and scoped applications.
Scoped GlideEmailOutbound - addAddress(String type, String address)

Adds the address to either the cc or bcc list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>Either cc or bcc, determines the list to which the address is added.</td>
</tr>
<tr>
<td>address</td>
<td>String</td>
<td>The recipient's email address.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

email.addAddress('cc', 'joe.employee@something.com');

Scoped GlideEmailOutbound - addAddress(String type, String address, String displayName)

Adds the recipient to either the cc or bcc list, but uses the display name instead of the address when showing the recipient.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>Either cc or bcc, determines the list to which the address is added.</td>
</tr>
<tr>
<td>address</td>
<td>String</td>
<td>The recipient's email address.</td>
</tr>
<tr>
<td>displayName</td>
<td>String</td>
<td>The name to be shown instead of the email address.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
Scoped GlideEmailOutbound - getSubject()

Returns the email's subject line.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The email's subject line.</td>
</tr>
</tbody>
</table>

Scoped GlideEmailOutbound - getWatermark()

Returns the email's watermark.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The email's watermark.</td>
</tr>
</tbody>
</table>

var watermark = email.getWatermark();

Scoped GlideEmailOutbound - GlideEmailOutbound()

Instantiates a scoped GlideEmailOutbound object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Scoped GlideEmailOutbound - setBody(String bodyText)
Sets the body of the email.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bodyText</td>
<td>String</td>
<td>The body of the email.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

email.setBody('Dear Sir, ...');

Scoped GlideEmailOutbound - setFrom(String address)
Sets the sender’s address.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>String</td>
<td>The sender’s email address.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

eemail.setFrom('joe.employee@something.com');
Scoped GlideEmailOutbound - setReplyTo(String address)

Sets the reply to address.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>String</td>
<td>The reply to email address.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
eemail.setReplyTo('joe.employee@something.com');
```

Scoped GlideEmailOutbound - setSubject(String subject)

Sets the email's subject line.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject</td>
<td>String</td>
<td>Text for the subject line.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
eemail.setSubject('Important Issues to discuss');
```

GlideEncrypter - Global

GlideEncrypter provides methods to encrypt and decrypt strings using the Triple DES algorithm.

The GlideEncrypter class is used in server scripts in the global scope. The GlideEncrypter class has two constructors:
• GlideEncrypter()
• GlideEncrypter(String key)

GlideEncrypter - decrypt(String encryptedString)
Decrypts a clear string using the Triple DES algorithm.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>encryptedString</td>
<td>String</td>
<td>String to be decrypted.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Clear text string.</td>
</tr>
</tbody>
</table>

```javascript
var encr = new GlideEncrypter();
var clearString = 'abcdefg';
var encrString = encr.encrypt(clearString);
var decrString = encr.decrypt(encrString);
gs.print("Decrypted string = " + decrString);
```

Output:
Decrypted string = abcdefg

GlideEncrypter - encrypt(String clearString)
Encrypts a clear string using the Triple DES algorithm.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clearString</td>
<td>String</td>
<td>String to be encrypted.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Encrypted string.</td>
</tr>
</tbody>
</table>
```javascript
var encr = new GlideEncrypter();
var clearString = 'abcdefg';
var encrString = encr.encrypt(clearString);
gs.print("Encrypted string = " + encrString);

Output:

Encrypted string = 3wjpvKtUIi4=
```

**GlideEncrypter - GlideEncrypter()**

Creates an instance of the GlideEncrypter class using a default (static) encryption key.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var encr = new GlideEncrypter();
```

**GlideEncrypter - GlideEncrypter(String key)**

Creates an instance of the GlideEncrypter class using a given encryption key.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Your encryption key must be exactly 24 characters. A key longer than 24 characters will be truncated.</td>
</tr>
</tbody>
</table>

```javascript
var encr = new GlideEncrypter(myKey);
```

**GlideScopedEvaluator - Global**

The GlideScopedEvaluator API allows you to evaluate scripts in a GlideRecord field from both scoped and global server scripts.

The GlideScopedEvaluator API evaluates scripts within the script field type. The scope of the record defines the scope of the script.
GlideScopedEvaluator - evaluateScript(GlideRecord grObj, String scriptField, Object variables)
Evaluates a script from a GlideRecord field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>grObj</td>
<td>GlideRecord</td>
<td>The GlideRecord containing a script expression.</td>
</tr>
<tr>
<td>scriptField</td>
<td>String</td>
<td>(Optional) The name of the field containing the script expression.</td>
</tr>
<tr>
<td>variables</td>
<td>Object</td>
<td>(Optional) A map of variables with name-value pairs. These variables are available to the script during execution of this method.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The result of the script execution.</td>
</tr>
</tbody>
</table>

```
//For this example, we created a table: "x_app_table" with two columns:
//"short_description", "test_script"
//"test_script" will store the script to be evaluated by GlideScopedEvaluator.
var now_GR = new GlideRecord('x_app_table');
now_GR.short_description = 'Testing GlideScopedEvaluator';
now_GR.test_script = "gs.getUser().getName() + ' says ' + greeting; ";
now_GR.insert();

//setup variables to be used by the script
var vars = {'greeting' : 'hello'};

//Evaluate the script from the field
var evaluator = new GlideScopedEvaluator();
gs.info(evaluator.evaluateScript(now_GR, 'test_script', vars));

// Now retrieve the result
evaluator.evaluateScript(gr, 'u_test_script', null);
gs.info(evaluator.getVariable('result'));
```

Output: admin says hello
Scoped equivalent

To use the `evaluateScript()` method in a scoped application, use the corresponding scoped method: `GlideScopedEvaluator - evaluateScript(GlideRecord grObj, String scriptField, Object variables)`.

**GlideScopedEvaluator - getVariable(String name)**

Returns a variable from a GlideScopedEvaluator object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the variable.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The value of the specified variable.</td>
</tr>
</tbody>
</table>

```javascript
//setting up a record that contains the script to be executed.
var now_GR = new GlideRecord('u_global_table');
now_GR.u_short_description = 'Calculate Addition';
now_GR.u_test_script = "result = x + y";
now_GR.insert();

var evaluator = new GlideScopedEvaluator();
evaluator.putVariable('x', 100);
evaluator.putVariable('y', 200);
evaluator.putVariable('result', null);

Output: 300
```

Scoped equivalent

To use the `getVariable()` method in a scoped application, use the corresponding scoped method: `GlideScopedEvaluator - getVariable(String name)`.

**GlideScopedEvaluator - GlideScopedEvaluator()**

Instantiates a GlideScopedEvaluator object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Scoped equivalent

To use the GlideScopedEvaluator() method in a scoped application, use the corresponding scoped method: GlideScopedEvaluator - GlideScopedEvaluator().

GlideScopedEvaluator - putVariable(String name, Object value)

Puts a variable into the GlideScopedEvaluator object. These variables are available to the script that this GlideScopedEvaluator object runs.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the variable.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>The value of the variable.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
//setting up a record that contains the script to be executed.
var now_GR = new GlideRecord('u_global_table');
now_GR.u_short_description = 'Calculate Addition';
now_GR.u_test_script = "result = x + y";
now_GR.insert();

var evaluator = new GlideScopedEvaluator();
evaluator.putVariable('x', 100);
evaluator.putVariable('y', 200);
evaluator.putVariable('result', null);

Output: 300
```
Scoped equivalent

To use the `putVariable()` method in a scoped application, use the corresponding scoped method: `GlideScopedEvaluator - putVariable(String name, Object value).

**GlideScopedEvaluator - Scoped**

The Glide ScopedEvaluator API allows you to evaluate scripts in a GlideRecord field from both scoped and global server scripts.

The GlideScopedEvaluator API evaluates records with script fields defined. The scope of the script is defined by the scope of the record.

**GlideScopedEvaluator - evaluateScript(GlideRecord grObj, String scriptField, Object variables)**

Evaluates a script from a GlideRecord field.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>grObj</td>
<td>GlideRecord</td>
<td>The GlideRecord containing a script expression.</td>
</tr>
<tr>
<td>scriptField</td>
<td>String</td>
<td>(Optional) The name of the field containing the script expression.</td>
</tr>
<tr>
<td>variables</td>
<td>Object</td>
<td>(Optional) A map of variables with name-value pairs. These variables are available to the script during execution of this method.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The result of the script execution.</td>
</tr>
</tbody>
</table>

```java
// For this example, we created a table: "x_app_table" with two columns:
"short_description", "test_script"
// "test_script" will store the script to be evaluated by GlideScopedEvaluator.
var now_GR = new GlideRecord('x_app_table');
now_GR.short_description = 'Testing GlideScopedEvaluator';
now_GR.test_script = "gs.getUser().getName() + ' says ' + greeting; ";
now_GR.insert();

// setup variables to be used by the script
```
var vars = {'greeting' : 'hello'};

//Evaluate the script from the field
var evaluator = new GlideScopedEvaluator();
gs.info(evaluator.evaluateScript(now_GR, 'test_script', vars));

// Now retrieve the result
evaluator.evaluateScript(gr, 'u_test_script', null);
gs.info(evaluator.getVariable('result'));

Output: admin says hello

**GlideScopedEvaluator - getVariable(String name)**

Returns a variable from a GlideScopedEvaluator object.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Object</td>
</tr>
</tbody>
</table>

//setting up a record that contains the script to be executed.
var now_GR = new GlideRecord('u_global_table');
now_GR.u_short_description = 'Calculate Addition';
now_GR.u_test_script = "result = x + y";
now_GR.insert();

var evaluator = new GlideScopedEvaluator();
evaluator.putVariable('x', 100);
evaluator.putVariable('y', 200);
evaluator.putVariable('result', null);

**GlideScopedEvaluator - GlideScopedEvaluator()**

Instantiates a GlideScopedEvaluator object.
### GlideScopedEvaluator - putVariable(String name, Object value)

Puts a variable into the GlideScopedEvaluator object. These variables are available to the script that this GlideScopedEvaluator object runs.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the variable.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>The value of the variable.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
//setting up a record that contains the script to be executed.
var now_GR = new GlideRecord('u_global_table');
now_GR.u_short_description = 'Calculate Addition';
now_GR.u_test_script = "result = x + y";
now_GR.insert();

var evaluator = new GlideScopedEvaluator();
evaluator.putVariable('x', 100);
evaluator.putVariable('y', 200);
evaluator.putVariable('result', null);
```

Output: 300

### GlideExcelParser - Scoped, Global

Parse .xlsx formatted Excel files and access file data in script.

The GlideExcelParser methods can be used in global and scoped scripts. Use the `sn_impex` namespace identifier to create a GlideExcelParser object.
GlideExcelParser - close()
Close the connection to the input stream and release the document.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideExcelParser - getColumnHeaders()
Returns a list of column headers from the parsed document.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of strings of column headers from the parsed document.</td>
</tr>
</tbody>
</table>

GlideExcelParser - getErrorMessage()
Returns the error message when the parse() method fails.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The error message.</td>
</tr>
</tbody>
</table>

GlideExcelParser - `getRow()`

Get the current row values and headers.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The row headers are property names and the row values are property values.</td>
</tr>
</tbody>
</table>

GlideExcelParser - `getSheetNames()`

Gets a list of all worksheet names in an Excel workbook.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array of strings as worksheet names ordered as positioned from left to right in the workbook. Results include any worksheets marked as hidden.</td>
</tr>
</tbody>
</table>

The following example shows how to parse an Excel file attachment multiple times to retrieve column headers and print values of each row for each worksheet. Results display column headers and values of three separate sheets in the Excel attachment.
var parser = new sn_impex.GlideExcelParser();

var attachment = new GlideSysAttachment();

// Use the attachment sys_id of an Excel file
var attachmentStream = attachment.getContentStream("<attachment_sys_id>");

// Set the source to be parsed
parser.setSource(attachmentStream);

// Get the worksheet names to be parsed in the Excel document
var list_sheet_name = parser.getSheetNames();

gs.info(" Sheet Names " + list_sheet_name.join(" ", ");

// Iterate over each worksheet in the Excel workbook
for (var i = 0; i < list_sheet_name.length; i++) {

    gs.info("*********************************************************************************
    *****
    Sheet name: " + list_sheet_name[i]);

    // Set the worksheet name to be parsed
    parser.setSheetName(list_sheet_name[i]);

    // Parse each worksheet set using setSheetName()
    if (parser.parse()) {
        //retrieve the column headers
        var headers = parser.getColumnHeaders();
        var header1 = headers[0];
        var header2 = headers[1];
        var header3 = headers[2];

        //print headers
        gs.info(header1 + "\t" || header2 + "\t" || header3);

        // Iterate over each row in the worksheet
        while (parser.next()) {
            var row = parser.getRow();
            //print row value for both columns
            gs.info(row[header1] + "\t" || row[header2] + "\t" || row[header3])
        }
    }
}

else
gs.info(parser.getErrorMessage());

Output:

*** Script:  Sheet Names Incident Table, sys_user Table, knowledge Table
*** Script:
**********************************************************************************
*** Script: Sheet name:    Incident Table
*** Script: Number ||Opened ||Short description
*** Script: INC00010111 || 2019-07-22 14:04:57 ||ATF : Test1
*** Script: INC0009009 || 2018-08-30 01:06:16 ||Unable to access the shared folder.
*** Script:
**********************************************************************************
*** Script: Sheet name:    sys_user Table
*** Script: User ID ||Name ||Email
*** Script: abel.tuter || Abel Tuter ||abel.tuter@example.com
*** Script: abraham.lincoln || Abraham Lincoln ||abraham.lincoln@example.com
*** Script: adela.cervantsz || Adela Cervantsz ||adela.cervantsz@example.com
*** Script: aileen.mottern || Aileen Mottern ||aileen.mottern@example.com
*** Script: alejandra.prenatt || Alejandra Prenatt ||alejandra.prenatt@example.com
*** Script:
**********************************************************************************
*** Script: Sheet name:    knowledge Table
*** Script: Number ||Short description ||Author
*** Script: KB99999999 || Microsoft Outlook Issues ||System Administrator
*** Script: KB0000033 || Eclipse configuration for Android development ||System Administrator
*** Script: KB0000032 || Getting Around in Windows ||System Administrator
*** Script: KB0000031 || How can I find the MAC address of my Ethernet or wireless interface? ||Sam Sorokin

GlideExcelParser - getTableInfo()

Gets table column types and max character length from a spreadsheet or CSV attachment.

⚠️ Note: This method is restricted to scoped applications.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Returns attachment data as a map in which the key is column name. Values contain column type and maximum length of a character string in the column. Access return values using the <code>getType()</code> and <code>getLength()</code> methods.</td>
</tr>
</tbody>
</table>

This example shows how to data from a 7-column spreadsheet attachment including row data such as 1, Meghan, 1/1/85, 1:00 AM, 10.00%, 224-432-8582, ½.

```javascript
var attachmentSysId = "<sys_id>";
var headerRowNumber = 6;

var attachmentStream = new GlideSysAttachment().getContentStream(attachmentSysId);
var parser = new sn_impex.GlideExcelParser();
parser.setHeaderRowNumber(headerRowNumber);
if (!parser.parse(attachmentStream)) {
    throw "Attachment could not be parsed as an Excel Spreadsheet " + attachmentSysId;
}
var tableInfo = parser.getTableInfo();
for (var name in tableInfo) {
    gs.info('Column Name : ' + name + ', Type : ' + tableInfo[name].getType() + ', Max Length : ' + tableInfo[name].getLength());
}

Output:

ProgressWorker: Attempted to get non-existent sys_progress_worker record, inserted 81b0ca5ddbb59730090c9ff9dbf9619c4
sn_appcreator: Column Name : My Number Col, Type : integer, Max Length : 1
sn_appcreator: Column Name : My String Col, Type : string, Max Length : 650
sn_appcreator: Column Name : My Date Col, Type : glide_date, Max Length : 6
sn_appcreator: Column Name : My DateTime Col, Type : glide_date_time, Max Length : 7
sn_appcreator: Column Name : My Percent Col, Type : decimal, Max Length : 6
sn_appcreator: Column Name : My Phone Col, Type : string, Max Length : 12
sn_appcreator: Column Name : My Fraction Col, Type : decimal, Max Length : 3
```
GlideExcelParser - GlideExcelParser()

Creates an instance of GlideExcelParser.

The API name space identifier "sn_impex" must be used when creating a GlideExcelParser object.

```
var parser = new sn_impex.GlideExcelParser();
var attachment = new GlideSysAttachment();
// use attachment sys id of an excel file
var attachmentStream = attachment.getContentStream(<attachment sys id>);

parser.parse(attachmentStream);

//retrieve the column headers
var headers = parser.getColumnHeaders();
var header1 = headers[0];
var header2 = headers[1];

//print headers
gs.info(header1 + " " + header2);

while(parser.next()) {
    var row = parser.getRow();
    //print row value for both columns
    gs.info(row[header1] + ' ' + row[header2])
}
```

GlideExcelParser - next()

Moves to the next row.

```
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### GlideExcelParser - parse(InputStream inputStream)

Parses an XLSX-formatted Excel document.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputStream</td>
<td>GlideScriptableInputStream</td>
<td>Excel document provided as an input stream.</td>
</tr>
</tbody>
</table>

**Note:** Do not set this value if using the setSource() method to parse the same source multiple times.

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates if the parse was successful. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Document was successful parsed.</td>
</tr>
<tr>
<td></td>
<td>• false: Document was not successfully parsed.</td>
</tr>
</tbody>
</table>

Example from a scripted REST API script where the Excel file is sent using an input stream in the request payload.

```java
var parser = new sn_impex.GlideExcelParser();
parser.parse(request.body.dataStream);
```

The following example shows how to parse an Excel file attachment multiple times to retrieve column headers and print values of each row for each worksheet. Results display column headers and values of three separate sheets in the Excel attachment.

```java
var parser = new sn_impex.GlideExcelParser();
```
var attachment = new GlideSysAttachment();

// Use the attachment sys_id of an Excel file
var attachmentStream = attachment.getContentStream("<attachment_sys_id>");

// Set the source to be parsed
parser.setSource(attachmentStream);

// Get the worksheet names to be parsed in the Excel document
var list_sheet_name = parser.getSheetNames();

gs.info(" Sheet Names "+ list_sheet_name.join(" ", ");

// Iterate over each worksheet in the Excel workbook
for (var i = 0; i < list_sheet_name.length; i++) {
    gs.info("*********************************************************************************
*****");
    gs.info("Sheet name: "+ list_sheet_name[i]);

    // Set the worksheet name to be parsed
    parser.setSheetName(list_sheet_name[i]);

    // Parse each worksheet set using setSheetName()
    if (parser.parse()) {
        //retrieve the column headers
        var headers = parser.getColumnHeaders();
        var header1 = headers[0];
        var header2 = headers[1];
        var header3 = headers[2];

        //print headers
        gs.info(header1 + "|" + header2 + "|" + header3);

        // Iterate over each row in the worksheet
        while (parser.next()) {
            var row = parser.getRow();
            //print row value for both columns
            gs.info(row[header1] + ' ' + row[header2] + ' ' + row[header3])
        }
    } else
        gs.info(parser.getErrorMessage());
}

Output:

*** Script: Sheet Names Incident Table, sys_user Table, knowledge Table
*** Script:
************************************************************************************
*** Script: Sheet name: Incident Table
*** Script: Number ||Opened ||Short description
*** Script: INC0010111 || 2019-07-22 14:04:57 ||ATF : Test1
*** Script: INC0009009 || 2018-08-30 01:06:16 ||Unable to access the shared folder.
*** Script:
************************************************************************************
*** Script: Sheet name: sys_user Table
*** Script: User ID ||Name ||Email
*** Script: abel.tuter || Abel Tuter ||abel.tuter@example.com
*** Script: abraham.lincoln || Abraham Lincoln ||abraham.lincoln@example.com
*** Script: adela.cervantsz || Adela Cervantsz ||adela.cervantsz@example.com
*** Script: aileen.mottern || Aileen Mottern ||aileen.mottern@example.com
*** Script: alejandra.prenatt || Alejandra Prenatt ||alejandra.prenatt@example.com
*** Script:
************************************************************************************
*** Script: Sheet name: knowledge Table
*** Script: Number ||Short description ||Author
*** Script: KB99999999 || Microsoft Outlook Issues ||System Administrator
*** Script: KB0000033 || Eclipse configuration for Android development ||System Administrator
*** Script: KB0000032 || Getting Around in Windows ||System Administrator
*** Script: KB0000031 || How can I find the MAC address of my Ethernet or wireless interface? ||Sam Sorokin

GlideExcelParser - setHeaderRowNumber(Number headerRowNumber)
Set the number of the header row to be retrieved.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>headerRowNumber</td>
<td>Number</td>
<td>The header row to be retrieved.</td>
</tr>
</tbody>
</table>
### GlideExcelParser - setNullToEmpty(Boolean empty)

Return an empty value instead of null when an Excel cell is not present.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>empty</td>
<td>Boolean</td>
<td>When true, cells that are not present return an empty value. When false, cells that are not present return null.</td>
</tr>
</tbody>
</table>

### GlideExcelParser - setSheetName(String sheetName)

Set the name of the sheet to be retrieved.

If both `setSheetNumber()` and `setSheetName()` are set, `setSheetName()` is used.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sheetName</td>
<td>String</td>
<td>The name of the sheet to be retrieved.</td>
</tr>
</tbody>
</table>

### GlideExcelParser - setSheetNumber(Number sheetNumber)

Set the number of the Excel sheet to be retrieved.
If both `setSheetNumber()` and `setSheetName()` are set, `setSheetNumber()` is ignored.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sheetNumber</td>
<td>Number</td>
<td>The Excel sheet number to retrieve.</td>
</tr>
</tbody>
</table>

### GlideExcelParser - `setSource(InputStream inputStream)`

Defines an input source for parsing multiple times or parsing each worksheet in an Excel file.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputStream</td>
<td>GlideScriptableInputStream</td>
<td>Excel document provided as an input stream.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to parse an Excel file attachment multiple times to retrieve column headers and print values of each row for each worksheet. Results display column headers and values of three separate sheets in the Excel attachment.

```java
var parser = new sn_impex.GlideExcelParser();

var attachment = new GlideSysAttachment();

// Use the attachment sys_id of an Excel file
var attachmentStream = attachment.getContentStream("<attachment_sys_id>");

// Set the source to be parsed
parser.setSource(attachmentStream);
```
// Get the worksheet names to be parsed in the Excel document
var list_sheet_name = parser.getSheetNames();

gs.info(" Sheet Names " + list_sheet_name.join(" ", ");

// Iterate over each worksheet in the Excel workbook
for (var i = 0; i < list_sheet_name.length; i++) {
    gs.info("*********************************************************************************
*****
Sheet name:    " + list_sheet_name[i]);

    // Set the worksheet name to be parsed
    parser.setSheetName(list_sheet_name[i]);

    // Parse each worksheet set using setSheetName()
    if (parser.parse()) {
        //retrieve the column headers
        var headers = parser.getColumnHeaders();
        var header1 = headers[0];
        var header2 = headers[1];
        var header3 = headers[2];

        //print headers
        gs.info(header1 + "\t\t\t" + header2 + "\t\t\t" + header3);

        // Iterate over each row in the worksheet
        while (parser.next()) {
            var row = parser.getRow();
            //print row value for both columns
            gs.info(row[header1] + '\t' + row[header2] + '\t' + row[header3])
        }
    } else
    gs.info(parser.getErrorMessage());
}

Output:

*** Script:  Sheet Names Incident Table, sys_user Table, knowledge Table
*** Script:
*********************************************************************************
Sheet name:    Incident Table
 *** Script: Number ||Opened ||Short description
GlideFilter - Scoped, Global

Enables filtering queries to determine if one or more records meet a specified set of requirements.

Methods for this API are accessible using the GlideFilter global object.

Case sensitivity

GlideFilter is case-sensitive by default. Use the setCaseSensitive() method to enable or disable case sensitivity. GlideRecord queries are case-insensitive.

The following example shows how a GlideRecord query is case-insensitive and results in the same user record with upper or lower case.

```javascript
var gr = new GlideRecord('sys_user');
gr.addQuery('first_name', 'Abel');
gr.query();
while (gr.next())
  gs.info("Upper case query: " + gr.getDisplayValue());

var gr = new GlideRecord('sys_user');
gr.addQuery('first_name', 'abel');
```
GlideRecord `query()` output shows the same results display regardless of case.

<table>
<thead>
<tr>
<th>Upper case query: Abel Tuter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower case query: Abel Tuter</td>
</tr>
</tbody>
</table>

The following example shows how GlideFilter only retrieves the matched value of the record for the upper case condition. The lower case condition does not provide a match.

```javascript
// Matches the 'Abel Tuter' user record
var gr = new GlideRecord('sys_user');
gr.query();
var condition = 'first_name=Abel';
var glideFilter = new GlideFilter(condition, 'filterCondition');
while (gr.next()) {
    if (glideFilter.match(gr, true))
        gs.info("Upper case condition: " + gr.getDisplayValue());
}

// The following code does not match 'Abel Tuter' user record
var gr = new GlideRecord('sys_user');
gr.query();
var condition = 'first_name=abel';
var glideFilter = new GlideFilter(condition, 'filterCondition');
while (gr.next()) {
    if (glideFilter.match(gr, true))
        gs.info("Lower case condition: " + gr.getDisplayValue());
}
```

The output reveals that the filter only returns upper case results with default case-sensitivity:

| Upper case condition: Abel Tuter |

The following example shows how to disable GlideFilter case-sensitivity with the `setCaseSensitive()` method. The filter matches the condition even though the case does not match the field value.

```javascript
// Matches the 'Abel Tuter' user record
var gr = new GlideRecord('sys_user');
gr.query();
var condition = 'first_name=Abel';
var glideFilter = new GlideFilter(condition, 'filterCondition');
```
while (gr.next()) {
    if (glideFilter.match(gr, true))
        gs.info("Upper case condition: " + gr.getDisplayValue());
}

// The following code disables case sensitivity and matches the same record
var gr = new GlideRecord('sys_user');
gr.query();

var condition = 'first_name=abel';
var glideFilter = new GlideFilter(condition, 'filterCondition');
glideFilter.setCaseSensitive(false);

while (gr.next()) {
    if (glideFilter.match(gr, true))
        gs.info("Lower case condition: " + gr.getDisplayValue());
}

The output reveals GlideFilter case-insensitive results:

Upper case condition: Abel Tuter
Lower case condition: Abel Tuter

Filter null values
To exclude null values from GlideFilter query results, add ISNOTEMPTY to the query condition. The following example shows how using the same encoded query with GlideRecord and GlideFilter produces different results.

var insertRecordsGr = new GlideRecord('u_test_table');
insertRecordsGr.deleteMultiple();
var dates = ['', '2021-01-29', '2021-01-30', '2021-01-31'];
dates.forEach(function(val) {
    insertRecordsGr.initialize();
    insertRecordsGr.u_date_field = val;
    insertRecordsGr.insert();
});

var now_GR = new GlideRecord('u_test_table');
now_GR.addEncodedQuery('u_date_field<javascript:gs.beginningOfToday()');
now_GR.query();

// Encoded query includes null values with GlideFilter, which might cause unexpected results
var condition = 'u_date_field<javascript:gs.beginningOfToday()';
var grWithGlideFilter = new GlideRecord('u_test_table');
grWithGlideFilter.query();
var filter = new GlideFilter(condition, 'filterCondition');

var countWithGlideFilter = 0;
while (grWithGlideFilter.next()) {
    if (GlideFilter.checkRecord(grWithGlideFilter, condition))
        countWithGlideFilter++;
}

gs.info('Record RowCount: ' + now_GR.getRowCount() + ' - Filter Count: ' +
countWithGlideFilter);

The output shows that GlideRecord returns three records, skipping empty date
values. The GlideFilter query counts the empty date value and returns a count
of four records.

Record RowCount: 3 - Filter Count: 4

To exclude null values from the GlideFilter results, add an AND condition
ISNOTEMPTY to the queried field.

// Encoded query includes null values with GlideFilter
var condition = 'u_date_field<=javascript:gs.beginningOfToday()^u_date_fieldISNOTEMPTY';

Output shows that the GlideFilter encoded query skips the null date value and
counts three results.

Record RowCount: 3 - Filter Count: 3

GlideFilter – GlideFilter(String filter, String title)
Instantiates a GlideFilter object.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Encoded query string in standard Glide format. See Encoded query strings. Results are case-sensitive, unless disabled using the setCaseSensitive() method. To exclude null values from GlideFilter query results, add ISNOTEMPTY to the query condition.</td>
</tr>
<tr>
<td>title</td>
<td>String</td>
<td>Descriptive title for the filter.</td>
</tr>
</tbody>
</table>
The following example shows how to filter the number of users named Rebekah.

```javascript
var now_GR = new GlideRecord('sys_user');
now_GR.query();
var condition = 'first_name=rebekah';
var filter = new GlideFilter(condition, 'filterCondition');
filter.setCaseSensitive(false);

var countResults = 0;
while (now_GR.next()) {
    if (filter.match(now_GR, true))
        countResults++;
}

gs.info('Number of users named Rebekah: ' + countResults);
```

Output:

```
Number of users named Rebekah: 2
```

**GlideFilter - checkRecord(GlideRecord now_GR, String filter, Boolean match)**

Compares a specified filter to the contents of a specified GlideRecord.

If the specified filter contains one condition, the method returns true if the record meets the condition.

Filters support multiple conditions, for example "active=true^number=abc^category=request". You can use the `match` parameter to define whether all conditions must be met to determine a match or just a single condition.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>GlideRecord to evaluate.</td>
</tr>
<tr>
<td>filter</td>
<td>String</td>
<td>Encoded query string in standard Glide format. See Encoded query strings.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To exclude null values from GlideFilter query results, add ISNOTEMPTY to the query condition.</td>
</tr>
</tbody>
</table>

**Note:** The `filter` values are case-sensitive. In addition, you cannot use `setCaseSensitive(false)` to change the case-sensitive value.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| match | Boolean    | Optional. Flag that indicates whether all conditions must be met if the filter parameter contains multiple conditions.  
Valid values:  
• true: All conditions must be met for the method to return true.  
• false: Only one of the conditions must be met for the method to return true.  
Default: true |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Results of the filter comparison.  
• true: Filter conditions were met.  
• false: Filter conditions were not met. |

The following example shows how to display true for each record in the Incident table that meets the filter condition. False otherwise.

```javascript
var rec = new GlideRecord('incident');
rec.query();

var bool = true;

while(rec.next())
{
    bool = GlideFilter.checkRecord(rec, "active=true");
    gs.info("number "+ rec.number + " is " + bool);
}
```

Output:

```
number INC0000060 is false  
number INC0009002 is false  
number INC0000009 is false  
...  
```
number INC0000015 is true
number INC0000016 is true
number INC0000017 is true
...

**GlideFilter - match(GlideRecord now_GR, String filter, Boolean match)**

Evaluates a filter against a specified GlideRecord.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>GlideRecord to evaluate.</td>
</tr>
</tbody>
</table>
| match   | Boolean     | Flag that indicates whether all filter conditions must match.
|         |             | Valid values:
|         |             | • true: All filter conditions must match.
|         |             | • false: Filter condition match is not required.
|         |             | Default: false                                  |

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Results of the match check.</td>
</tr>
<tr>
<td></td>
<td>• true: Filter conditions were met.</td>
</tr>
<tr>
<td></td>
<td>• false: Filter conditions were not met.</td>
</tr>
</tbody>
</table>

The following example shows how to ensure the filter conditions match against the GlideRecord provided.

```javascript
var now_GR = new GlideRecord('sys_user');
now_GR.query();
var condition = 'first_name=rebekah';
var filter = new GlideFilter(condition, 'filterCondition');
filter.setCaseSensitive(false);

var countResults = 0;
while (now_GR.next()) {
    if (filter.match(now_GR, true))
```
countResults++;
}

gs.info('Number of users named Rebekah: ' + countResults);

Output:
Number of users named Rebekah: 2

GlideFilter - setCaseSensitive(Boolean caseSensitive)
Enables or disables case-sensitive filter results.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| caseSensitiveBoolean | Boolean          | Flag that indicates whether the filter is case-sensitive. Valid values: 
|                    |                  | • true: The filter is case-sensitive. 
|                    |                  | • false: The filter is case-insensitive.          |
|                    |                  | Default: true                                    |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to disable case-sensitive results.

```javascript
var now_GR = new GlideRecord('sys_user');
own_GR.query();

var condition = 'first_name=rebekah';
var filter = new GlideFilter(condition, 'filterCondition');

// Disable case-sensitive filtering
filter.setCaseSensitive(false);

while (now_GR.next()) {
    if (filter.match(now_GR, true))
```
Output shows that the filter retrieves upper case results with a lower case condition.

GlideFlow - Client

Use the GlideFlow JavaScript API for client-side interactions with actions, flows, and subflows.

Use the GlideFlow API anywhere in the platform that accepts client scripts. The action, flow, or subflow must be set as client callable, and have a valid ACL using the Manage Security feature in Flow designer.

Some of the methods within GlideFlow return promise objects. A promise represents the eventual result of an asynchronous operation. For more information on promises, see Promise - Javascript MDN or AngularJS documentation.

Using this API, you can:

• Start actions, flows, or subflows via a script.
• Get an existing execution.
• Get the status and any available outputs.
• Wait for the completion of an action, flow, or subflow.

There is no constructor for the GlideFlow class. Access GlideFlow methods using the GlideFlow global object.

GlideFlow - getExecution(String executionId)
Get an existing execution instance by ID.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>executionId</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>A promise of an execution object.</td>
</tr>
</tbody>
</table>

In this example, the code gets an execution, then waits for it to be completed before logging the executions completion status and outputs using console.log.

```javascript
// Get an existing action and await completion
(function() {
  GlideFlow.getExecution('79cd437e0b202300a150a95e93673ae3')
    .then(function(execution) {
      return execution.awaitCompletion();
    }, errorResolver)
    .then(function(completion) {
      var status = completion.status;
      console.log(status);
      // Available Outputs:
      var outputs = completion.outputs;
      console.log(outputs);
    }, errorResolver());

  function errorResolver(error) {
    // Handle errors in error resolver
    console.error(error);
  }
})();
```

**GlideFlow - startAction(String scopedName, Map inputs)**

Start an action.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopedName</td>
<td>String</td>
<td>The scoped name of the flow to be executed.</td>
</tr>
<tr>
<td>inputs</td>
<td>Object</td>
<td>An object containing inputs defined for the action.</td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>Object</td>
<td>An object containing details on the action execution.</td>
<td></td>
</tr>
</tbody>
</table>

In this example, the code starts the global action_name action using arguments in the inputs input object variable. Upon completion, the example uses console.log or console.error to report on the success or failure of the flow.

```javascript
// Start an action and await completion.
(function() {
    var inputs = {};

    inputs['input1'] = 'string input'; // String

    GlideFlow.startAction('global.action_name', inputs)
        .then(function(execution) {
            return execution.awaitCompletion();
        }, errorResolver)
        .then(function(completion) {
            var status = completion.status;
            console.log(status);

            // Available Outputs:
            var outputs = completion.outputs;
            console.log(outputs);
        }, errorResolver());

    function errorResolver(error) {
        // Handle errors in error resolver
        console.error(error);
    }
})();
```

**GlideFlow - startFlow(String scopedName.flowName, Map inputs)**

Start a flow.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopedName</td>
<td>String</td>
<td>The scoped name of the flow to be executed.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputs</td>
<td>Object</td>
<td>An object containing inputs defined for the flow.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object containing details on the flow execution.</td>
</tr>
</tbody>
</table>

This example flow is normally triggered when a record on the incident table is updated. Because you are activating the flow from Client script, you must provide this information. The code creates an inputs variable that contains the current record and the table for the record.

```javascript
// Start a Flow
(function() {
    var inputs = {};
    inputs['current'] = { // GlideRecord
        table : 'incident',
        sys_id : '79cd437e0b202300a150a95e93673ae3'
    };
    inputs['table_name'] = 'incident';
    GlideFlow.startFlow('global.flow_name', inputs)
    .then(
        function(execution) {
            console.log('Started flow_name with execution id :' + execution.getExecutionId());
        },
        function(error) {
            console.log('Unable to start flow: ' + error);
        }
    );
})();
```

**GlideFlow - startSubflow(String scopedName.subflowName, Map inputs)**

Start a subflow.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopedName</td>
<td>String</td>
<td>The scoped name of the flow to be executed.</td>
</tr>
<tr>
<td>inputs</td>
<td>Object</td>
<td>An object containing inputs used for the subflow.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object containing details on the subflow execution.</td>
</tr>
</tbody>
</table>

In this example, the code starts the global subflow_name subflow using arguments in the inputs array variable. Upon completion, the example uses console.log or console.error to report on the success or failure of the flow.

```javascript
// Start an action and await completion.
(function() {
    var inputs = {};
    inputs['input1'] = 'string input'; // String

    GlideFlow.startSubflow('global.subflow_name', inputs)
        .then(function(execution) {
            return execution.awaitCompletion();
        }, errorResolver)
        .then(function(completion) {
            var status = completion.status;
            console.log(status);

            // Available Outputs:
            var outputs = completion.outputs;
            console.log(outputs);
            }, errorResolver());

    function errorResolver(error) {
        // Handle errors in error resolver
        console.error(error);
    }
})();
```
**GlideFlow - execution.awaitCompletion()**

Returns a completion object for the execution.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object that contains completion details for a flow or action execution.</td>
</tr>
</tbody>
</table>

In this example, an action is executed using `startAction()`, which returns an execution object. The code then uses `awaitCompletion()` on this execution object, which returns a completion object. The code uses this completion object to log the status and outputs within the execution.

```javascript
(function() {
    var inputs = {};

    inputs['input1'] = 'string input'; // String

    GlideFlow.startAction('global.action_name', inputs)
        .then(function(execution) {
            return execution.awaitCompletion();
        }, errorResolver)
        .then(function(completion) {
            var status = completion.status;
            console.log(status);

            // Available Outputs:
            var outputs = completion.outputs;
            console.log(outputs);
        }, errorResolver());

    function errorResolver(error) {
        // Handle errors in error resolver
        console.error(error);
    }

})(function() {
    var inputs = {};

    inputs['input1'] = 'string input'; // String

    GlideFlow.startAction('global.action_name', inputs)
        .then(function(execution) {
            return execution.awaitCompletion();
        }, errorResolver)
        .then(function(completion) {
            var status = completion.status;
            console.log(status);

            // Available Outputs:
            var outputs = completion.outputs;
            console.log(outputs);
        }, errorResolver());

    function errorResolver(error) {
        // Handle errors in error resolver
        console.error(error);
    }
```

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**GlideFlow - execution.getExecutionStatus()**

Returns a string containing the execution status of the current execution.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A string containing the execution status.</td>
</tr>
</tbody>
</table>

In this example, the code obtains an execution object using the getExecution method. The getExecution method requires an ID, which is returned by the method used to start the execution. The code then uses getExecutionStatus() to determine whether the execution has been completed before continuing.

```javascript
(function() {
  GlideFlow.getExecution('mamIN4Q35vmEFc744EwJV5GHRsz8fMjG')
  .then(function(execution) {
    execution.getExecutionStatus().then(
      function(status) {
        if (status === 'COMPLETE')
          execution.getOutputs().then(
            function(outputs) {
              console.log(outputs);
            },
            errorResolver
          ),
        errorResolver
      },
      errorResolver
    );
  },
  errorResolver
});

function errorResolver(error) {
  // Handle errors in error resolver
}
```
GlideFlow - execution.getOutputs()

Returns an outputs object for the execution.

Use this method to access output generated by the execution of an action, flow, or subflow.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object containing outputs for an action, flow, or subflow.</td>
</tr>
</tbody>
</table>

In this example, the code obtains an execution object using the getExecution method. After the execution is complete, the code uses getOutputs() to return an outputs object, which it then logs using the console.log method.

```javascript
console.error(error);
}
})

// Get an existing action, getStatus, and getOutputs if complete
(function() {
    GlideFlow.getExecution('mamIN4Q35vmEFv744EwJv5GHRsz8fmJG')
    .then(function(execution) {
        execution.getExecutionStatus().then(
            function(status) {
                if (status === 'COMPLETE')
                    execution.getOutputs().then(
                        function(outputs) {
                            console.log(outputs);
                        },
                        errorResolver
                    ),
                    errorResolver
                });
            },
            errorResolver
        ),
        errorResolver
    });
```
function errorResolver(error) {
    // Handle errors in error resolver
    console.error(error);
}
}}();

GlideForm - Client

The GlideForm API provides methods to customize forms.

GlideForm.js is the JavaScript class containing the methods. The global object g_form is used to access GlideForm methods. GlideForm methods are only used on the client. These methods are used to make custom changes to the form view of records. All validation of examples was done using client scripts.

Some of these methods can also be used in other client scripts (such as Catalog Client Scripts or Wizard Client Scripts), but must first be tested to determine whether they will work as expected.

⚠️ **Note:** The methods getControl(), getHelpTextControl(), getElement(), and getFormElement() are deprecated for mobile devices. For information on using GlideForm for mobile, see Mobile Client GlideForm (g_form) Scripting and Migration.

There is no constructor for the GlideForm class. Access GlideForm methods using the g_form global object.

GlideForm - addDecoration(String fieldName, String icon, String title)

Adds an icon on a field’s label.

Adding the same item twice is prevented; however, you can add the same icon with a different title.

⚠️ **Note:** This method is not supported by Service Catalog.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field name.</td>
</tr>
<tr>
<td>icon</td>
<td>String</td>
<td>The font icon to show next to the field. Supported icons - icon-user, icon-user-group, icon-lightbulb, icon-home, icon-mobile, icon-comment, icon-mail, icon-locked, icon-</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

| title        | String | The text title for the icon.                                                 |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
const g_form = new GlideForm();
g_form.addDecoration('caller_id', 'icon-star', 'preferred member');
```

GlideForm - addDecoration(String fieldName, String icon, String title, String color)

Adds an icon on a field's label.

Adding the same item twice is prevented; however, you can add the same icon with a different title.

⚠️ Note: This method is not supported by Service Catalog.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>String</td>
<td>A CSS color.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
g_form.addDecoration('caller_id', 'icon-star', 'Mark as Favorite', 'color-green');
```

GlideForm - addErrorMessage(String message)

Displays the specified error message at the top of the form.

This message appears for approximately four seconds and then disappears. This timeout is not configurable at this time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Message to display.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
g_form.addErrorMessage('This is an error');
```
GlideForm - addFormMessage(String message, String type)
Displays a floating form message at the top of the form detail section. The message does not cover UI actions.

See also:
• clearAllFormMessages()
• clearFormMessages()

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Message to display.</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>The type of message. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• error</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• info</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• warning</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add form messages of each type.

```java
    g_form.addFormMessage('info message','info');
    g_form.addFormMessage('warning message','warning');
    g_form.addFormMessage('error message','error');
    g_form.addFormMessage('info2 message','info');
    g_form.addFormMessage('warning2 message','warning');
    g_form.addFormMessage('error2 message','error');
```

GlideForm - addInfoMessage(String message)

Adds the specified informational message to the top of the form.

This message appears for approximately four seconds and then disappears. This timeout is not configurable at this time.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Message to display.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
g_form.addInfoMessage('The top five fields in this form are mandatory');
```

**GlideForm - addOption(String fieldName, String choiceValue, String choiceLabel)**

Adds a choice to the end of a choice list field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The name of the field.</td>
</tr>
<tr>
<td>choiceValue</td>
<td>String</td>
<td>The value to be stored in the database.</td>
</tr>
<tr>
<td>choiceLabel</td>
<td>String</td>
<td>The value displayed.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
g_form.addOption('priority', '6', '6 - Really Low');
```

**GlideForm - addOption(String fieldName, String choiceValue, String choiceLabel, Number choiceIndex)**

Adds a choice to the choice list field at the position specified.

```javascript
g_form.addOption('priority', '6', '6 - Really Low');
```
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field name.</td>
</tr>
<tr>
<td>choiceValue</td>
<td>String</td>
<td>The value stored in the database.</td>
</tr>
<tr>
<td>choiceLabel</td>
<td>String</td>
<td>The value displayed.</td>
</tr>
<tr>
<td>choiceIndex</td>
<td>Number</td>
<td>Order of the choice in the list. The index is into a zero based array.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
GlideForm.addOption('priority', '2.5', '2.5 - Moderately High', 3);
```

**GlideForm - clearMessages()**

Removes all informational and error messages from the top of the form.

Removes informational and error messages added with `GlideForm.addInfoMessage()` and `GlideForm.addErrorMessage()`.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
GlideForm.clearMessages();
```

**GlideForm - clearAllFormMessages()**

Removes all form messages of any type.
See also:

- `addFormMessage()`
- `clearFormMessages()`

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to clear all messages from the form.

```javascript
g_form.clearAllFormMessages();
```

**GlideForm - clearFormMessages(String type)**

Removes all form messages of a given type.

See also:

- `addFormMessage()`
- `clearAllFormMessages()`

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| type | String| The type of message. Valid values:  
  - error  
  - info   
  - warning |
The following example shows how to clear all error messages from the form:

```javascript
g_form.clearFormMessages('error');
```

**GlideForm - clearOptions(String fieldName)**

Removes all options from the choice list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideForm - clearValue(String fieldName)**

Removes any value(s) from the field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideForm - disableAttachments()**

Prevents file attachments from being added.
This method is not available on the mobile platform. If this method is run on a mobile platform, no action occurs.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideForm - enableAttachments()**

Allows file attachments to be added. Shows the paper clip icon.

This method is not available on the mobile platform. If this method is run on a mobile platform, no action occurs.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideForm - flash(String fieldName, String color, Number count)**

Used to draw attention to a particular field. Flashes the specified color for a specified duration of time in the specified field.

This method is not supported by Service Catalog.

This method is not available on the mobile platform. If this method is run on a mobile platform, no action occurs.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Specifies the field to highlight in the following format: &quot;&lt;table-name&gt;.&lt;field-name&gt;&quot;.</td>
</tr>
<tr>
<td>color</td>
<td>String</td>
<td>RGB color or acceptable CSS color.</td>
</tr>
<tr>
<td>count</td>
<td>Number</td>
<td>Specifies how long the label will flash. Options include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2: Flashes for 1 second</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0: Flashes for 2 seconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• -2: Flashes for 3 seconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• -4: Flashes for 4 seconds</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
g_form.flash("incident.number", "#FFFACD", 0);
```

GlideForm - getActionName()

Returns the most recent action name, or, for a client script, the sys_id of the UI action clicked.

ℹ️ Note: Not available in Wizard client scripts.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The current action name.</td>
</tr>
</tbody>
</table>
function onSubmit() {
    var action = g_form.getActionName();
    alert('You pressed ' + action);
}

**GlideForm - getBooleanValue(String fieldName)**

Returns a Boolean value for the specified field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns false if the field value is false or undefined; otherwise returns true.</td>
</tr>
</tbody>
</table>

**GlideForm - getControl(String fieldName)**

Returns the HTML element for the specified field.

Compound fields may contain several HTML elements. This method is generally not necessary as there are built-in methods that use the fields on a form.

If the field is a reference field and the control is a choice list, `getControl()` may not return a control as expected. In this case, use `sys_select.<table name>..<field name>`. This method is not available in mobile scripts or Service Portal scripts.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTMLElement</td>
<td>The field's HTML element.</td>
</tr>
</tbody>
</table>

**GlideForm - getDecimalValue(String fieldName)**

Returns the decimal value of the specified field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The name of the field.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The decimal value of the specified field.</td>
</tr>
</tbody>
</table>

```javascript
function onChange(control, oldValue, newValue, isLoading) {
    alert(g_form.getDecimalValue('percent_complete'));
}
```

**GlideForm - getElement(String id)**

Returns the HTML element specified by the parameter.

Compound fields may contain several HTML elements. This method is generally not necessary as there are built-in methods that use the fields on a form. This method is not available in mobile scripts or Service Portal scripts.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>The field id.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTMLElement</td>
<td>The field's HTML element.</td>
</tr>
</tbody>
</table>

**GlideForm - getFormElement()**

Returns the HTML element for the form.

This method is not available in mobile scripts or Service Portal scripts.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTMLFormElement</td>
<td>The HTML element for the form.</td>
</tr>
</tbody>
</table>

**GlideForm - getHelpTextControl(String fieldName)**

Returns the HTML element of the help text for the specified field.

This method is applicable to service catalog variables only.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTMLElement</td>
<td>Help text field's HTML element.</td>
</tr>
</tbody>
</table>
**GlideForm - getIntValue(String fieldName)**

Returns the integer value of the field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field name.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Integer value of the field.</td>
</tr>
</tbody>
</table>

**GlideForm - getLabelOf(String fieldName)**

Returns the plain text value of the field label.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field name</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The label text.</td>
</tr>
</tbody>
</table>

```javascript
if (g_user.hasRole('itil')) {
    var oldLabel = g_form.getLabelOf('comments');
    g_form.setLabelOf('comments', oldLabel + ' (Customer visible)');
}
```

**GlideForm - getOption(String fieldName, String choiceValue)**

Returns the option element for a selected box named `fieldName` where `choiceValue` matches the option value.

**Note:** This method does not work on read-only fields.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
<tr>
<td>choiceValue</td>
<td>String</td>
<td>Value of the option.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTMLElement</td>
<td>The HTMLElement for the option. Returns null if the field or option is not found.</td>
</tr>
</tbody>
</table>

The following example shows how to get the label for a choice list value.

```
// Get the label for a choice list value
// fieldName is 'category'

function onChange(control, oldValue, newValue, isLoading) {
  var choiceValue = g_form.getValue('category');
  var choiceLabel = g_form.getOption('category', choiceValue).text;
}
```

**GlideForm - getRelatedListNames()**

Returns an array of related lists from the current form in the order in which they appear on that form.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array of related lists from the current form in the order in which they appear on that form.</td>
</tr>
</tbody>
</table>
for (var i = 0; i < listNames.length; i++) {
    this.showRelatedList(listNames[i]);
}

**GlideForm - getReference(String fieldName, Function callBack)**

Returns the GlideRecord for a specified field.

If a callback function is present, this routine runs asynchronously. The browser (and script) processing continues normally until the server returns the reference value, at which time, the callback function is invoked. If a callback function is not present, this routine runs synchronously and processing halts (causing the browser to appear to hang) while waiting on a server response.

⚠️ **Important:** It is strongly recommended that you use a callback function.

Callback function support for ServiceCatalogForm.getReference is available.

⚠️ **Note:** Using this method requires a call to the server which requires additional time and may introduce latency to your page. Use this method with caution. For additional information, see Client script design and processing.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
<tr>
<td>callBack</td>
<td>Function</td>
<td>Name of the call back function.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>GlideRecord object for the specified field.</td>
</tr>
<tr>
<td></td>
<td>If the specified reference cannot be found, it returns an initialized GlideRecord object where currentRow = -1 and rows.length = 0.</td>
</tr>
</tbody>
</table>

```javascript
function onChange(control, oldValue, newValue, isLoading) {
    g_form.getReference('caller_id', doAlert); // doAlert is our callback function
}

function doAlert(caller) { // reference is passed into callback as first arguments
    ...
}
```
if (caller.getValue('vip') == 'true') {
    alert('Caller is a VIP!');
}

GlideForm - getSectionNames()

Returns all section names, whether visible or not.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array of strings</td>
<td>The section names.</td>
</tr>
</tbody>
</table>

GlideForm - getSections()

Returns an array of the form's sections.

This method is not available on the mobile platform. If this method is run on a mobile platform, no action occurs.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array of HTML elements</td>
<td>The form's sections.</td>
</tr>
</tbody>
</table>

function onChange(control, oldValue, newValue, isLoading) {
    //this example was run on a form divided into sections (Change form)
    // and hid a section when the "state" field was changed
    var sections = g_form.getSections();
if (newValue == '2') {
    g_form.setSectionDisplay(sections[1], false);
} else {
    g_form.setSectionDisplay(sections[1], true);
}

**GlideForm - getTableName()**

Returns the name of the table to which this record belongs.

On the server side, the table for the current record can be retrieved with current.sys_class_name or current.getTableName().

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the table.</td>
</tr>
</tbody>
</table>

function onLoad() {
    if (g_form.isNewRecord()) {
        var tableName = g_form.getTableName(); //Get the table name
    }
}

**GlideForm - getUniqueValue()**

Returns the sys_id of the record displayed in the form.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The record’s sys_id.</td>
</tr>
</tbody>
</table>

```
function onLoad() {
    var incSysid = g_form.getUniqueValue();
    alert(incSysid);
}
```

**GlideForm - getValue(String fieldName)**

Returns the value of the specified form field.

This method also supports getting values from a multi-row variable set (MRVS). To obtain data from fields within an MRVS, you must first use `JSON.parse(getValue('<mrvs_field_name>'))` to obtain the MRVS array, and then use indexing to access the fields within the row objects. For more details, see the code example below.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field whose value to return.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>String</td>
<td>Value of the specified field.</td>
</tr>
</tbody>
</table>

The following example shows how to get the short description from the current form.

```
function onChange(control, oldValue, newValue, isLoading) {
    alert(g_form.getValue('short_description'));
}
```

The following example shows how to get values from an MVRS. In this example, salaries are being managed through the Service Catalog. The client script searches all rows within the MRVS for the value entered in the **Job title** and then updates the matching entries within the MVRS with what is entered in the **Job title**.
Salary field. The MRVS is named "variable_set_1" and contains the following fields within each row object: Employee name [employee_name], Job title [employee_job_title], and Salary [employee_salary]. In addition, the Catalog Item contains: Job title [job_title] and Salary [salary].

```javascript
function onChange(control, oldValue, newValue, isLoading) {
  if (isLoading || newValue == '') {
    return;
  }

  // Get the MRVS
  var multiRowVariableSet = JSON.parse(g_form.getValue('variable_set_1'));

  for (var i = 0; i < multiRowVariableSet.length; i++) {
    // Check if the entered job title matches the title in the current MRVS row
    if (multiRowVariableSet[i].employee_job_title == g_form.getValue("job_title")) {
      // Update the value of a matching field with the new salary
      multiRowVariableSet[i].employee_salary = newValue;
    }
  }

  // Update the MRVS
  g_form.setValue('variable_set_1', JSON.stringify(multiRowVariableSet));
}
```

**GlideForm - hideAllFieldMsgs()**

Hides all field messages.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideForm - hideAllFieldMsgs(String type)**

Hides all field messages of the specified type.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>The type of message. Valid values: • error • info</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideForm - hideErrorBox(String fieldName)**

Hides the error message placed by `showErrorBox()`.

Whenever possible, use `hideFieldMsg()` rather than this method whenever possible.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The name of the field or control.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideForm - hideFieldMsg(String fieldName)**

Hides the last message placed by `showFieldMsg()`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
</tbody>
</table>
Returns
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideForm - hideFieldMsg(String fieldName, Boolean clearAll)**

Hides the messages placed by `showFieldMsg()`.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
<tr>
<td>clearAll</td>
<td>Boolean</td>
<td>When true, all messages for the field are cleared. When false, only the last message is removed.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
    g_form.hideFieldMsg('impact', true);
```

**GlideForm - hideRelatedList(String listTableName)**

Hides the specified related list on the form.

This method is not available on the mobile platform. If this method is run on a mobile platform, no action occurs.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>listTableName</td>
<td>String</td>
<td>Name of the related list. Use the sys_id to hide a list through a relationship.</td>
</tr>
</tbody>
</table>
GlideForm - hideRelatedLists()

Hides all related lists on the form.

This method is not available on the mobile platform. If this method is run on a mobile platform, no action occurs.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

GlideForm - isLiveUpdating()

Returns true while a live update is being done on the record the form is showing.

This can be used in an `onChange()` client script to determine if a change to the record is because of a live update from another session. The client script can then decide what action to take, or not to take. This applies to systems using UI16 with live forms enabled.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if a live update is happening on the record displayed by the form.</td>
</tr>
</tbody>
</table>

**GlideForm - isMandatory(String fieldName)**

Returns true if the field is mandatory.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the field is required, false otherwise.</td>
</tr>
</tbody>
</table>

**GlideForm - isNewRecord()**

Returns true if the record has never been saved.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the record has not been saved; otherwise false.</td>
</tr>
</tbody>
</table>

```javascript
function onLoad() {
    if (g_form.isNewRecord()){
        alert('New Record!');
    }
}
```
**GlideForm - isSectionVisible(String sectionName)**

Returns true if the section is visible.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true when the section is visible; otherwise, false is returned.</td>
</tr>
</tbody>
</table>

**GlideForm - onUserChangeValue(Function fn)**

Registers a custom event listener that detects when any field in the current form is modified by a user.

When a form field is modified, the event listener calls the function that is passed in when the listener is initially registered. This listener is only triggered when a user makes a change to a field on the form. Changes from client scripts, UI policies, or any other non-user interactions, do not trigger the listener.

⚠️ **Note:** This method does not work for journal fields.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fn</td>
<td>Function</td>
<td>Function to call when a user changes the value of a field within the current form. This is actually the function code, not just the function name. This function must accept the following three arguments:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• field name</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• original field value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• updated field value</td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Function to call to unregister the onUserChangeValue event listener.</td>
<td></td>
</tr>
</tbody>
</table>

**Example**

```javascript
var handler = function(fieldname, originalValue, newValue) {
  console.log('The field (' + fieldname + ') has a new value of: ' + newValue); // function code
}

var unregister = g_form.onUserChangeValue(handler);

// To unregister the event listener
unregister();
```

**GlideForm - refreshSlushbucket(String fieldName)**

You can update a list collector variable.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the slush bucket.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
g_form.refreshSlushbucket('bucket');
```

**GlideForm - removeDecoration(String fieldname, String icon, String title)**

Removes the icon from the specified field that matches the icon and title.

ℹ️ **Note:** This method is not supported by Service Catalog.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Field name.</td>
</tr>
<tr>
<td>icon</td>
<td>String</td>
<td>Name of the icon to remove.</td>
</tr>
<tr>
<td>title</td>
<td>String</td>
<td>The icon's text title (name).</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
function onChange(control, oldValue, newValue, isLoading) {
  // if the caller_id field is not present, then we can't add an icon anywhere
  if (!g_form.hasField('caller_id'))
    return;
  if (!newValue)
    return;

g_form.getReference('caller_id', function(ref) {
  g_form.removeDecoration('caller_id', 'icon-star', 'VIP');
  if (ref.getValue('vip') == 'true')
    g_form.addDecoration('caller_id', 'icon-star', 'VIP');
});
}
```

**GlideForm - removeDecoration(String fieldName, String icon, String title, String color)**

Removes the icon from the specified field that matches the icon, title, and color.

ℹ️ **Note:** This method is not supported by Service Catalog.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Field name.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>icon</td>
<td>String</td>
<td>Name of the icon to remove.</td>
</tr>
<tr>
<td>title</td>
<td>String</td>
<td>The icon's text title (name).</td>
</tr>
<tr>
<td>color</td>
<td>String</td>
<td>A CSS color</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
g_form.removeDecoration('caller_id', 'icon-star', 'VIP', 'blue');
```

**GlideForm - removeOption(String fieldName, String choiceValue)**

Removes the specified option from the choice list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
<tr>
<td>choiceValue</td>
<td>String</td>
<td>The value stored in the database. This is not the label.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
g_form.removeOption('priority', '1');
```

**GlideForm - save()**

Saves the record without navigating away (update and stay).
GlideForm - `setDisabled(String fieldName, Boolean disable)`

Makes the specified field available or unavailable.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
<tr>
<td>disable</td>
<td>Boolean</td>
<td>When true disables the field. When false enables the field.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideForm - `setDisplay(String fieldName, Boolean display)`

Displays or hides a field.

This method cannot hide a mandatory field with no value. If the field is hidden, the space is used to display other items. Whenever possible, use a UI policy instead of this method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>display</td>
<td>Boolean</td>
<td>When true displays the field, when false hides the field.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
function onChange(control, oldValue, newValue, isLoading, isTemplate) {
    // If the page isn't loading
    if (!isLoading) {
        // If the new value isn't blank
        if (newValue != '') {
            g_form.setDisplay('priority', false);
        } else {
            g_form.setDisplay('priority', true);
        }
    }
    //glideform function
}
```

### GlideForm - setLabelOf(String fieldName, String label)

Sets the plain text value of the field label.

⚠️ **Note:** This method is not supported by Service Catalog.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field name.</td>
</tr>
<tr>
<td>label</td>
<td>String</td>
<td>The field text label.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
if (g_user.hasRole('itil')) {
    var oldLabel = g_form.getLabelOf('comments');
    g_form.setLabelOf('comments', oldLabel + ' (Customer visible)');
}

GlideForm - setMandatory(String fieldName, Boolean mandatory)
Makes the specified field mandatory.
Whenever possible, use a UI policy rather than this method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
<tr>
<td>mandatory</td>
<td>Boolean</td>
<td>When true makes the field mandatory. When false makes the field optional.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideForm - setReadOnly(String fieldName, Boolean readOnly)
Makes the specified field read-only or editable.
Whenever possible, use a UI policy instead of this method.
To make a mandatory field read-only, you must first remove the mandatory requirement for that field by using the setMandatory() method.
Once you set a field to read-only, you cannot use the setValue() method to update the value of that field. If you need to set the value in this way, you must set the readOnly value to false.

Note: The function name setReadonly() also works.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
</tbody>
</table>
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>readOnly</td>
<td>Boolean</td>
<td>Flag that determines whether the associate field is editable or read-only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Set field to read-only</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Set field to be editable</td>
</tr>
</tbody>
</table>

## GlideForm - setSectionDisplay(String sectionName, Boolean display)

Shows or hides a section.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sectionName</td>
<td>String</td>
<td>The section name is lower case with an underscore replacing the first space in the name, and with the remaining spaces being removed, for example &quot;Section Four is Here&quot; becomes &quot;section_fourishere&quot;. Other non-alphanumeric characters, such as ampersand (&amp;), are removed. Section names can be found by using the getSectionNames() method.</td>
</tr>
<tr>
<td>display</td>
<td>Boolean</td>
<td>When true shows the section. When false hides the section.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true when successful.</td>
</tr>
</tbody>
</table>
This method also supports setting values in a multi-row variable set (MRVS). You must first use `JSON.parse(getValue('<mrvs_field_name>'))` to obtain the MRVS array and then use indexing to update the fields within the row objects. Once all values are updated in the MRVS, use the `setValue()` method to save the updated MRVS array. For more details, see the code example below.

⚠️ **Note:** The method `setValue()` can cause a stack overflow when used in an OnChange client script. This is because every time the value is set, it will register as a change, which may re-trigger the OnChange client script. To prevent this, perform a check that will validate that the new value will be different from the old value. For example, before performing `setValue(shortDesc, newValue.toUpperCase());`, validate that the short description is not already uppercase. This will prevent the client script from applying the `toUpperCase()` more than once.

⚠️ **Note:** When defining a value in a choice list, be sure to use the number value rather than the label.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the form field to update.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>String value to set in the specified field.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the short description in the current form.

```javascript
  g_form.setValue('short_description', 'replace this with appropriate text');
```

The following example shows how to set values in an MVRS. In this example, salaries are being managed through the Service Catalog. The client script searches all rows within the MRVS for the value entered in the **Job title** and then updates the matching entries within the MVRS with what is entered in the **Salary** field. The MRVS is named "variable_set_1" and contains the following fields within each row object: Employee name [employee_name], Job title [employee_job_title], and Salary [employee_salary]. In addition, the Catalog Item contains: Job title [job_title] and Salary [salary].
function onChange(control, oldValue, newValue, isLoading) {
    if (isLoading || newValue == '') {
        return;
    }

    // Get the MRVS
    var multiRowVariableSet = JSON.parse(g_form.getValue('variable_set_1'));

    for (var i = 0; i < multiRowVariableSet.length; i++) {
        // Check if the entered job title matches the title in the current MRVS row
        if (multiRowVariableSet[i].employee_job_title == g_form.getValue("job_title")) {
            // Update the value of a matching field with the new salary
            multiRowVariableSet[i].employee_salary = newValue;
        }
    }

    // Update the MRVS
    g_form.setValue('variable_set_1', JSON.stringify(multiRowVariableSet));
}

GlideForm - setValue(String fieldName, String value, String displayValue)
Sets the value of a specified form field to the value of a specified display value in a reference record.

To improve performance by preventing a round trip when setting the value for a reference field, use this method, not setValue(fieldName, value). When setting multiple reference values for a list collector field, pass arrays in the value and displayValue parameters.

ℹ️ Note: The method `setValue()` can cause a stack overflow when used in an OnChange client script. This is because every time the value is set, it will register as a change, which may re-trigger the OnChange client script. To prevent this, perform a check that will validate that the new value will be different from the old value. For example, before performing `setValue(shortDesc, newValue.toUpperCase());`, validate that the short description is not already uppercase. This will prevent the client script from applying the `toUpperCase()` more than once.

ℹ️ Note: When defining a value in a choice list, be sure to use number value rather than the label.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the form field to update.</td>
</tr>
<tr>
<td>value</td>
<td>String or Array</td>
<td>Sys_id of the reference record to use to update the field. If the specified field is a GlideList, this parameter can contain an array of sys_ids. In this case, the method performs a lookup of all records specified in the array and those values are used to update the contents of the specified field (related list).</td>
</tr>
<tr>
<td>displayValue</td>
<td>String or Array</td>
<td>Field within the specified reference record to use to update the specified field. For example, in the User [sys_user] table it might be userName. If the specified field is a GlideList, this parameter can contain an array of display value names. For additional information on display values, see Display value.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows passing the sys_id of the reference record that contains the userName field to use to update the assigned_to form field.

```javascript
const userSysID = 'some sys_id';
const userName = 'some display value';
g_form.setValue('assigned_to', userSysID, userName);
```

This example shows passing an array of reference record sys_ids and an array of corresponding display value names to use to update the form fields in the GlideList glide-list_field_name.

```javascript
const sysIDArray = ['sys_id1', 'sys_id2'];
const displayNameArray = ['display_value1', 'display_value2'];
g_form.setValue('glide-list_field_name', sysIDArray, displayNameArray);
```

**GlideForm - setVisible(String fieldName, Boolean display)**

Displays or hides the field.

On desktop UI, the space is left blank when hidden. On Mobile or Service Portal UI, the space is filled in my other fields when hidden. This method cannot hide mandatory fields with no value.
Use UI Policy rather than this method whenever possible.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field name.</td>
</tr>
<tr>
<td>display</td>
<td>Boolean</td>
<td>When true displays the field. When false hides the field.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
function onChange(control, oldValue, newValue, isLoading, isTemplate) {
  //If the page isn't loading
  if (!isLoading) {
    //If the new value isn't blank
    if (newValue != '') {
      g_form.setVisible('priority', false);
    }
    else
      g_form.setVisible('priority', true);
  }
}
```

**GlideForm - showErrorBox(String name, String message)**

Displays an error message under the specified form field (either a control object or the name of the field). If the control or field is currently off the screen, the form scrolls to the control or field.

A global property (glide.ui.scroll_to_message_field) is available that controls automatic message scrolling when the form field is off screen (scrolls the form to the control or field). The `showFieldMsg()` method is a similar method that requires a type parameter.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the control or field.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The message to be displayed.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideForm - showErrorBox(String name, String message, Boolean scrollForm)**

Displays an error message under the specified form field (either a control object or the name of the field). If the control or field is currently off the screen and the scrollForm parameter is true, the form scrolls to the control or field.

A global property (glide.ui.scroll_to_message_field) is available that controls automatic message scrolling when the form field is off screen (scrolls the form to the control or field). The `showFieldMsg()` method is a similar method that requires a type parameter.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field or control.</td>
</tr>
<tr>
<td>message</td>
<td>String</td>
<td>Message to display.</td>
</tr>
<tr>
<td>scrollForm</td>
<td>Boolean</td>
<td>When true scrolls the form to the field. When false the form does not scroll to the field.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideForm - showFieldMsg(String field, String message, String type)**

Displays either an informational or error message under the specified form field (either a control object or the name of the field). If the control or field is off the screen, the form is scrolled to the field.
A global property (glide.ui.scroll_to_message_field) is available that controls automatic message scrolling when the form field is off screen (scrolls the form to the control or field).

The showErrorBox() method is a shorthand method that does not require the type parameter.

Note: This method does not work with the journal_field type field in UI16.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Name of the field or control.</td>
</tr>
<tr>
<td>message</td>
<td>String</td>
<td>Message to display.</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>&quot;error&quot;,&quot;info&quot;, or &quot;warning&quot;.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
g_form.showFieldMsg('impact','Low impact response time can be one week','info');
```

**GlideForm - showFieldMsg(String field, String message, String type, Boolean scrollForm)**

Displays either an informational or error message under the specified form field (either a control object or the name of the field). If the control or field is currently off the screen and scrollForm is true, the form is scrolled to the field.

A global property (glide.ui.scroll_to_message_field) is available that controls automatic message scrolling when the form field is off screen (scrolls the form to the control or field).

The showErrorBox() method is a shorthand method that does not require the type parameter.

Note: This method does not work with the journal_field type field in UI16.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Name of the field or control.</td>
</tr>
<tr>
<td>message</td>
<td>String</td>
<td>Message to display.</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>&quot;error&quot;, &quot;info&quot;, or &quot;warning&quot;.</td>
</tr>
<tr>
<td>scrollForm</td>
<td>Boolean</td>
<td>When true, the form scrolls to the field if it is off screen. When false, the form does not scroll.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
g_form.showFieldMsg('impact','Low impact not allowed with High priority','error',false);
```

#### GlideForm - showRelatedList(String listTableName)

Displays the specified related list on the form.

This method is not available on the mobile platform. If this method is run on a mobile platform, no action occurs.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>listTableName</td>
<td>String</td>
<td>Name of the related list.</td>
</tr>
</tbody>
</table>

### GlideForm - showRelatedLists()

Displays all the form's related lists.
This method is not available on the mobile platform. If this method is run on a mobile platform, no action occurs.

### GlideForm - submit()
Saves the record.
The user is taken away from the form, returning them to where they were.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### GlideForm - submit(String verb)
Performs the UI action specified by the parameter.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>verb</td>
<td>String</td>
<td>An action_name from a sys_ui_action record. The action name must be for a visible form button.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Mobile GlideForm (g_form) - Client**

Mobile GlideForm (g_form) methods enable you to work with forms on the mobile platform.

Use these methods in any script that targets a mobile device.

**MobileGlideForm (g_form) - addDecoration(String fieldName, String icon, String text)**

Adds a decorative icon next to a field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field name.</td>
</tr>
<tr>
<td>icon</td>
<td>String</td>
<td>The font icon to show next to the field.</td>
</tr>
<tr>
<td>text</td>
<td>String</td>
<td>The text title for the icon (used for screen readers).</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example adds a VIP icon next to the caller.

```javascript
function onChange(control, oldValue, newValue, isLoading) {
    // if the caller_id field is not present, then we can't add an icon anywhere
    if (!g_form.hasField('caller_id'))
        return;

    if (!newValue)
        return;

    g_form.getReference('caller_id', function(ref) {
```
g_form.removeDecoration('caller_id', 'icon-star', 'VIP');

if (ref.getValue('vip') == 'true')
    g_form.addDecoration('caller_id', 'icon-star', 'VIP');
});
}

**MobileGlideForm (g_form) - getLabel(String fieldName)**

Gets the form label text.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field name.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The label text.</td>
</tr>
</tbody>
</table>

if (g_user.hasRole('itil')) {
    var oldLabel = g_form.getLabel('comments');
    g_form.setLabel('comments', oldLabel + ' (Customer visible)');
}

**MobileGlideForm (g_form) - hasField(String fieldName)**

Determines if a field is present on the form.

Present means that it can be shown, not that it is visible.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field to look for.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the field is present on the form; false, if it is not. On the form means that the field is part of g_form. It could still be hidden, read-only, mandatory, or invalid.</td>
</tr>
</tbody>
</table>

This example makes the assigned_to field mandatory if the assignment_group field is on the form.

```javascript
if (g_form.hasField('assignment_group'))
    g_form.setMandatory('assigned_to', true);
```

**MobileGlideForm (g_form) - removeDecoration(String fieldName, String icon, String text)**

Removes a decorative icon from next to a field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field name.</td>
</tr>
<tr>
<td>icon</td>
<td>String</td>
<td>The icon to remove.</td>
</tr>
<tr>
<td>text</td>
<td>String</td>
<td>The text title for the icon.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
function onChange(control, oldValue, newValue, isLoading) {
    // if the caller_id field is not present, then we can't add an icon anywhere
    if (!g_form.hasField('caller_id'))
        return;
    if (!newValue)
        return;
```
```javascript
function onBeforeLoad() {
  g_form.getReference('caller_id', function(ref) {
    g_form.removeDecoration('caller_id', 'icon-star', 'VIP');

    if (ref.getValue('vip') == 'true') {
      g_form.addDecoration('caller_id', 'icon-star', 'VIP');
    }
  });
}
```

**MobileGlideForm (g_form) - setLabel(String fieldName, String label)**

Sets the form label text.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field name.</td>
</tr>
<tr>
<td>label</td>
<td>String</td>
<td>The field label text.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example changes the comments label.

```javascript
if (g_user.hasRole('itil')) {
  var oldLabel = g_form.getLabel('comments');
  g_form.setLabel('comments', oldLabel + ' (Customer visible)');
}
```

**GlideFormScratchpad - Scoped, Client**

The scoped GlideFormScratchpad class implements the g_scratchpad object for scoped applications.

The scoped GlideFormScratchpad class has no constructor and no methods. The g_scratchpad object behaves identically for global and scoped applications.

The g_scratchpad object provides a mechanism for passing information from the server to the client when the client requires information not available on a form. This can be accomplished by creating a business rule to put the information in the g_scratchpad object and accessing the information in a client script.
For an example on using this class, Example: g_scratchpad on the Developer Site.

**GlideGuidV3 - Client**
You can create a globally unique identifier.
You access the GlideGuidV3 methods using the g_guid global object.

**GlideGuidV3 - generate(Number stringLength)**
Creates a globally unique identifier 32 characters long, or as specified with the optional length argument.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stringLength</td>
<td>Number</td>
<td>The desired string length, must be between 1 and 32 inclusive. This parameter is optional. If not specified, the returned string will be 32 characters long.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The globally unique identifier.</td>
</tr>
</tbody>
</table>

**GlideHTTPRequest - Global**
Utility methods to perform common functions with Glide HTTP requests.
Use GlideHTTPRequest in global server-side scripts. To use this class, instantiate a GlideHTTPRequest object using the constructor. The constructor requires an endpoint URL as an input parameter.

**GlideHTTPRequest - addHeader(String name, String value)**
Add a header to your HTTP request.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The header name</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The header value</td>
</tr>
</tbody>
</table>
GlideHTTPRequest - addParameter(String name, String value)
Add a parameter to your HTTP request.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The header name</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The header value</td>
</tr>
</tbody>
</table>

GlideHTTPRequest - executeMethod(HttpMethod method)
Executes the method defined in the parameter.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>method</td>
<td>HttpMethod</td>
<td>Choose an HTTP method</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Delete</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Fetch</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Get</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Post</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Put</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>The status value.</td>
</tr>
</tbody>
</table>
**GlideHTTPRequest - getErrorMessage()**

Returns the last error encountered.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Returns a text string that describes the last error encountered</td>
</tr>
</tbody>
</table>

**GlideHTTPRequest - setBasicAuth(String userName, String password)**

Set a user name and password for basic authentication.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userName</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>password</td>
<td>String</td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideHTTPRequest - setContentType(String type)**

Set the content type for your HTTP request.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>The content type.</td>
</tr>
</tbody>
</table>
### GlideHTTPRequest - setFollowRedirect(boolean followRedirect)
Enable or disable the follow redirect option.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>followRedirect</td>
<td>Boolean</td>
<td>This value is true by default. Set to false to prevent following redirects.</td>
</tr>
</tbody>
</table>

### GlideHTTPRequest - setHttpTimeout(int timeout)
Set the HTTP timeout value in milliseconds.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeout</td>
<td>Integer</td>
<td>The timeout in milliseconds.</td>
</tr>
</tbody>
</table>

### GlideHTTPRequest - setLogLevel(String logLevel)
Set the log level for HTTP requests.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>logLevel</td>
<td>String</td>
<td>The level of logging available. Choose from one of the following values.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• basic</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• elevated</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• all</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideHTTPRequest - setupProxy(String host, String port)**

Set the proxy host and port.

Additional information about the method that does not belong in the short description.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>String</td>
<td>The proxy host</td>
</tr>
<tr>
<td>port</td>
<td>String</td>
<td>The proxy port</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideHTTPResponse**

Utility methods to perform common functions with Glide HTTP responses.

**GlideHTTPResponse - getAllHeaders()**

Returns a list of all headers.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>List&lt;IHTTPHeader&gt;</td>
<td>A list containing all headers.</td>
</tr>
</tbody>
</table>

GlideHTTPResponse - getBody()

Returns a string containing a body for the HTTP response.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The HTTP response body.</td>
</tr>
</tbody>
</table>

Note: If the stream has already been accessed, it cannot be accessed again. In this case the method will return null.

GlideHTTPResponse - getCookies()

Returns a list containing the cookies for the HTTP response.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>List&lt;String&gt;</td>
<td>A list containing the name and value of each cookie as a string.</td>
</tr>
</tbody>
</table>
**GlideHTTPResponse - getErrorCode()**

Return the current error code.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Integer</td>
</tr>
</tbody>
</table>

**GlideHTTPResponse - getHeader(String name)**

Returns the header specified in the parameter.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

**GlideHTTPResponse - getHeaders()**

Returns a map of headers.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Map</td>
</tr>
</tbody>
</table>
GlideHTTPResponse - getHttpMethod()

Returns the HTTP method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HttpMethod</td>
<td>The HTTP method (GET, POST, etc).</td>
</tr>
</tbody>
</table>

GlideHTTPResponse - getStatusCode()

Returns the last status code.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>Returns the HTTP response status code.</td>
</tr>
</tbody>
</table>

GlideHTTPResponse - haveError()

Returns true if an error has occurred.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if an error has occurred, or false if there are no errors</td>
</tr>
</tbody>
</table>
**GlideImpersonate - Global**

The global GlideImpersonate API enables administrators to pose as another authenticated user for testing purposes.

Use the GlideImpersonate API when you need to act as another user so to test functionality within your application. When impersonating another user, the administrator has access to exactly what the impersonated user would have access to in the system, including the same menus and modules.

⚠️ **Note:** The system records anything the administrator does while impersonating another user as having been done by that user.

**GlideImpersonate - canDebug(String userSysId)**

Verifies whether the specified user can perform debugging on scripts.

In order for a user to be able to debug scripts, they must be on a developer instance. Debugging is not allowed on production instances.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userSysId</td>
<td>String</td>
<td>sys_Id of the user to verify for debugging capability.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True: User is able to debug the application.</td>
</tr>
<tr>
<td>False: User is not able to debug the application.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
userDebug: function(userId) {
  var impUser = new GlideImpersonate();
  impUser.canDebug(userId);
}
```

**GlideImpersonate - canImpersonate(String userSysId)**

Verifies whether the current user can impersonate the specified user.
If the current user is not assigned the admin role, the user to impersonate is inactive, or there are other issues with impersonating the specified user, the method returns "false" and the user cannot be impersonated.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userSysId</td>
<td>String</td>
<td>sys_Id of the user to impersonate</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the current user can impersonate the specified user. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Current user can impersonate the specified user.</td>
</tr>
<tr>
<td></td>
<td>• false: Current user cannot impersonate the specified user.</td>
</tr>
</tbody>
</table>

```javascript
function onlineImpersonate(userSysId) {
  if (!GlideImpersonate().canImpersonate(userSysId)){
    gs.addInfoMessage("No access to impersonate " + userSysId);
  } else {
    GlideImpersonate().impersonate(userSysId);
  }
  return;
}
```

### GlideImpersonate - impersonate(String userSysId)

Sets the user ID for the current administrator to the passed-in user ID, enabling the administrator to act as that user.

When impersonating another user, the administrator has access to exactly what the impersonated user would have access to in the system, including the same menus and modules. Only use this method when testing functionality in an application. Ensure that once you are finished impersonating a user that you call the method again with the administrator sys_Id to stop the impersonation.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userSysId</td>
<td>String</td>
<td>Sys_id of the user to impersonate.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the user that was logged in prior to the impersonation request.</td>
</tr>
</tbody>
</table>

The following example shows how to get the current user object.

```javascript
var user = gs.getUserDisplayName();
gs.print("The current user display name is: " + user);

var impUser = new GlideImpersonate();
impUser.impersonate("62826bf03710200044e0bfc8bcbe5df1");
var user = gs.getUserDisplayName();
gs.print("The impersonated user display name is: " + user);
```

Output:

```
The current user display name is: System Administrator
The impersonated user display name is: Abel Tuter
```

### GlideImpersonate - isImpersonating()

Determines whether the current user is impersonating another user.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True: User is currently impersonating another user.</td>
</tr>
<tr>
<td></td>
<td>False: User is not currently impersonating another user.</td>
</tr>
</tbody>
</table>
function abortOnImpersonate() {
  if (GlideImpersonate().isImpersonating()){
    current.setAbortAction(true);
    gs.addInfoMessage("Transaction canceled due to Impersonation");
  }
  return;
}

**GlideImportLog - Scoped, Global**

Writes log entries directly to the Import Log `[import_log]` table.

A GlideImportLog object consumable by the GlideImportSetTransformer API. This object is not required for the GlideImportSetTransformer API, it enables you to associate import logs with a specific Import Set transformation.

**Related information**

- GlideImportSetRun
- GlideImportSetTable
- GlideImportSetTransformer
- GlideImportSetTransformMap
- GlideTransformLog

**GlideImportLog - error(String message, String source)**

Logs a message of type Error to the Import Log `[import_log]` table.

**GlideImportLog**

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Log message. Maximum length 4000 characters.</td>
</tr>
<tr>
<td>source</td>
<td>String</td>
<td>Optional. Source field value attached to the Import Log record that defines where in the Import Set process this message was logged from. For example, Loading could represent loading step, Cleanup could represent during the cleanup after the transform, and so on.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
var importLog = new GlideImportLog();
importLog.error('Error executing transform');
```

GlideImportLog - getImportRunHistory()

Returns a sys_id of the Import Run record associated with this Import Log.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of an Import Set Run record from the Transform History [sys_import_set_run] table.</td>
</tr>
</tbody>
</table>

```
var importSet = new GlideRecord('sys_import_set');
importSet.name = 'Scripted Import Set';
importSet.short_description = 'Import set from scripted rest api';
importSet.table_name = importSetTableName;
var importSetID = importSet.insert();
var importLog = new GlideImportLog();
var transformer = new GlideImportSetTransformer();
transformer.setLogger(importLog);
transformer.transformAllMaps(importSet);
var importRunSysId = importLog.getImportRunHistory();
```

GlideImportLog - GlideImportLog(GlideImportSetRun importSetRun, String source)

Instantiates a GlideImportLog object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>importSetRun</td>
<td>GlideImportSetRun</td>
<td>Optional. Import Set Run record on which all logs are to be associated with.</td>
</tr>
<tr>
<td>source</td>
<td>String</td>
<td>Optional. Source field value attached to the Import Log record that defines where in the Import Set process this message was logged from. For example, Loading could represent loading step, Cleanup could represent during the cleanup after the transform, and so on.</td>
</tr>
</tbody>
</table>

Example without optional source parameter.

```javascript
var importSetRun = new GlideImportSetRun();
var importLog = new GlideImportLog(importSetRun);
```

Example using optional source parameter.

```javascript
var importSetRun = new GlideImportSetRun();
var importLog = new GlideImportLog(importSetRun, 'Scripted ImportSetTransformer');
```

GlideImportLog - info(String message, String source)

Logs a message of type Info to the Import Log [import_log] table.

GlideImportLog

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Log message. Maximum length 4000 characters.</td>
</tr>
<tr>
<td>source</td>
<td>String</td>
<td>Optional. Source field value attached to the Import Log record that defines where in the Import Set process this message was logged from. For example, Loading could represent loading step, Cleanup could represent during the cleanup after the transform, and so on.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importLog = new GlideImportLog();
importLog.info('Successfully executed transform');
```

**GlideImportLog - setImportRunHistory(String importRunHistory)**

Associates the GlideImportLog object with a specific Import Set Run record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>importRunHistory</td>
<td>String</td>
<td>The sys_id of a record from the Transform History [sys_import_set_run] table.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importRunSysId = '4aa3a8d55ba10010953330ad5981c79f';
var importLog = new GlideImportLog();
importLog.setImportRunHistory(importRunSysId);
```

**GlideImportLog - warn(String message, String source)**

Logs a message of type Warn to the Import Log [import_log] table.

**GlideImportLog**

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Log message. Maximum length 4000 characters.</td>
</tr>
<tr>
<td>source</td>
<td>String</td>
<td>Optional. Source field value attached to the Import Log record that defines where in the Import Set process this message was logged from. For example, Loading could</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>represent loading step, Cleanup could represent during the cleanup after the transform, and so on.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importLog = new GlideImportLog();
importLog.warn('Transform taking longer than expected.');
```

**GlideImportSetRun - Scoped, Global**

Creates an Import Set Run record which can be consumed by the GlideImportSetTransformer API.

**Related information**

- GlideImportLog
- GlideImportSetTable
- GlideImportSetTransformer
- GlideImportSetTransformMap
- GlideTransformLog

**GlideImportSetRun - getImportSetRunSysID()**

Gets the sys_id of the Import Set Run associated with the transformation.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>The sys_id of the Transform Histories [sys_import_set_run] record associated with the transform.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importSetRun = new GlideImportSetRun();
var importSetRunSysId = importSetRun.getImportSetRunSysID();
```

**GlideImportSetRun - GlideImportSetRun(String importSetID)**

Instantiates a GlideImportSetRun object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>importSetID</td>
<td>String</td>
<td>Optional. The sys_id of the import set record listed in the Import Sets [sys_import_set] table. If not set, a new [sys_import_set] record is created and the GlideImportSetRun object represents this record.</td>
</tr>
</tbody>
</table>

```javascript
var importSetRun = new GlideImportSetRun();
```

**GlideImportSetTable - Scoped, Global**

The GlideImportSetTable API provides methods to create temporary Import Set tables.

These methods create the Import Set table using a dynamic, standard naming convention. Tables must have at least one defined column. Modification and deletion of existing Import Set tables is not supported.

A scheduled job named Scripted Import Set Deleter runs every seven days by default. Scripted Import Set Deleter deletes all Import Sets, Transform Maps, Transform Entries, and drops the Import Set Tables associated with the tables created by this API.

You can use the GlideImportSetTable methods in global and scoped scripts. Use the `sn_impex` namespace identifier to create a GlideImportSetTable object.
Related information

GlideImportLog
GlideImportSetRun
GlideImportSetTransformer
GlideImportSetTransformMap
GlideTransformLog

**GlideImportSetTable - GlideImportSetTable(String tableLabel)**

Instantiates a GlideImportSetTable object.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableLabel</td>
<td>String</td>
<td>Label of the Import Set table created upon calling the <code>create()</code> method.</td>
</tr>
</tbody>
</table>

```javascript
var importSetTable = new sn_impex.GlideImportSetTable("temp user table");
```

**GlideImportSetTable - addDateTimeColumn(String columnLabel)**

Creates a GlideDateTime column.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columnLabel</td>
<td>String</td>
<td>Label of the GlideDateTime column to create in the Import Set table.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importSetTable = new sn_impex.GlideImportSetTable("temp user table");
importSetTable.addDateTimeColumn('start date');
var tableStructure = importSetTable.create();
```

**GlideImportSetTable - addStringColumn(String columnLabel, Number length)**

Creates a string column.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columnLabel</td>
<td>String</td>
<td>Label of the string column to create in the Import Set table.</td>
</tr>
<tr>
<td>length</td>
<td>Number</td>
<td>Optional. Maximum column length. Default: 40 characters</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importSetTable = new sn_impex.GlideImportSetTable("temp user table");
importSetTable.addStringColumn('first name', 50);
importSetTable.addStringColumn('last name', 50);
var tableStructure = importSetTable.create();
```

### GlideImportSetTable - create()

Creates the Import Set table.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JSON object in the following format:</td>
</tr>
<tr>
<td></td>
<td>• tableName: String. Database name of the table.</td>
</tr>
<tr>
<td></td>
<td>• columns: Object. Map of table column labels to column names provided by</td>
</tr>
<tr>
<td></td>
<td>the addDateTimeColumn() and addStringColumn() methods.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>column_label</td>
<td>String. Label of the string column in the Import Set table.</td>
</tr>
<tr>
<td>column_name</td>
<td>String. Name of the string column in the Import Set table.</td>
</tr>
</tbody>
</table>

```javascript
// Create Import Set table
var importSetTable = new sn_impex.GlideImportSetTable("temp user table");
importSetTable.addStringColumn('first name', 40);
importSetTable.addStringColumn('last name', 40);
importSetTable.addDateTimeColumn('start date');
var tableStructure = importSetTable.create();

/*
tableStructure = {
"tableName": "imp_staging_table_1417601730000",
"tableLabel": "temp user table",
"columns": {
    "first name": "u_first_name",
    "last name": "u_last_name",
    "start date": "u_start_date"
}
}
*/
var importSetTableName = tableStructure['tableName'];
var columns = tableStructure['columns'];
```

**GlideImportSetTransformer - Scoped, Global**

Creates an Import Set Transformer object used to execute an Import Set Transform.

To execute an Import Set transform:

1. Create an Import Set table using the `GlideImportSetTable` API.
2. Create a Transform Map using the `GlideImportSetTransformMap` API.
3. Create or use an existing Import Set using the GlideRecord API.

4. Execute an Import Set Transform using the GlideImportSetTransformer API.

Once you have created the Import Set record it must be consumed by the GlideImportSetTransformer API in order to trigger the transform.

```javascript
var transformer = new GlideImportSetTransformer();
transformer.transformAllMaps(importSetGr);
```

**Related information**

- GlideImportLog
- GlideImportSetRun
- GlideImportSetTable
- GlideImportSetTransformMap
- GlideRecord
- GlideTransformLog

**GlideImportSetTransformer - GlideImportSetTransformer()**

Instantiates a GlideImportSetTransformer object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var transformer = new GlideImportSetTransformer();
```

**GlideImportSetTransformer - getImportSetRun()**

Gets the Import Set Run object associated with a GlideImportSetTransformer object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>GlideImportSetRun</td>
<td>GlideImportSetRun object created by running a transformation or the argument supplied to the setImportSetRun() method.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importSet = new GlideRecord('sys_import_set');
importSet.short_description = 'Import set from scripted rest api';
importSet.table_name = importSetTableName;
var importSetID = importSet.insert();
var transformer = new GlideImportSetTransformer();
transformer.transformAllMaps(importSet);
var importSetRun = transformer.getImportSetRun();
```

**GlideImportSetTransformer - isError()**

Returns a Boolean value specifying whether or not there was an error during a transformation.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether there is an error.</td>
</tr>
<tr>
<td></td>
<td>• true: Error.</td>
</tr>
<tr>
<td></td>
<td>• false: Success.</td>
</tr>
</tbody>
</table>

```javascript
var importSet = new GlideRecord('sys_import_set');
importSet.short_description = 'Import set from scripted rest api';
importSet.table_name = importSetTableName;
var importSetID = importSet.insert();
var transformer = new GlideImportSetTransformer();
transformer.transformAllMaps(importSet);
if(transformer.isError()) {
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
GlideImportSetTransformer - setImportSetID(String id)

Associates an Import Set record with a specific GlideImportSetTransformer object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>The sys_id of an Import Set record from the Import Sets [sys_import_set] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importSet = new GlideRecord('sys_import_set');
importSet.short_description = 'Import set from scripted rest api';
importSet.table_name = importSetTableName;
var importSetID = importSet.insert();
var transformer = new GlideImportSetTransformer();
transformer.setImportSetID(importSetID);
transformer.transformAllMaps(importSet);
```

GlideImportSetTransformer - setImportSetRun(GlideImportSetRun importSetRun)

Associates an Import Set Run object with a specific GlideImportSetTransformer object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>importSetRun</td>
<td>GlideImportSetRun</td>
<td>GlideImportSetRun object representing the ImportSetRun record to track the transformation history.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importSet = new GlideRecord('sys_import_set');
importSet.short_description = 'Import set from scripted rest api';
importSet.table_name = importSetTableName;
var importSetID = importSet.insert();
var importSetRun = new GlideImportSetRun(importSetID);
var transformer = new GlideImportSetTransformer();
transformer.setImportSetRun(importSetRun);
transformer.transformAllMaps(importSet);
```

**GlideImportSetTransformer - setLogger(GlideImportLog log)**

Associates a GlideImportLog object with a specific GlideImportSetTransformer object.

If this method is not called prior to calling the `transformAllMaps()` method, a GlideImportLog object is created internally on the server side.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>log</td>
<td>GlideImportLog</td>
<td>GlideImportLog object to link to the Import Set history. Once set, any call from the GlideImportLog object is associated with the Import Set run history for that specific transform.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importSet = new GlideRecord('sys_import_set');
importSet.short_description = 'Import set from scripted rest api';
importSet.table_name = importSetTableName;
var importSetID = importSet.insert();
var importLog = new GlideImportLog();
```
```javascript
var transformer = new GlideImportSetTransformer();
transformer.setLogger(importLog);
transformer.transformAllMaps(importSet);
```

**GlidelImportSetTransformer - setMapID(String mapID)**

Associates a Transform Map with a specific GlidelImportSetTransformer object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mapID</td>
<td>String</td>
<td>A sys_id from the Table Transform Maps [sys_transform_map] table.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importSet = new GlideRecord('sys_import_set');
importSet.short_description = 'Import set from scripted rest api';
importSet.table_name = importSetTableName;
var importSetID = importSet.insert();
var transformMapId = '<sys id of transform map>';
var transformer = new GlidelImportSetTransformer();
transformer.setMapID(transformMapId);
transformer.transformAllMaps(importSet);
```

**GlidelImportSetTransformer - setSyncImport(Boolean synchronous)**

Enables running synchronous transformations.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>synchronous</td>
<td>Boolean</td>
<td>Sets transformation mode. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Run transformation synchronously</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Run transformation asynchronously</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importSet = new GlideRecord('sys_import_set');
importSet.short_description = 'Import set from scripted rest api';
importSet.table_name = importSetTableName;
var importSetID = importSet.insert();
var transformer = new GlideImportSetTransformer();
transformer.setSyncImport(True);
transformer.transformAllMaps(importSet);
```

GlideImportSetTransformer - transformAllMaps(GlideRecord importSet, String rowSysId)

Executes the Import Set transformation.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>importSet</td>
<td>GlideRecord</td>
<td>GlideRecord of the import set to execute.</td>
</tr>
<tr>
<td>rowSysId</td>
<td>String</td>
<td>Optional. Restricts transformation to a single record by passing the sys_id of a record from the Import Sets [sys_import_set] table. Default: Transform all rows</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
```javascript
var importSet = new GlideRecord('sys_import_set');
importSet.short_description = 'Import set from scripted rest api';
importSet.table_name = importSetTableName;
var importSetID = importSet.insert();
var transformer = new GlideImportSetTransformer();
transformer.transformAllMaps(importSet);
```

GlideImportSetTransformMap - Scoped, Global

Provides methods to create transform maps and transform entries.

You can use the GlideImportSetTransformMap methods in global and scoped scripts. Use the sn_impex namespace identifier to create a GlideImportSetTransformMap object.

Related information

- GlideImportLog
- GlideImportSetRun
- GlideImportSetTable
- GlideImportSetTransformer
- GlideTransformLog

GlideImportSetTransformMap - addDateTimeTransformEntry(String sourceColumn, String targetColumn, Boolean coalesce, String dateTimeFormat)

Add a GlideDateTime transform entry to a transform map.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>targetColumn</td>
<td>String</td>
<td>Target column from target table to transform to.</td>
</tr>
</tbody>
</table>
| coalesce       | Boolean | Optional. Flag that indicates whether to insert the record into the target table
|                |         | • true: If true and target table contains a record with the same value in this field, the record is updated, that is, not inserted. |
|                |         | • false: Default. Record is inserted into the target table.                  |
| sourceColumn   | String  | Source column from Import Set table to transform from.                      |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dateTimeFormat</td>
<td>String</td>
<td>Optional. GlideDateTime format of the transform, for example, yyyy-MM-dd HH:mm:ss. If empty, the default value is the system date time format.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var transformMap = new sn_impex.GlideImportSetTransformMap("temp user map name", importSetTableName, targetTableName,);
transformMap.addDateTimeTransformEntry("u_start date", "first day", true, "yyyy-MM-dd HH:mm:ss");
var transformMapId = transformMap.create();
```

**GlideImportSetTransformMap - addTransformEntry(String sourceColumn, String targetColumn, Boolean coalesce)**

Adds a string transform entry to a transform map.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>targetColumn</td>
<td>String</td>
<td>Source column from Import Set table to transform from.</td>
</tr>
<tr>
<td>coalesce</td>
<td>String</td>
<td>Target column from target table to transform to.</td>
</tr>
<tr>
<td>sourceColumn</td>
<td>Boolean</td>
<td>Optional. Flag that indicates whether to insert the record into the target table</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: If true and target table contains a record with the same value in this field, the record is updated, that is, not inserted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Default. Record is inserted into the target table.</td>
</tr>
</tbody>
</table>
GlideImportSetTransformMap - create()
Creates a new transform map.

Important: To create a transform map you must have defined at least one transform entry using either `addDateTimeTransformEntry()` or `addTransformEntry()`.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_id of the created transform map.</td>
</tr>
</tbody>
</table>

GlideImportSetTransformMap - GlideImportSetTransformMap(String transformMapName, String importSetTableName, String targetTableName)
Instantiates a GlideImportSetTransformMap object.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>transformMapName</td>
<td>String</td>
<td>Name of the map.</td>
</tr>
<tr>
<td>importSetTableName</td>
<td>String</td>
<td>Name of the Import Set table.</td>
</tr>
<tr>
<td>targetTableName</td>
<td>String</td>
<td>Name of the target table.</td>
</tr>
</tbody>
</table>

```javascript
var transformMap = new sn_impex.GlideImportSetTransformMap("temp user map name", importSetTableName, targetTableName);
```

### GlideList2 (g_list) - Client

GlideList2 is a JavaScript class used to customize (v2) lists.

The variable `g_list` is used to access a specified list object. The `g_list` variable is not available to the related lists form link UI action. It is available to the lists form link UI action.

These methods are used in UI Context Menus and UI Actions.

#### GlideList2 - addFilter(String filter)

Adds a single term to the list query filter.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Query string condition to add.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
g_list.addfilter("active=true");
```

### GlideList2 - get(Object DOMelement)

Returns the GlideList2 object for the list that contains the specified item.
### GlideList2 - `get(String ListID)`

Returns the `GlideList2` object for the list specified.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ListID</td>
<td>String</td>
<td>The list ID for which you want the <code>GlideList2</code> object.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The GlideList2 object or null if not found.</td>
</tr>
</tbody>
</table>

```javascript
function assignLabelActionViaLookupModal(tableName, listId) {
  var list = GlideList2.get(listId);
  if (!list)
    return;

  assignLabelViaLookup(tableName, sysIds, list.getView());
}
```

### GlideList2 - `getChecked()`

Returns a comma-separated list of the sys_ids for the items that are checked in the list.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Comma-separated list of the sys_ids for the items that are checked in the list. Does not check to determine that the items returned are allowed to be executed.</td>
</tr>
</tbody>
</table>

```javascript
function removeLabelActionViaLookupModal(tableName, listId) {
    var list = GlideList2.get(listId);
    if (!list)
        return;
    var sysIds = list.getChecked();
    if (!sysIds)
        return;
    removeLabelViaLookup(tableName, sysIds);
}
```

**GlideList2 - getFixedQuery()**

Returns the sysparm_fixed query.

A fixed query is the part of the query that cannot be removed from the breadcrumb (i.e., it is fixed for the user). It is specified by including a `sysparm_fixed_query` parameter for the application module.
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The fixed query string for the list.</td>
</tr>
</tbody>
</table>

```javascript
var list = GlideList2.get(container.readAttribute('list_id'));
var filter = this._getFilter(element);
var fixedQuery = list.getFixedQuery();
if (fixedQuery)
    filter += '^' + filter;
```

**GlideList2 - getGroupBy()**

Returns the field or comma-separated list of fields that are used to group the list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>none</td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The field or comma-separated list of fields that are used to group the list.</td>
</tr>
</tbody>
</table>

```javascript
function runFilterV2Lists(name, filter) {
    var list = GlideList2.get(name);
    if (list) {
        var groupBy = list.getGroupBy();
        if (groupBy)
            filter += '^' + groupBy;

        list.setFilterAndRefresh(filter);
    }
}
```

**GlideList2 - getListName()**

Returns the name of the list, which is usually the table name.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The list name (usually the table name).</td>
</tr>
</tbody>
</table>

```
var list = GlideList2.get(name);
var listName = list.getListName();
```

GlideList2 - getOrderBy()

Returns the first field used to order the list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The field used for order, or a blank.</td>
</tr>
</tbody>
</table>

```
var list = GlideList2.get(listId);
if (!list)
  return;
var orderBy = list.getOrderBy();
```

GlideList2 - getParentTable()

Returns the name of the parent table for a related list (the table associated with the form).
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The parent table name.</td>
</tr>
</tbody>
</table>

for (var id in GlideLists2) {
  var list = GlideLists2[id];
  if (list.getName() == listTableName && list.getContainer() == tableName)
    return list.getContainer();
}

GlideList2 - getQuery(Boolean orderBy, Boolean groupBy, Boolean fixed, Boolean all)

Returns the encoded query string for the list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>orderBy</td>
<td>Boolean</td>
<td>(Optional) If true, includes the orderBy in the encoded query string.</td>
</tr>
<tr>
<td>groupBy</td>
<td>Boolean</td>
<td>(Optional) If true, includes the groupBy in the encoded query string.</td>
</tr>
<tr>
<td>fixed</td>
<td>Boolean</td>
<td>(Optional) If true, includes fixed query in the encoded query string.</td>
</tr>
<tr>
<td>all</td>
<td>Boolean</td>
<td>(Optional) If true, includes orderBy, groupBy, and fixed query.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Encoded query string for the list.</td>
</tr>
</tbody>
</table>
```
var list = GlideList2.get(this.listID);
var ajax = new GlideAjax("AJAXJellyRunner", "AJAXJellyRunner.do");
   ajax.addParam("sysparm_query_encoded", list.getQuery({groupby: true, orderby: true}));
   ajax.addParam("sysparm_table", list.getTableName());
   ajax.addParam("sysparm_view", list.getView());
```

**GlideList2 - getRelated()**

Returns the related list field that associates the related list to the parent form.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Field that connects the list to the parent form.</td>
</tr>
</tbody>
</table>

```
var list = GlideList2.get(name);
var related = list.getRelated();
if (related)
   ajax.addParam("sysparm_is_related_list", "true");
```

**GlideList2 - getTableName()**

Returns the table name for the list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Returns the table name for the list.</td>
</tr>
</tbody>
</table>

GlideList2.getListsForTable = function(table) {
    var lists = [];
    for (var id in GlideLists2) {
        var list = GlideLists2[id];
        if (list.getTableName() == table)
            lists.push(list);
    }
    return lists;
}

GlideList2 - getView()

Returns the view used to display the list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>none</td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the view.</td>
</tr>
</tbody>
</table>

function assignLabelActionViaLookupModal(tableName, listId) {
    var list = GlideList2.get(listId);
    if (!list)
        return;
    assignLabelViaLookup(tableName, sysIds, list.getView());
}
GlideList2 - getTitle()

Returns the list title.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The list title.</td>
</tr>
</tbody>
</table>

```javascript
var list = GlideList2.get(name);
var listTitle = list.getTitle();
```

GlideList2 - isUserList()

Returns true if the list has been personalized by the user by choosing the list mechanic and changing the list layout.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the list layout has been changed.</td>
</tr>
</tbody>
</table>

```javascript
var list = GlideList2.get(listId);
if (!list)
  return;
if (list.isUserList())
  var tableName = list.getTableName();
```
GlideList2 - refresh(Number firstRow, String additionalParms)

Refreshes the list. The **orderBy** part of the list filter is ignored so that the list uses its natural ordering when it is refreshed.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>firstRow</td>
<td>Number</td>
<td>(Optional) The first row to appear in the list. If not specified, the first row of the current is used.</td>
</tr>
<tr>
<td>additionalParms</td>
<td>String</td>
<td>(Optional) name-value pairs that are submitted with the list refresh request.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
$timeout(function() {
  if (GlideList.lists) {
    var list = GlideList.get(name);
    if (list) {
      if (sortBy) {
        if (sortDirection == 'ASC')
          list.sort(sortBy);
        else
          list.sortDescending(sortBy);
      }
      list.refresh();
    }
  }
})
```

GlideList2 - refreshWithOrderBy(Number firstRow, String description)

Refreshes the list. The **orderBy** part of the list filter is included if it is specified for the list.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>firstRow</td>
<td>Number</td>
<td>(Optional) The first row to appear in the list.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>(Optional) name=value pairs that are submitted with the list refresh request.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
ga.getXML(function(serverResponse) {
  var response = serverResponse.responseXML.getElementsByTagName("response")[0];
  if (response) {
    var list = GlideList2.getByName("backlog_stories");
    list.refreshWithOrderBy();
    var status = response.getAttribute('status');
    $j('html, body').animate({scrollTop: $j("#"+data.record.sys_id).offset().top}, 500);
    if (status == 'failure') {
      alert('${gs.getMessage("Story cannot be created. Team is not associated with any project.")}');
    }
  }
});
```

GlideList2 - setFilter(String filter)

Sets the encoded query string for the list, ignoring the `orderBy` and `groupBy` parts of the query string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Encoded query string.</td>
</tr>
</tbody>
</table>
null

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
list = GlideList2.get($(side+"ContentDivRelease").select(".list_div")[0].getAttribute("id"));
if (list) {
    list.setFilter("active=true");
    list.refresh(1);
}
```

**GlideList2 - setFilterAndRefresh(String filter)**

Sets the encoded query string for the list, including the `orderBy` and `groupBy` if specified, and then refreshes the list using the new filter.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Encoded query string.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
function updateListFilter(projectID) {
    var list = GlideList2.getByName("backlog_stories");
    var fixedQuery = $('hdn_additional_filters').value;
    if(!projectID) {
        list.setFilterAndRefresh(fixedQuery + `ORDERBYteam_index`);
        list.setOrderBy("team_index");
    }
}
```

**GlideList2 - setFirstRow(Number rowNum)**

Sets the first row that appears in the list when the list is refreshed.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rowNum</td>
<td>Number</td>
<td>Row number of the first row to display.</td>
</tr>
</tbody>
</table>

Returs

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var nextRow = 0;
var rowsPerPage = 20;
var list = GlideList2.get(listId);
if (!list)
    return;
list.setFirstRow(nextRow);
nextRow = nextRow + rowsPerPage;
```

GlidleSt2 - setGroupBy(String groupBy)

Sets the list **groupBy** criteria for a single field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupBy</td>
<td>String</td>
<td>Optional. The <strong>groupBy</strong> criteria for the list.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
function runContextAction(listId) {
    var g_list = GlideList2.get(listId);
    g_list.setGroupBy('');
    g_list.refresh(1);
}
```
GlideList2 - setOrderBy(String orderBy)
Sets the orderBy criteria for the list.

For a single order by field use orderBy field or orderByDescField. For multiple fields, use orderByField1^orderByField2^orderByField3. orderBy specifies ascending order and orderByDesc specifies descending. These prefix strings are optional. If not specified orderBy is assumed.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>orderBy</td>
<td>String</td>
<td>Single or multiple order by fields.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

updateOrderBy: function(orderBy){
    var list = GlideList2.get(this.listID);
    if (list)
        list.setOrderBy(orderBy);
};

GlideList2 - setRowsPerPage(Number rows)
Sets the number of rows per page to display.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rows</td>
<td>Number</td>
<td>The number of rows to display</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
link: function(scope) {
    var list = GlideList2.get(scope.listId);
    list.setRowsPerPage(scope.maxRows);
    list.setFilterAndRefresh(scope.tableQuery);
}

**GlideList2 - showHideGroups(Boolean showFlag)***

Shows or hides all the groups within the list and saves the current collapsed/expanded state of the groups as a user preference.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showFlag</td>
<td>Boolean</td>
<td>If <strong>true</strong>, shows the groups within the list.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

function showHideAllGroups(showFlag) {
    var list = GlideList2.get(listId);
    if (!list)
        return;
    list.showHideGroups(showFlag);
}

**GlideList2 - showHideList(Boolean showFlag)***

Displays or hides the list and saves the current collapsed/expanded state of the list as a user preference.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showFlag</td>
<td>Boolean</td>
<td>If <strong>true</strong>, displays the list.</td>
</tr>
</tbody>
</table>
### GlideList2.toggleAll

```javascript
GlideList2.toggleAll = function(expandFlag) {
  for (var id in GlideLists2) {
    var list = GlideLists2[id];
    list.showHideList(expandFlag);
  }
}
```

### GlideList2 - sort(String field)

Sorts the list in ascending order and saves the choice.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Specifies the field used to sort the list.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
**GlideList2 - sortDescending(String field, Number amount)**

Sorts the list in descending order and saves the choice.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Specifies the field used to sort the list.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
$timeout(function() {
    if (GlideList.lists) {
        var list = GlideList.get(name);
        if (list) {
            if (sortBy) {
                if (sortDirection == 'ASC')
                    list.sort(sortBy);
                else
                    list.sortDescending(sortBy);
            }
            list.refresh();
        }
    }
})
```

**GlideList2 - toggleList()**

Toggles the display of the list and saves the current collapsed/expanded state of the list as a user preference.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = GlideList2.get(listId);
if (!list)
    return;
list.toggleList();
```

**GlideList2 - toggleListNoPref()**

Toggles the display of the list but does not save the current collapsed/expanded state of the list as a user preference.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = GlideList2.get(listId);
if (!list)
    return;
list.toggleListNoPref();
```

**GlideListV3 (g_list) - Client**

Use GlideListV3 to manipulate lists.

You access the GlideListV3 methods by using the g_list global object. These methods are used in UI context menus and UI actions. The g_list object is not available for related lists on the form link UI action.

**Note:**

This API is no longer supported. Consider using the GlideList2() API instead.
**GlideListV3 - addFilter(String filter)**  
Adds a single term to the list query filter.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Query string condition to add.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideListV3 - get(String listId)**  
Returns the GlideList object for specified list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>listId</td>
<td>String</td>
<td>The list name.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The GlideList object for the specified list, or null if not found.</td>
</tr>
</tbody>
</table>

**GlideListV3 - get(Object DomElement)**  
Returns the GlideList object for the specified DOM element.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DomElement</td>
<td>Object</td>
<td>The DOM element ID for which you want the GlideList object.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The GlideList object for the specified DOM element. Returns null if the DOM element is not found.</td>
</tr>
</tbody>
</table>

GlideListV3 - getChecked()

Returns a comma-separated list of sys_ids for checked items in the list. Does not return items that are not allowed to be executed.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Comma-separated list of the sys_ids for checked items in the list. Does not return items that are not allowed to be executed.</td>
</tr>
</tbody>
</table>

GlideListV3 - getFixedQuery()

Returns the sysparm_fixed query.

The fixed query is the part of the query that cannot be removed from the breadcrumb (i.e., it is fixed for the user). It is specified by including a sysparm_fixed_query parameter for the application module.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The fixed query string for the list.</td>
</tr>
</tbody>
</table>
GlideListV3 - getFormTarget()
Returns the form's target attribute.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The form's target attribute.</td>
</tr>
</tbody>
</table>

GlideListV3 - getGroupBy()
Returns the field or comma-separated list of fields that are used to group the list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The field or comma-separated list of fields used to group the list.</td>
</tr>
</tbody>
</table>

GlideListV3 - getListName()
Returns the name of the list, which is usually the table name.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>The list name.</td>
<td></td>
</tr>
</tbody>
</table>

GlideListV3 - getOrderBy()

Returns the first field used to order the list.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

GlideListV3 - getParentTable()

Returns the name of the parent table (the table associated with the form).

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

GlideListV3 - getQuery(Object options)

Returns the encoded query string for the list.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>The options can be one or more of the following.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• orderby - include ORDERBY in the query</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• groupby - include GROUPBY in the query</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• fixed - include sysparm_fixed_query in the query</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• all - include all the options in the query</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Encoded query string for the list.</td>
</tr>
</tbody>
</table>

**GlideListV3 - getReferringUrl()**

Returns the referring URL.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Returns the parent form's URL, or '*' if there is no parent form.</td>
</tr>
</tbody>
</table>

**GlideListV3 - getRelated()**

Returns the related list field that associates the related list to the parent form.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Field that connects the list to the parent form.</td>
</tr>
</tbody>
</table>

**GlideListV3 - getRelatedListType()**

Returns the related list type.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The relationship table type.</td>
</tr>
</tbody>
</table>

**GlideListV3 - getRelationshipId()**

Returns the relationship record id, if this is type REL related list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_id of the relationship record.</td>
</tr>
</tbody>
</table>

**GlideListV3 - getRowCount()**

Returns the number of rows returned by the query.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of rows returned by the query.</td>
</tr>
</tbody>
</table>

**GlideListV3 - getRowsPerPage()**

Returns the number of rows to be displayed on a page.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of rows to be displayed on a page.</td>
</tr>
</tbody>
</table>

**GlideListV3 - getTableName()**

Returns the table name of the list.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The list's table name.</td>
</tr>
</tbody>
</table>
GlideListV3 - getTitle()

Returns the list title.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The list title.</td>
</tr>
</tbody>
</table>

GlideListV3 - getView()

Returns the view used to display the list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the view</td>
</tr>
</tbody>
</table>

GlideListV3 - isUserList()

Returns true if the list has been personalized by the user.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Boolean</td>
<td>True if the list layout has changed.</td>
<td></td>
</tr>
</tbody>
</table>

**GlideListV3 - refresh(Number firstRow, Object additionalParams)**
Refreshes the list. The orderBy part of the list filter is ignored so that the list's natural ordering is used.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>firstRow</td>
<td>Number</td>
<td>(Optional) The first row to display in the list. If not specified, the list's current first row is used.</td>
</tr>
<tr>
<td>additionalParams</td>
<td>Object</td>
<td>(Optional) Name- value pairs that are submitted with the list refresh request.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideListV3 - refreshWithOrderBy(Number firstRow, Object additionalParams)**
Refreshes the list using the orderBy fields.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>firstRow</td>
<td>Number</td>
<td>(Optional) The first row to display in the list. If not specified, the list's current first row is used.</td>
</tr>
<tr>
<td>additionalParams</td>
<td>Object</td>
<td>(Optional) Name- value pairs that are submitted with the list refresh request.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideListV3 - setFilter(String filter, Boolean saveOrderBy, Boolean saveGroupBy)

Sets the encoded query string for the list ignoring the orderBy and groupBy parts of the query string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>An encoded query string.</td>
</tr>
<tr>
<td>saveOrderBy</td>
<td>Boolean</td>
<td>The default is false. When true uses the orderBy part of the query.</td>
</tr>
<tr>
<td>saveGroupBy</td>
<td>Boolean</td>
<td>The default is false. When true uses the groupBy part of the query.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideListV3 - setFilterAndRefresh( String filter)

Sets the encoded query string for the list, and then refreshes the list using the new filter.

This preserves the groupby and orderby parameters.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Encoded query string.</td>
</tr>
</tbody>
</table>
GlideListV3 - setFirstRow(Number firstRow)

Sets the first row to be displayed when the list is refreshed.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>firstRow</td>
<td>Number</td>
<td>The row number in the list.</td>
</tr>
</tbody>
</table>

GlideListV3 - setFormTarget(String target)

Specifies where to display the response from the form.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>String</td>
<td>The form.target attribute value to use.</td>
</tr>
</tbody>
</table>

GlideListV3 - setGroupBy(String groupBy)

Sets the `groupBy` criteria for the list, for a single field or multiple fields.
For a single field, use field or groupByField. The groupBy prefix is optional. For multiple fields use field1^field2^field3 or groupByField1^groupByField2^groupByField3.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>groupBy</td>
<td>The group by criteria for the list.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideListV3 - setOrderBy(String orderBy)**

Sets the orderBy criteria for the list.

For a single order by field use orderBy field or orderByDescField. For multiple fields, use orderByField1^orderByField2^orderByField3. orderBy specifies ascending order and orderByDesc specifies descending. These prefix strings are optional. If not specified orderBy is assumed.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>orderBy</td>
<td>String</td>
<td>Single or multiple order by fields.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideListV3 - setReferringUrl(String url)**

Sets the parent form referring url.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>The parent form’s URL</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### GlideListV3 - setRowsPerPage(Number numRows)

Set the number of rows to display on a page.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numRows</td>
<td>Number</td>
<td>The number of rows to display on a page.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### GlideListV3 - showHideGroups(Boolean showFlag)

Displays or hides all of the groups within the list and saves the current collapsed/expanded state of the groups as a user preference.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showFlag</td>
<td>Boolean</td>
<td>When true, displays the groups within the list.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
**GlideListV3 - showHideList(Boolean showFlag)**
Displays or hides the list and saves the current collapsed/expanded state of the list as a user preference.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showFlag</td>
<td>Boolean</td>
<td>When true, displays the list.</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideListV3 - sort(String field)**
Sort the list in ascending order.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>The field to be used to sort the list.</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideListV3 - sortDescending(String field)**
Sorts the list in descending order.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>The field used to sort the list.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
GlideListV3 - toggleList()
Toggles the list display between collapsed and expanded, and saves the state as a user preference.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

GlideListV3 - toggleListNoPref()
Toggles the list display between collapsed and expanded, but does not save the state as a user preference.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

GlideListProperties - Global
Create a list and set list properties. For example, define whether a list has a filter, breadcrumbs, and search.
Use GlideListProperties in global server-side scripts. To use this class, instantiate a GlideListProperties object using the constructor.

For an example of this class in the base system, configure a list and select All. The tabbed list of options uses the personalize_all UI page and personalize_all_list UI macro to set list properties such as title, context menu, and breadcrumbs.

GlideListProperties - GlideListProperties()

Instantiates a GlideListProperties object.

```
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

var list = new GlideListProperties();
```

GlideListProperties - getListID()

Returns the unique ID for a list.

```
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

var list = new GlideListProperties();
var getID = list.getListID();
gs.print(getID);

Output: 3519f77ad95f5700964f387107a8a394
```

GlideListProperties - getName()

Returns the name of the list.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the list.</td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var setName = list.setListName("my custom list");
var getName = list.getListName();
gs.print(getName);
```

Output: my custom list

**GlideListProperties - getTitle()**

Get the title of a list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The title of the list.</td>
</tr>
</tbody>
</table>

Optional example explanation

```javascript
var list = new GlideListProperties();
var title = list.setTitle("My title");
var getTitle = list.getTitle();
gs.print(getTitle);
```

Output: My title
** GlideListProperties - hasActions() **

Returns whether or not the **Actions on select rows** option is enabled for a list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns <strong>true</strong> if the actions option is enabled for a list.</td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var actions = list.setHasActions(true);
var hasActions = list.hasActions();
gs.print(hasActions);
```

Output: true

** GlideListProperties - hasBottomNav() **

Returns whether or not a list has navigation at the bottom.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>If returns <strong>true</strong> the list has bottom navigation.</td>
</tr>
</tbody>
</table>

```javascript
var lp = new GlideListProperties();
var Nav = lp.setHasBottomNav(true);
var hasNav = lp.hasBottomNav();
gs.print(hasNav);
```

Output: true

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**GlideListProperties - hasBottomVCR()**

Returns whether or not the page navigation controls appear in the footer of a list.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>If true the page navigation controls appear in the footer of a list</td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var vcr = list.setHasBottomVCR(true);
var hasVCR = list.hasBottomVCR();
gs.print(hasVCR);
```

Output: true

**GlideListProperties - hasFilter()**

Returns whether or not a list has a filter.

The filter property is a parent of the breadcrumbs property. If the filter property is listed as false and the breadcrumb is listed as true, `hasFilter()` still returns true because the child property is marked as true.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>If true a filter icon appears with the list, or the breadcrumb property is listed as true. If false both the filter property and the breadcrumb property are marked as false.</td>
</tr>
</tbody>
</table>
var list = new GlideListProperties();
var filter = list.setHasFilter(true);
var breadcrumbs = list.setHasBreadcrumbs(true);
var hasFilter = list.hasFilter();
gs.print(hasFilter);

Output: true

GlideListProperties - hasHeader()
Returns whether or not a list has a header.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns <strong>true</strong> if a list has a header.</td>
</tr>
</tbody>
</table>

var list = new GlideListProperties();
var header = list.setHasHeader(true);
var hasHeader = list.hasHeader();
gs.print(hasHeader);

Output: true

GlideListProperties - hasHeaderContextMenu()
Returns whether or not a header context menu is enabled for a list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Boolean</td>
<td>If true a context menu displays next to each column header in a list.</td>
</tr>
</tbody>
</table>

```
var list = new GlideListProperties();
var header = list.setHeaderContextMenu(true);
var hasHeader = list.getHeaderContextMenu();
gs.print(hasHeader);
```

Output: true

**GlideListProperties - hasListMechanic()**

Returns whether list personalization is enabled for a list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Boolean</td>
<td>If true the list mechanic is enabled for a list and the Personalize List</td>
</tr>
<tr>
<td></td>
<td></td>
<td>icon appears on the page.</td>
</tr>
</tbody>
</table>

```
var list = new GlideListProperties();
var mechanic = list.setHasListMechanic(true);
var hasMechanic = list.hasListMechanic();
gs.print(hasMechanic);
```

Output: true

**GlideListProperties - hasPopup()**

Returns whether or not a list can have popup windows.
GlideListProperties - hasPopup()

Returns true if the list allows popups.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the list allows popups.</td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var popup = list.setHasPopup(true);
var hasPopup = list.hasPopup();
gs.print(hasPopup);
```

Output: true

GlideListProperties - hasRowContextMenu()

Returns whether or not rows in a list have a context menu.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>If true a list row can have a context menu.</td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var contextMenu = list.setHasRowContextMenu(true);
var hasContextMenu = list.hasRowContextMenu();
gs.print(hasContextMenu);
```

Output: true

GlideListProperties - hasSearch()

Returns whether or not the search bar is enabled for a list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>If true a list row can have a context menu.</td>
</tr>
</tbody>
</table>
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>If true the search bar appears in the header of a list.</td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var search = list.setHasSearch(true);
var hasSearch = list.hasSearch();
gs.print(hasSearch);
```

Output: true

**GlideListProperties - hasTitle()**

Returns whether or not the list title appears in the list header.

The title context menu is a child property of title. If `setHasTitleContextMenu` is set to true, `hasTitle` also returns true, even if `setHasTitle` is set to false.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>If true the list title appears in the list header.</td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var title = list.setHasTitle(true);
var contextMenu = list.setHasTitleContextMenu(true); 
var hasTitle = list.hasTitle();
gs.print(hasTitle);
```

Output: true
GlideListProperties - hasTitleContextMenu()

Returns whether a context menu appears in a list header.

The title context menu is a child property of title. If `setHasTitleContextMenu` is set to true, `hasTitle` also returns true, even if `setHasTitle` is set to false.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Boolean</td>
</tr>
</tbody>
</table>

Optional example explanation

```javascript
var list = new GlideListProperties();
var contextMenu = list.setHasTitleContextMenu(true);
var hasContextMenu = list.hasTitleContextMenu();
gs.print(hasContextMenu);
```

Output: true

GlideListProperties - hasTopVCR()

Returns whether or not the page navigation controls appear in the header of a list.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Boolean</td>
</tr>
</tbody>
</table>
var list = new GlideListProperties();
var vcr = list.setHasTopVCR(true);
var hasVCR = list.hasTopVCR();
gs.print(hasVCR);

Output: true

**GlideListProperties - isOmitFilter()**

Returns whether or not the omit filter option has been selected.

The ListControl omit flags take precedence in that if they are set, they negate the setting of their corresponding flag. For example, if the show filter flag has been set to true, but the ListControl omit filter is true, then checking hasFilter returns false.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Boolean</td>
</tr>
</tbody>
</table>

var lp = new GlideListProperties();
var omitFilter = lp.isOmitFilter();
gs.print(omitFilter);

Output: false

**GlideListProperties - isSaveFilterHidden()**

Returns whether the Save Filter button is hidden in the condition builder.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>If <strong>true</strong> the Save Filter button is hidden in the condition builder.</td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var SaveFilter = list.setSaveFilterHidden(true);
var hasSaveFilter = list.isSaveFilterHidden();
gs.print(hasSaveFilter);
```

Output: true

**GlideListProperties - isShowLinks()**

Returns whether or not a list shows links.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var links = list.setShowLinks(true);
var hasLinks = list.isShowLinks();
gs.print(hasLinks);
```

Output: true

**GlideListProperties - isToggleHeader()**

Returns whether or not toggling the header columns is available for a list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>If <strong>true</strong> users can show or hide the column headers for a table.</td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var toggle = list.setToggleHeader(true);
var hasToggle = list.isToggleHeader();
gs.print(hasToggle);
```

Output: true

### GlideListProperties - setCanChangeView(Boolean onOff)

Determine whether the user can change the view for the list.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If <strong>false</strong> users cannot change the list view. By default, changing views is enabled.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var changeView = list.setCanChangeView(true);
```

### GlideListProperties - setCanGroup(Boolean onOff)

Determine whether users can group items in a list.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If <strong>false</strong>, the group by option does not appear in the column context menu. By default the group by option appears in the list context menu.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### GlideListProperties - setCanGroup(Boolean onOff)

Determine whether the sort option is available in a list.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If false, the sort option does not appear in column list context menu, and users cannot click the column title to change the order of the list.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var canGroup = list.setCanGroup(true);
```

### GlideListProperties - setCanSort(Boolean onOff)

Determine whether the sort option is available in a list.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If false, the sort option does not appear in column list context menu, and users cannot click the column title to change the order of the list.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var canSort = list.setCanSort(true);
```

### GlideListProperties - setContextMenus(Boolean onOff)

Displays or hides all of the available context menus for a list.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If set to true displays the title context menu, header context menu, and list context menu for a list.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var contextMenus = list.setContextMenus(true);
```
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var context = list.setContextMenus(true);
```

**GlideListProperties - setHasActions(Boolean)**

Determine whether the Actions on select rows options display at the bottom of a list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If true displays action options for a list.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var actions = list.setHasActions(true);
```

**GlideListProperties - setHasBottomNav(Boolean onOff)**

Determine whether the navigation actions at the bottom of a list are hidden or not.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>When true adds navigation to the bottom of a list.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var lp = new GlideListProperties();
var bottom = lp.setHasBottomNav(true);
```

**GlideListProperties - setHasBreadcrumbs(Boolean onOff)**

Determine whether or not breadcrumbs appear at the top of a list.

Breadcrumbs are a child of filters. To hide breadcrumbs completely, you need to also set the filter to false.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If true, breadcrumbs appear at the top of a list.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var breadcrumbs = list.setHasBreadcrumbs(true);
```

**GlideListProperties - setHasBottomVCR(Boolean onOff)**

Determine whether the first page, last page, next page, and previous page buttons appear at the bottom of the list.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If true, the first page, last page, next page, and previous page buttons appear at the bottom of the list.</td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```java
var list = new GlideListProperties();
var bottomVCR = list.setHasBottomVCR(true);
```

### GlideListProperties - setHasFilter(Boolean onOff)
Determine whether or not the filter displays as part of a list.

The filter is a parent of breadcrumbs. To remove the filter, you need to set both the filter and the breadcrumbs to false.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>onOff</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

```java
var list = new GlideListProperties();
var filter = list.setHasFilter(true);
```

### GlideListProperties - setHasHeader(Boolean onOff)
Determine whether or not a list displays a header.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>onOff</td>
</tr>
</tbody>
</table>
### GlideListProperties - setHasHeaderContextMenu(Boolean onOff)

Determine whether or not the context menu appears next to each column in a list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If true the context menu appears next to each column in a list.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var header = list.setHasHeader(true);
```

### GlideListProperties - setHasListMechanic(Boolean onOff)

Determine whether or not a list has the option for personalization.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If true the list mechanic is enabled and the Personalize List icon appears on the page.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var actions = list.setHasHeaderContextMenu(true);
```
GlideListProperties - setHasPopup(Boolean onOff)

Determine whether the list has a popup or modal window.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If true the list can have popup windows.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideListProperties - setHasRowContextMenu(Boolean onOff)

Determines whether or not list rows have a context menu.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>When true list rows can have a context menu.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var list = new GlideListProperties();
var contextMenu = list.setHasRowContextMenu(true);

**GlideListProperties - setHasSearch(Boolean onOff)**
Determine whether search appears for a list.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If <strong>true</strong> the search bar appears in the list header.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var list = new GlideListProperties();
var search = list.setHasSearch(true);

**GlideListProperties - setHasTitle(Boolean onOff)**
Determine whether the list title appears in the header.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If <strong>true</strong> the title of the list appears in the list header.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var list = new GlideListProperties();
var title = list.setHasTitle(true);

**GlideListProperties - setHasTitleContextMenu(Boolean onOff)**
Determine whether or not a list has a context menu in the header.
### GlideListProperties - setHasTitleContextMenu(Boolean onOff)

Determine whether or not a list has the context menu appears next to the list title in the header.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If true the context menu appears next to the list title in the header</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var contextMenu = list.setHasTitleContextMenu(true);
```

### GlideListProperties - setHasTopVCR(Boolean onOff)

Determine whether or not a list has the page navigation controls in the list header.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If true the page navigation controls appear in the header of a list</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var vcr = list.setHasTopVCR(true);
```

### GlideListProperties - setHideRows(Boolean onOff)

Determine whether rows are visible in a list.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If true all of the rows are hidden for a list.</td>
</tr>
</tbody>
</table>

**GlideListProperties - setSaveFilterHidden(Boolean onOff)**

Determine whether the Save Filter button appears in the condition builder.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If true the Save Filter button is hidden.</td>
</tr>
</tbody>
</table>

**GlideListProperties - setListID(String ID)**

Set the unique ID for a list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unique ID</td>
<td>String</td>
<td>The unique ID for the list.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var setID = list.setListID("a9dd1483d99f5700964f387107a8a3ec");
var getID = list.getListID();
gs.print(getID);
```

Output: a9dd1483d99f5700964f387107a8a3ec

**GlideListProperties - setListName(String name)**

Defines a name for the list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the list.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var name = list.setListName("my custom list");
```

**GlideListProperties - setShowLinks(Boolean onOff)**

Whether or not a list includes links.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If true list includes related links.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var links = list.setShowLinks(true);
```

GlideListProperties - setTitle(String title)

Defines the list title.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>String</td>
<td>Title for the list.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var title = list.setTitle("My title");
```

GlideListProperties - setToggleHeader(Boolean onOff)

Determine whether users can show or hide column headers for a table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If <strong>true</strong> an icon appears in the header that allows users to show or hide the column headers.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
```javascript
var list = new GlideListProperties();
var toggle = list.setToggleHeader(true);

GlideListProperties - setVCR(Boolean onOff)
Determine whether the first page, last page, next page, and previous page buttons appear at the top and bottom of the list.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOff</td>
<td>Boolean</td>
<td>If false, the list does not have any of the page navigation buttons for a list.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var list = new GlideListProperties();
var VCR = list.setVCR(true);

GlideLocale - Scoped
GlideLocale provides information about display information for the local instance.
There is no constructor for a GlideLocale object. Use the get() method to get a GlideLocale object.

Scoped GlideLocale - get()
Returns the GlideLocale object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## GlideLocale

### GlideLocale

**Description**

The GlideLocale object.

```javascript
var locale = GlideLocale.get();
```

### Scoped GlideLocale - getDecimalSeparator()

**Returns**

**Type**

String

**Description**

The decimal separator.

```javascript
var locale = GlideLocale.get();
var decimalSeparator = locale.getDecimalSeparator();
gs.info( "The decimal separator is " + decimalSeparator);
```

**Output:**

The decimal separator is .

### Scoped GlideLocale - getGroupingSeparator()

**Returns**

**Type**

String

**Description**

The grouping separator.

```javascript
var locale = GlideLocale.get();
var groupingSeparator = locale.getGroupingSeparator();
gs.info( "The grouping separator is " + groupingSeparator);
```
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The grouping separator.</td>
</tr>
</tbody>
</table>

```javascript
var locale = GlideLocale.get();
var groupingSeparator = locale.getGroupingSeparator();
gs.info( "The grouping separator is " + groupingSeparator);
```

Output:

```
The grouping separator is ,
```

**GlideMenu (g_menu and g_item) - Client**

GlideMenu methods are used in UI Context Menus, in the onShow scripts to customize UI Context Menu items.

There is no constructor for the GlideMenu class. Access GlideMenu methods using the g_menu global object.

- **g_menu** is the UI Context Menu that is about to be shown. The onShow script can make changes to the appearance of the menu before it is displayed using these methods.

- **g_item** is the current UI Context Menu item that is about to be shown. It is used in several of the g_menu methods to specify an item.

**GlideMenu - clearImage(GlideMenuItem item)**

Clears the image for an item.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>GlideMenuItem</td>
<td>Specifies the item to have its image removed from display.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
// Clears the image for an item
...
g_menu.clearImage(g_item);
```
GlideMenu - clearSelected()
Clears any selection images from items in the menu.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideMenu - getItem(String itemID)
Returns a menu item by item ID.

It can be necessary to find an item in a menu so that it can be changed before being displayed. Each menu item may be assigned a unique ID when the menu item is created (either from a UI Context Menu entry or from the addAction() method in the Dynamic Script Action).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>itemID</td>
<td>String</td>
<td>Specifies the item to be returned.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideMenuItem</td>
<td>The menu item</td>
</tr>
</tbody>
</table>

GlideMenu - setDisabled(GlideMenuItem item)
Disables a menu item so that it cannot be selected. The disabled menu item is displayed in a lighter color (grayed out) to indicate it is disabled.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>GlideMenuItem</td>
<td>The item to be disabled.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
g_menu.setDisabled(g_item);
```

**GlideMenu - setEnabled(GlideMenuItem item)**

Enables the specified menu item.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>GlideMenuItem</td>
<td>The item to be enabled.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
g_menu.setEnabled(g_item);
```

**GlideMenu - setHidden(GlideMenuItem item)**

Hides the specified menu item.

When hiding menu items, the separator bars are not adjusted, so it is possible to end up with the menu showing two separators in a row.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>GlideMenuItem</td>
<td>The item to be hidden.</td>
</tr>
</tbody>
</table>
GlideMenu - setHidden(GlideMenuItem item)
Sets an image for an item.

Parameters
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>GlideMenuItem</td>
<td>the item to have the image displayed.</td>
</tr>
</tbody>
</table>

GlideMenu - setImage(GlideMenuItem item, String imgSrc)
Sets the display label for a menu item. The label may contain HTML.

Parameters
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>GlideMenuItem</td>
<td>the item to be labeled.</td>
</tr>
<tr>
<td>label</td>
<td>String</td>
<td>the label to be displayed. The string may contain HTML.</td>
</tr>
</tbody>
</table>

Returns
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
g_menu.setLabel(g_item, "This is a new label");

**GlideMenu - setVisible(GlideMenuItem item)**

Displays the specified item.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>item</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

g_menu.setVisible(g_item);

**GlideModalFormV3 - Client**

Displays a form in a GlideModal.

General usage of the GlideModalForm class involves creating the object, setting any preferences, and then rendering the GlideModalForm.

```javascript
var d = new GlideModalForm('dialog title', 'table_name_or_form_name', [callback on completion of submit])
    d.setPreference('name', 'value');
    d.render();
```

Specify the query parameters that are passed to the form using `setPreference()`. Any name/value pair that you specify with `setPreference()` is sent along with the form POST request to display the form.

The GlideFormModal is set to fill the height of the document window.

**GlideModalFormV3 - addParm(String name, String value)**

Sets the specified form field to the specified value.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Form field name. If the specified name is not a field in the associated modal form, it is ignored.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value to set the specified form field to.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to call `addParm()` to set the value of the sys_id field the modal form.

```javascript
function openDevice(deviceSysID, deviceName) {
  var uName = gel('hidden_user_name').value + "'s ";
  deviceName = new String(deviceName).escapeHTML();
  var gp = new GlideModalForm(uName + deviceName, "cmn_notif_device", refreshNotifPage);
  gp.addParm('sys_id', deviceSysID);
  gp.render();
}
```

**GlideModalFormV3 - GlideModalForm(String title, String tableName, Function onCompletionCallback, Boolean readOnly)**

Creates an instance of the GlideModalForm class.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>String</td>
<td>Modal form title.</td>
</tr>
<tr>
<td>tableName</td>
<td>String</td>
<td>Table being shown.</td>
</tr>
<tr>
<td>onCompletionCallback</td>
<td>Function</td>
<td>Function to call after the form has been submitted and processed on the server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The callback function has the form <code>callbackFunction(String action_verb, String sys_id, String table, String displayValue) where:</code></td>
</tr>
</tbody>
</table>
Parameters (continued)
Name

Type

Description

• action_verb: Name of the UI action
executed. Examples are sysverb_insert
(Submit button), sysverb_cancel,
sysverb_save (Save button).
• sys_id: Sys_id of the affected record.
• table: Name of the table containing the
record.
• displayValue: Value that appears on the
form.
readOnly

Boolean Optional. Flag that indicates whether the
modal form should be set to read only.
Valid values:
• true: Set form to read only.
• false: Set for to read/write.
Default: false

This example shows how to instantiate a GlideModalForm object.
function openDevice(deviceSysID, deviceName) {
var uName = gel('hidden_user_name').value + "'s ";
deviceName = new String(deviceName).escapeHTML();
var gp = new GlideModalForm(uName + deviceName, "cmn_notif_device", refreshNotifPage);
gp.addParm('sys_id', deviceSysID);
gp.render();
}

GlideModalFormV3 - render()
Shows the modal form.
Parameters
Name

Type

Description

None

2370

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


This example shows how to call `render()` to display the modal.

```javascript
function openDevice(deviceSysID, deviceName) {
    var uName = gel('hidden_user_name').value + "'s ";
    deviceName = new String(deviceName).escapeHTML();
    var gp = new GlideModalForm(uName + deviceName, "cmn_notif_device", refreshNotifPage);
    gp.addParm('sys_id', deviceSysID);
    gp.render();
}
```

**GlideModalFormV3 - setCompletionCallback(Function callbackFunction)**

Sets the function to be called when the form has been successfully submitted and processed by the server.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callbackFunction</td>
<td>Function</td>
<td>Callback function to call when the form has been successfully processed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The callback function has the form</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>callbackFunction(String action_verb, String sys_id, String table, String displayValue)</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>where:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>action_verb</code>: action_name from a sys_ui_action record</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>sys_id</code>: Sys_id of the affected record</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>table</code>: Name of the table containing the record</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>displayValue</code>: Value that appears on the form</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
This example shows how to set the onload callback function of the associated modal.

```javascript
function handleCreateOrEdit(targetFieldName, sourceFieldName, adapterRuleId, transformerSysId) {
    dialog = new GlideModalForm('Edit Adapter Rule', 'sys_adapter_rule');
    dialog.setSysID(adapterRuleId); // Pass in sys_id to edit existing record
    dialog.addParm('sysparm_form_only', 'true'); // Add or remove related lists
    dialog.setOnloadCallback(hideModalForm);
    dialog.setCompletionCallback(handleAdapterCreatedOrUpdated);
    dialog.render(); // Open the dialog
}
function handleAdapterCreatedOrUpdated(action_verb, sys_id, table, displayValue) {
    var draftRecordTransformer = g_form.getValue('draft_record_transformer');
    if(draftRecordTransformer == null || draftRecordTransformer.length == 0) {
        // sync Sticky Replications if it is enabled.
        var ajax = new GlideAjax('ReplicationPoolUtil');
        ajax.addParam('sysparm_name', 'syncStickyReplicationSet');
        ajax.addParam('sysparm_entry_set', g_form.getValue('entry_set'));
        ajax.getXMLWait();
    }
}
```

**GlideModalFormV3 - setOnloadCallback(Function callbackFunction)**

Sets the function to be called after the form has been loaded.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callbackFunction</td>
<td>Function</td>
<td>Function to call after the form has been loaded. The callback function has the form callbackFunction(GlideModalForm obj)</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to set the on load callback function of the associated modal.
function handleCreateOrEdit(targetFieldName, sourceFieldName, adapterRuleId, transformerSysId) {
    dialog = new GlideModalForm('Edit Adapter Rule', "sys_adapter_rule");
    dialog.setSysID(adapterRuleId); //Pass in sys_id to edit existing record
    dialog.addParm('sysparm_form_only', 'true'); //Add or remove related lists
    dialog.setOnloadCallback(hideModalForm);
    dialog.setCompletionCallback(handleAdapterCreatedOrUpdated);
    dialog.render(); //Open the dialog
}

GlideModalFormV3 - setSysID(String sys_id)
Sets the object's sys_id preference.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The id preference. One of the query parameters passed to the form.</td>
</tr>
</tbody>
</table>

This example shows how to use the `setSysID()` method to initialize the value of the `sys_id`.

function(startDate, endDate) {
    var dialog = new GlideModalForm("Add Schedule Item", "cmn_schedule_span");
    dialog.setSysID("-1");
    dialog.addParm("sysparm_collection", "cmn_schedule");
    dialog.addParm("sysparm_collectionID", this.sysId);
    dialog.addParm("sysparm_collection_key", "schedule");

    var q = "schedule=" + this.sysId + "^start_date_time=" + startDate.serializeInUserFormat() + "^end_date_time=" + endDate.serializeInUserFormat() + "^";

    if (startDate.isAllDay(endDate))
        q += "^all_day=true";
dialog.addParm("sysparm_query", q);
dialog.render();
}

GlideModalV3 - Client

Provides methods for displaying a content overlay.

This is a fully-featured replacement for GlideWindow and GlideDialogWindow.

Example overlay

GlideModalV3 - get(String id)

Get a GlideModal object by ID.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>The element id of the GlideModal object.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideModal</td>
<td>The object.</td>
</tr>
</tbody>
</table>

GlideModalV3 - getPreference(String name)

Returns the value of the specified preference (property).

Invoking actions that create the modal typically also create the necessary preferences for the modal using the GlideModalV3 - setPreference(String name, String value) method. The UI page client script then consumes these preferences using this method.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the preference value to retrieve. This value must have previously been set on the modal using the GlideModalV3 - setPreference(String name, String value) method.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Specified preference's value.</td>
</tr>
</tbody>
</table>

This example shows a simple case of setting a preference and then retrieving that preference from a specified modal.

```javascript
var gm = new GlideModal('UI_dialog_name');
//Sets the dialog title
gm.setTitle('Show title');

//sets the value of the preference table
gm.setPreference('table', 'incident');

//gets the value of the preference table
var title = gm.getPreference('table');
```

**GlideModalV3 - GlideModal(String id, Boolean readOnly, Number width)**

Creates an instance of the GlideModalV3 class.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>The UI page to load into the modal.</td>
</tr>
<tr>
<td>readOnly</td>
<td>Boolean</td>
<td>When true, hides the close button.</td>
</tr>
<tr>
<td>width</td>
<td>Number</td>
<td>The width in pixels.</td>
</tr>
</tbody>
</table>

**GlideModalV3 - render()**

Renders the UI page in the modal.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gm = new GlideModal("UI_dialog_name");

//Sets the dialog title
gm.setTitle('Show title');
gm.setWidth(550);

//Opens the dialog
gm.render();
```

#### GlideModalV3 - renderWithContent(String html)
Display a modal with the specified HTML content.

The `renderWithContent()` method replaces the `render()` method, and does not request a UI page to render.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>html</td>
<td>String</td>
<td>The HTML content to be shown in the modal.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### GlideModalV3 - renderWithContent(Object html)
Display a modal with the specified HTML content.
The `renderWithContent()` method replaces the `render()` method, and does not request a UI page to render.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>html</td>
<td>Object</td>
<td>The HTML content to be shown in the modal.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideModalV3 - setPreference(String name, String value)**

Sets the specified field on the current form to the specified value.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the form field to update. If this field does not exist on the current form, the request is ignored.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value to store in the specified form field.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gm = new GlideModal('UI_dialog_name');
//Sets the dialog title
gm.setTitle('Show title');
gm.setPreference('table', 'task');
gm.setPreference('name', 'value');

//Opens the dialog
gm.render();
```
GlideModalV3 - setPreferenceAndReload(Array properties)
Set the properties and reload the modal.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>properties</td>
<td>Array</td>
<td>An array of name-value pairs to be set.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideModalV3 - setTitle(String title)
Sets the title of the modal.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>String</td>
<td>The title to be displayed</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gm = new GlideModal('UI_dialog_name');

//Sets the dialog title
gm.setTitle('Show title');
gm.setPreference('name', 'value');

//Opens the dialog
gm.render();
```

GlideModalV3 - setWidth(Number width)
Set the width in pixels.
The modal is boxed into predefined system sizes.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>The number of pixels.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gm = new GlideModal('UI_dialog_name');
//Sets the dialog title
gm.setTitle('Show title');
gm.setPreference('name', 'value');
gm.setWidth(550);
//Opens the dialog
gm.render();
```

**GlideModalV3 - switchView(String newView)**

Change the view and reload the modal.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>newView</td>
<td>String</td>
<td>The view to use.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideNavigationV3 - Client**

Provides methods to control and refresh the navigator and main frame.
The GlideNavigation methods are accessed using the g_navigation global object.

**GlideNavigationV3 - open(String url, String target)**

Redirects to a new URL.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>The URL to load. It can be any URL supported by the browser.</td>
</tr>
<tr>
<td>target</td>
<td>String</td>
<td>Optional. The frame in which to load the content specified by the URL.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Current frame</td>
</tr>
</tbody>
</table>

This example shows how to open the list of active incidents within an instance and display the content in the current frame.

```javascript
g_navigation.open('incident_list.do?sysparm_query=active=true');
```

**GlideNavigationV3 - openPopup(String url, String name, String features, Boolean noStack)**

Opens the specified URL in a popup window.

The **features** parameter is part of the DOM specification and is passed through. For more information on the available feature list, refer to the Mozilla Developer Network.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>URL to open.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Window name.</td>
</tr>
<tr>
<td>features</td>
<td>String</td>
<td>Comma separated list of features for the popup window.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| noStack | Boolean   | Flag that indicates whether to append `sysparm_stack=no` to the URL. This parameter helps prevent unexpected behavior when using the form back button. Valid values:  
  • true: Append `sysparm_stack=no` to the URL.  
  • false: Do not append `sysparm_stack=no` to the URL. |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Window</td>
<td>Instance of the new window.</td>
</tr>
</tbody>
</table>

This example shows how to open the list of active incidents within a popup window called "Active Incidents", and enable the resizable, scrollbars, and status features on the window.

```javascript
const openPopup = (url, title, attributes, closeOnNavExit) => {
  return g_navigation.openPopup(url, title, attributes, closeOnNavExit);
}

g_navigation.openPopup('incident_list.do?sysparm_query=active=true', 'Active Incidents', 'resizable,scrollbars,status', true);
```

**GlideNavigationV3 - openRecord(String tableName, String sys_id)**

Redirects to a record. The record displays in the navigator frame.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table containing the record to display.</td>
</tr>
<tr>
<td>sys_id</td>
<td>String</td>
<td>Sys_id of the record to display.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to open a specified incident record in the navigator frame.
GlideNavigationV3 - refreshNavigator()

Refreshes content in the navigator frame.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to refresh the content in the navigator frame.

```javascript
g_navigation.refreshNavigator();
```

GlideNavigationV3 - reloadWindow()

Reloads the current frame.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to refresh the content in the current frame.

```javascript
g_navigation.reloadWindow();
```

GlideNotificationV3 - Client

You can show messages over the page content.
The GlideNotification method is accessed using the `g_notification` global object. List V3 must be activated for the `g_notification` object to be available.

**GlideNotificationV3 - show(String type, String message)**
Displays the specified string over the page content as the specified type of message.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>The type of message - error, warning, or info.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
// Displays an info message at the top of the screen
nowapi.g_notification.show("info", "The record has been updated");

// Displays an error message at the top of the screen
nowapi.g_notification.show("error", "You need to provide notes!");
```

**GlideOAuthClient - Scoped, Global**
Use these methods for requesting and revoking OAuth refresh and access tokens.

This API can be used in global and scoped scripts. In scoped scripts use the `sn_auth` namespace identifier.

**GlideOAuthClient - getToken(String requestID, String oauthProfileID)**
Retrieves the access and refresh tokens for the client.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>requestID</td>
<td>String</td>
<td>Request ID from the OAuth Requestor Profile [oauth_requestor_profile] table, which references the OAuth Entity Profile [oauth_entity_profile] table.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oauthProfileID</td>
<td>String</td>
<td>OAuth profile ID from the OAuth Entity Profile [oauth_entity_profile] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideOAuthToken</td>
<td>The access and refresh tokens for the client.</td>
</tr>
</tbody>
</table>

This example code shows how to retrieve access and refresh tokens from the instance database.

```java
function dumpToken(token) {
    if(token) {
        gs.info("AccessToken:" + token.getAccessToken());
        gs.info("AccessTokenExpiresIn:" + token.getExpiresIn());
        gs.info("RefreshToken:" + token.getRefreshToken());
    }
}

var oAuthClient = new sn_auth.GlideOAuthClient();
var token = oAuthClient.getToken('248e3017c302301089a7dd5c2840dda5',
 '9c4e78d3c302301089a7dd5c2840dd76');
dumpToken(token);
```

Output:

```
*** Script:
  AccessToken:6MRxD3TRYYViaoKr-JCy3KiaObPu4C9k8afo3MYf9q8zDyHQr8UzMSM3Md2alfaES1rzSYe5ydqgbOwpm7TA
  AccessTokenExpiresIn:1207
  RefreshToken:sc0iTK-0PcVkr14HXPM3vT0FyOPO8iCqC10huQoDSSLBGUSnmEv_fUfJzGWCBb_StsXlOz6r8qF-hRhURWTA
```

**GlideOAuthClient - requestToken(String clientId, String jsonString)**

Retrieves the token for the client, with the request parameters encoded in JSON format.
This example shows a resource owner password grant type request, with request parameters encoded in JSON format.

```java
var oAuthClient = new GlideOAuthClient();
var params = {grant_type: "password", username: "itil", password: 'itil'};
var json = new JSON();
var text = json.encode(params);
var tokenResponse = oAuthClient.requestToken('TestClient', text);
var token = tokenResponse.getToken();
gs.log("AccessToken:" + token.getAccessToken());
gs.log("AccessTokenExpiresIn:" + token.getExpiresIn());
gs.log("RefreshToken:" + token.getRefreshToken());
```

**GlideOAuthClient - requestTokenByRequest(String clientName, GlideOAuthClientRequest request)**

Retrieves the token for the client, with the client name and the request set into a GlideOAuthClientResponse object.
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideOAuthClientResponse</td>
<td>The token for the client.</td>
</tr>
</tbody>
</table>

GlideOAuthClient - revokeToken(String clientName, String accessToken, String refreshToken, GlideOAuthClientRequest request)

Revolves the access or refresh token for the client, with the request and optional header parameters set into a GlideOAuthClientRequest object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clientName</td>
<td>String</td>
<td>The client name.</td>
</tr>
<tr>
<td>accessToken</td>
<td>String</td>
<td>The access token.</td>
</tr>
<tr>
<td>refreshToken</td>
<td>String</td>
<td>The refresh token.</td>
</tr>
<tr>
<td>request</td>
<td>GlideOAuthClientRequest</td>
<td>The request.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideOAuthClientResponse</td>
<td>The token for the client.</td>
</tr>
</tbody>
</table>

GlideOAuthClientRequest - Scoped, Global

Use these methods for handling OAuth client requests.

This API can be used in global and scoped scripts. In scoped scripts use the `sn_auth` namespace identifier.

GlideOAuthClientRequest - getGrantType()

Retrieves the grant type.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The grant type.</td>
</tr>
</tbody>
</table>

**GlideOAuthClientRequest - getHeader(String name)**

Retrieves the HTTP headers for the string you provide.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the parameter.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StringMap</td>
<td>The string map with the HTTP headers.</td>
</tr>
</tbody>
</table>

**GlideOAuthClientRequest - getHeaders()**

Retrieves the HTTP headers.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StringMap</td>
<td>The string map with the HTTP headers.</td>
</tr>
</tbody>
</table>

**GlideOAuthClientRequest - getParameter(String name)**

Retrieves the parameters for the parameter name you provide.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The parameter name for which you want the parameters.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The parameters.</td>
</tr>
</tbody>
</table>

**GlideOAuthClientRequest - getPassword()**

Retrieves the password.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The password.</td>
</tr>
</tbody>
</table>

**GlideOAuthClientRequest - getRefreshToken()**

Retrieves the refresh token.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The refresh token.</td>
</tr>
</tbody>
</table>
**GlideOAuthClientRequest - getScope()**
Retrieves the scope.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>none</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

**GlideOAuthClientRequest - getUserName()**
Retrieves the user name.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>none</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

**GlideOAuthClientRequest - setGrantType(String grantType)**
Sets the grant type for the string you provide.

⚠️ **Note:** You only need to set the grant type if it is not already defined in the OAuth provider profile.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>name</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideOAuthClientRequest - setHead(String name, String value)**

Retrieves the HTTP headers for the string you provide.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the parameter.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The value of the parameter.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideOAuthClientRequest - setParameter(String name, String value)**

Sets the parameters for the name:value pair of strings you provide.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The parameter name for which you want the parameters.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The value of the parameter.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideOAuthClientRequest - setPassword(String password)**

Sets the password with the string you provide.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>password</td>
<td>String</td>
<td>The user name.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideOAuthClientRequest - setRefreshToken(String refreshToken)

Sets the refresh token with the string you provide.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>refreshToken</td>
<td>String</td>
<td>The refresh token.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows a resource owner password grant type request.

```javascript
var tokenRequest = new GlideOAuthClientRequest();
tokenRequest.setGrantType("password");
tokenRequest.setUserName("itil");
tokenRequest.setPassword("itil");
tokenRequest.setScope(null);

var oAuthClient = new GlideOAuthClient();
var tokenResponse = oAuthClient.requestToken("TestClient", tokenRequest);
gs.log("Error:"+ tokenResponse.getErrorMessage());

var token = tokenResponse.getToken();
if(token){
gs.log("AccessToken:"+ token.getAccessToken());
gs.log("AccessTokenExpiresIn:"+ token.getExpiresIn());
}
This example shows a refresh token grant type request.

```javascript
var tokenRequest = new GlideOAuthClientRequest();
tokenRequest.setGrantType("refresh_token");

// Set refresh token
var oAuthClient = new GlideOAuthClient();
tokenRequest.setRefreshToken("N-GtdSVLkWP_Cr-TysXdmNy59ZYaJmu5ZaS4YaSluXDm0kCkInEnu-hwM5SsGYSFw+jxauVmoaq7xJNoalXFQ");
tokenRequest.setScope(null);

var oAuthClient = new GlideOAuthClient();
tokenResponse = oAuthClient.requestToken("TestClient", tokenRequest);

gs.log("Error:" + tokenResponse.getErrorMessage());
token = tokenResponse.getToken();
if (token){
    gs.log("AccessToken:" + token.getAccessToken());
    gs.log("AccessTokenExpiresIn:" + token.getExpiresIn());
    gs.log("AccessTokenSysID:" + token.getAccessTokenSysID());
    gs.log("RefreshToken:" + token.getRefreshToken());
    gs.log("RefreshTokenSysID:" + token.getRefreshTokenSysID());
}
```

### GlideOAuthClientRequest - setScope(String scope)

Sets the scope for the string you provide.

⚠️ **Note:** You only need to set the scope if it is not already defined in the OAuth provider.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scope</td>
<td>String</td>
<td>The scope.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### GlideOAuthClientRequest - setUserName(String userName)

Sets the user name with the string you provide.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userName</td>
<td>String</td>
<td>The user name.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### GlideOAuthClientResponse - Scoped, Global

Use these methods for handling OAuth client responses.

This API can be used in global and scoped scripts. In scoped scripts use the `sn_auth` namespace identifier.

#### GlideOAuthClientResponse - getBody()

Retrieves all of the response information, including instance information.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The response information.</td>
</tr>
</tbody>
</table>

#### GlideOAuthClientResponse - getContentType()

Retrieves the HTTP response content header from an external OAuth provider.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>GlideOAuthClientResponse - getErrorMessage()</td>
<td>String</td>
<td>The HTTP response header.</td>
</tr>
</tbody>
</table>

GlideOAuthClientResponse - getErrorMessage()
Retrieves the error message if authentication is not successful.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The error message.</td>
</tr>
</tbody>
</table>

GlideOAuthClientResponse - getResponseCode()
Retrieves the HTTP response code from the external OAuth provider.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The HTTP response code.</td>
</tr>
</tbody>
</table>

GlideOAuthClientResponse - getResponseParameters()
Retrieves the error message if authentication is not successful.
GlideOAuthClientResponse - getToken()
Retrieves the refresh token.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>none</td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MapString</td>
<td>The response content.</td>
</tr>
</tbody>
</table>

GlideOAuthToken - Scoped, Global
Use the GlideOAuthToken methods for retrieving OAuth access token and information about the access token.

This API can be used in global and scoped scripts. In scoped scripts use the sn_auth namespace identifier.

GlideOAuthToken - getAccessToken()
Retrieves the access token associated with the GlideOAuthToken object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Access token.</td>
</tr>
</tbody>
</table>

This example code shows how to retrieve access and refresh tokens from the instance database. See also GlideOAuthClient.

```javascript
function dumpToken(token) {
    if(token) {
        gs.info("AccessToken:" + token.getAccessToken());
        gs.info("AccessTokenExpiresIn:" + token.getExpiresIn());
        gs.info("RefreshToken:" + token.getRefreshToken());
    }
}

var oAuthClient = new sn_auth.GlideOAuthClient();
var token = oAuthClient.getToken('248e3017c302301089a7dd5c2840dda5',
    '9c4e78d3c302301089a7dd5c2840dd76');
dumpToken(token);
```

Output:

```text
*** Script: 
   AccessToken:6MRxD3TRYYViakR-JCy3Kia0xBPu4C9k8oao3MYf9q8z3yHq8Uz3M3D2afal7ES1rzSYe5ydqgbOWpm7TA
*** Script: AccessTokenExpiresIn:1207
*** Script: 
   RefreshToken:sc0iTK-0PcVr16HXP3vT0Fy0PO81CqC10huQ00DSSLBGUSnEv_fUfJzGWCBb_StsXIOz6r8qF-hRhURWTA
```

GlideOAuthToken - getAccessTokenSysID()

Deprecated. Retrieves the sys_id of the token ID associated with the GlideOAuthToken object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the access token.</td>
</tr>
</tbody>
</table>

**GlideOAuthToken - getExpiresIn()**

Retrieves the lifespan of the access token associated with the GlideOAuthToken object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number (Long)</td>
<td>Lifespan of the token.</td>
</tr>
<tr>
<td>Unit: Seconds</td>
<td></td>
</tr>
</tbody>
</table>

This example code shows how to retrieve access and refresh tokens from the instance database. See also GlideOAuthClient.

```javascript
function dumpToken(token) {
  if(token) {
    gs.info("AccessToken:" + token.getAccessToken());
    gs.info("AccessTokenExpiresIn:" + token.getExpiresIn());
    gs.info("RefreshToken:" + token.getRefreshToken());
  }
}

var oAuthClient = new sn_auth.GlideOAuthClient();
var token = oAuthClient.getToken('248e3017c302301089a7dd5c2840dda5',
  '9c4e78d3c302301089a7dd5c2840dd76');
dumpToken(token);
```

Output:

```plaintext
*** Script:***
AccessToken:6MRxD3TRYYvIaoKr-JCy3KiaOxBPu4C9k8oaf03MYf9q8zDYHqr8UZnSM3Md2alfaES1rzSYe5ydqgHwpm7TA
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### GlideOAuthToken - getRefreshToken()

Refreshes the token associated with the GlideOAuthToken object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Refreshed token.</td>
</tr>
</tbody>
</table>

This example code shows how to retrieve access and refresh tokens from the instance database. See also GlideOAuthClient.

```javascript
function dumpToken(token) {
    if(token) {
        gs.info("AccessToken:" + token.getAccessToken());
        gs.info("AccessTokenExpiresIn:" + token.getExpiresIn());
        gs.info("RefreshToken:" + token.getRefreshToken());
    }
}

var oAuthClient = new sn_auth.GlideOAuthClient();
var token = oAuthClient.getToken('248e3017c302301089a7dd5c2840dda5', '9c4e78d3c302301089a7dd5c2840dd76');
dumpToken(token);
```

**Output:**

```plaintext
*** Script: AccessTokenExpiresIn:1207
 *** Script: AccessToken:6MRxD3TRYYViakKr-JCy3K1a0xBPu4C9k8oaf03Myf9q8zqfHq8UzMSM3Md2a1faES1rzSYe5ydqg
bOwpm7TA
 *** Script: AccessToke
```
GlideOAuthToken - getRefreshTokenSysID()

Deprecated. Retrieves the sys_id of the refresh token.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the refresh token.</td>
</tr>
</tbody>
</table>

GlideOAuthToken - getScope()

Retrieves the token scope, which is the amount of access granted by the access token.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Scope of the access token.</td>
</tr>
</tbody>
</table>

GlidePluginManager - Scoped

The scoped GlidePluginManager API provides a method for determining if a plugin has been activated.

**Scoped GlidePluginManager - isActive(String pluginID)**

Determines if the specified plugin has been activated.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pluginID</td>
<td>String</td>
<td>Unique plugin identifier.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates if the plugin is active.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Plugin is active.</td>
</tr>
<tr>
<td></td>
<td>• false: Plugin is inactive.</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('sys_plugins');
var queryString = "active=0^ORactive=1";
now_GR.addEncodedQuery(queryString);
now_GR.query();
var pMgr = new GlidePluginManager();

while (now_GR.next()) {
    var name = now_GR.getValue('name');
    var pID = now_GR.getValue('source');
    var isActive = pMgr.isActive(pID);
    if (isActive)
        gs.info('The plugin ' + name + ' is active');
}
```

Output:

The plugin Country Lookup Data is active
The plugin Database Replication is active
The plugin REST API Provider is active
The plugin Ten Cool Things is active

GlideQuery - Scoped, Global

The GlideQuery API is an alternative to GlideRecord to perform CRUD operations on record data from server-side scripts.

The GlideQuery API lets you:
• Use standard JavaScript objects and types for queries and results.
• Quickly diagnose query errors with additional checks and clear error messages.
• Simplify your code by avoiding boiler-plate query patterns.
• Avoid common performance issues without needing deeper knowledge of GlideRecord.

Use the GlideQuery API in scoped or global server-side scripts. When used within a scoped app, it must be prefixed with the global scope.

```javascript
new global.GlideQuery('sys_user')
// ...
```

This API requires the GlideQuery [com.sn_glidequery] plugin.

**Implementation**

This API works together with the Stream and Optional APIs in a builder pattern where the method calls chain together, each method building on the returned result of the previous method. Use methods to define the attributes of the query. The methods do not execute until you call a terminal method, a method that returns a query result, allowing you to define the requirements of the query before executing it.

If the query returns a single record, the system wraps the result in an Optional object. If the query returns a stream of records, the system wraps the result in a Stream object. These objects let you manage the result using a set of methods in each API.

For example, here's a script that performs a query on the Task table and groups the records by priority and returns groups with total reassignments greater than four.

```javascript
var query = new global.GlideQuery('task')
    .where('active', true) //Returns new GlideQuery object with a "where" clause.
    .groupBy('priority') //Returns new GlideQuery object with a "group by" clause.
    .aggregate('sum', 'reassignment_count') //Returns new GlideQuery object with a "sum(reassignment_count)" clause.
    .having('sum', 'reassignment_count', '>', 4) //Returns new GlideQuery object with a "having reassignment_count > 4" clause.
    .select() //Returns a stream of records wrapped in a Stream object.
    .toArray(10); //Terminal method in the Stream class that executes the query and returns the result.
```
Error handling

The GlideQuery API throws an error when your query has a problem, and includes a clear explanation to help guide you. GlideQuery checks for:

- Invalid fields
- Invalid value types for a field
- Invalid values for choice fields
- Invalid query operators

For example, this code sample would throw an error because the queried field does not exist in the table.

```javascript
new global.GlideQuery('task')
  .where('id', '4717dfe5a9fe198100450448b2404c16') // should be 'sys_id'
  .select('description', 'severity')
  .toArray(100);
// Error: Unable to find field 'id' in table 'task'. Known fields: active, activity_due, ...
```

This code sample would throw an error because the data type of one of the arguments is incorrect.

```javascript
new global.GlideQuery('task')
  .where('priority', 'one') // priority is an integer (should be 1)
  .select('description', 'severity')
  .toArray(100);
// Error: Unable to match value ['one'] with field 'priority' in table 'task'. Expecting type 'integer'
```

Reuse

Because GlideQuery objects are immutable, you can reuse them later in other parts of your code. For example, this script creates a query and then uses the GlideQuery object later to generate a report.

```javascript
var highPriorityTasks = new global.GlideQuery('task')
  .where('active', true)
  .where('priority', 1);

generateReport(highPriorityTasks);
notifyOwners(highPriorityTasks);
var avgReassignmentCount = highPriorityTasks
  .avg('reassignment_count')
  .orElse(0)
```
Limitations

The GlideQuery API does not support:

- Reading or writing to tables that do not allow access from other scopes.
- Reading encoded queries.
- GlideDate or GlideDateTime objects, which are read as JavaScript strings.
- FX Currency fields.
- Queries with ambiguous conditional logic. For example, the following query is unclear because the system does not know whether to execute (active = true AND name != null) OR last_name = Luddy OR active = true AND (name != null OR last_name = Luddy).

```javascript
var user = new global.GlideQuery('sys_user')
  .where('active', true)
  .whereNotNull('name')
  .orWhere('last_name', 'Luddy')
  .selectOne()
  .get()
```

See the `where()` method to understand how to nest a child query instead.

ℹ️ Note: Because the GlideQuery API converts GlideRecord objects into standard JavaScript objects, it may take longer to execute queries. To reduce performance issues, avoid creating loops that iterate over large numbers of records.

GlideQuery - GlideQuery(String table)

Instantiates a GlideQuery object used to build and execute record queries.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>Table to query.</td>
</tr>
</tbody>
</table>

This example instantiates a query of the User table.

```javascript
var query = new global.GlideQuery('sys_user');
```

GlideQuery - aggregate(String aggregateType, String field)

Aggregates a field using a specified aggregation function.
Use this method to build queries that aggregate against multiple fields or use multiple aggregate functions, or if you must use the `groupBy()` method. If you only want to aggregate against one field with one function, and you don’t need to use `groupBy()`, then use one of these methods instead:

- `avg()`
- `min()`
- `max()`
- `count()`

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>aggregateType</code></td>
<td>String</td>
<td>The type of aggregation function to perform. Options include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>min</code>: Returns the smallest value of all matching records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>max</code>: Returns the largest value of all matching records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>sum</code>: Returns the sum of all matching records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>avg</code>: Returns the average of all matching records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>count</code>: Returns the number of number of matching records.</td>
</tr>
<tr>
<td><code>field</code></td>
<td>String</td>
<td>Field on which to perform the operation.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQuery</td>
<td>The query object being built.</td>
</tr>
</tbody>
</table>

This example performs a query on the Task table that groups the records by priority, adds the numbers in the reassignment count field for each group, and returns groups with total reassignments greater than four.

```javascript
var query = new global.GlideQuery('task')
    .where('active', true) //Returns the GlideQuery object to add more attributes to the query.
    .groupBy('priority') //Returns the GlideQuery object to add more attributes to the query.
```
$.aggregate('sum', 'reassignment_count') //Returns the GlideQuery object to add more attributes to the query.
$.having('sum', 'reassignment_count', '>', 4) //Returns the GlideQuery object to add more attributes to the query.
$.select() //Returns a stream of records wrapped in a Stream object.
$.toArray(10); //Terminal method in the Stream class that executes the query and returns the result.

gs.info(JSON.stringify(query, null, 2));

**Output:**

```json
[
  {
    "group":{
      "priority":1
    },
    "sum":{
      "reassignment_count":11
    }
  },
  {
    "group":{
      "priority":3
    },
    "sum":{
      "reassignment_count":6
    }
  },
  {
    "group":{
      "priority":5
    },
    "sum":{
      "reassignment_count":5
    }
  }
]
```

**GlideQuery - avg(String field)**

Returns the aggregate average of a given numeric field.

You can only use this method on fields of the following types:
• Integer
• Long
• Floating Point Number
• Double
• Currency

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Field on which to perform the operation.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional</td>
<td>Object that contains the aggregate average of the given field.</td>
</tr>
</tbody>
</table>

This example shows how to return the average number of faults in the cmdb_ci table.

```javascript
var faults = new global.GlideQuery('cmdb_ci')
  .avg('fault_count')
  .orElse(0);

gs.info(JSON.stringify(faults));
```

Output:

```
0.0037
```

**GlideQuery - count()**

Returns the number of records that match the query.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of records that match the query.</td>
</tr>
</tbody>
</table>

This example returns the number of active records in the User table.

```javascript
var userCount = new global.GlideQuery('sys_user')
  .where('active', true)
  .count();
```

Output:

612

**GlideQuery - deleteMultiple()**

Deletes all records in the table specified by the preceding Where clauses.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This example deletes all active records in the User table where the last name is Jeter.

```javascript
var query = new global.GlideQuery('sys_user')
  .where('active', true)
  .where('last_name', 'Jeter')
  .deleteMultiple();
```

**GlideQuery - disableAutoSysFields()**

Disables updating system fields, or fields with a name that starts with the `sys` prefix, such as `sys_created_on`, `sys_updated_on`, and `sys_mod_count`. Only applies to the specified query.
This example adds a record to the task table, but does not set system fields. Without calling this method, the below example would update sys_updated_on, sys_mod_count, and so on.

```javascript
var query = new global.GlideQuery('task')
  .disableAutoSysFields()
  .insert({ description: 'example', priority: 1 });
```

**GlideQuery - disableWorkflow()**

Disables any business rules, flows, workflows, or audit records that would run or be created as the result of the query.

This example updates multiple records in the Task table without triggering any automatic business processes.

```javascript
var query = new global.GlideQuery('task')
  .disableWorkflow()
  .where('active', true)
  .updateMultiple({ priority: 1 });
```
GlideQuery - forceUpdate()

Forces a database update even when no record changes are made. For example, you can use this method to force a business rule to execute.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQuery</td>
<td>The query object being built.</td>
</tr>
</tbody>
</table>

This example force updates Task records with a certain sys_id.

```javascript
var forceUpdate = new global.GlideQuery('task')
  .forceUpdate()
  .where('sys_id', taskId)
  .update()
```

GlideQuery - get(String key, Array selectedFields)

Returns a single record from the query.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Sys_id of the record to return.</td>
</tr>
<tr>
<td>selectedFields</td>
<td>Array</td>
<td>Optional. Additional fields to return in the result. Default: The system always returns the sys_id.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional</td>
<td>Object used to interact with a single record.</td>
</tr>
</tbody>
</table>

Example that returns a record based on sys_id.
var user = new global.GlideQuery('sys_user')
  .get('5137153cc611227c000bbd1bd8cd2005', ['first_name', 'last_name'])  //Returns an Optional object.
  .orElse({ first_name: 'Default', last_name: 'User' });  //Method in the Optional class to return a default value.

gs.info(JSON.stringify(user, null, 2));

Output:

{  
  "sys_id":"5137153cc611227c000bbd1bd8cd2005",
  "first_name":"Fred",
  "last_name":"Luddy"
}

**GlideQuery - getBy(Object keyValues, Array selectedFields)**

Returns an Optional object containing a single record based on a set of name-value pairs to query by. Assumes the '=' operator for each name-value pair.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>keyValues</td>
<td>Object</td>
<td>Object where the keys are the name of the fields, and the values are the values to query for.</td>
</tr>
<tr>
<td>selectedFields</td>
<td>Array</td>
<td>Optional. Additional fields to return in the result. Default: The system always returns the sys_id.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional</td>
<td>Object used to interact with a single record.</td>
</tr>
</tbody>
</table>

**Example that returns a record by querying for a user's name.**

```javascript
var user = new global.GlideQuery('sys_user')
  .getBy(
    {  
      first_name: 'Fred',
      last_name: 'Luddy'
    },  
    ['first_name', 'last_name', 'city', 'active']  
  )  // select first_name, last_name, city, active
  .orElse(
    {}  
  );
```
first_name: 'Nobody',
last_name: 'Found',
city: 'Nowhere',
active: false
});

gs.info(JSON.stringify(user, null, 2));

Output:

{
  "first_name":"Fred",
  "last_name":"Luddy",
  "city":null,
  "active":true,
  "sys_id":"5137153cc611227c000bbd1bd8cd2005"
}

**GlideQuery - groupBy(String fields)**

Groups the query results by a designated field or fields.

You must use this method with the `aggregate()` method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>String or Array of Strings</td>
<td>Field or fields to group the results by.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQuery</td>
<td>The query object being built.</td>
</tr>
</tbody>
</table>

This example performs a query on the Task table that groups the records by priority, adds the numbers in the reassignment count field for each group, and returns groups with total reassignments greater than four.

```javascript
var query = new global.GlideQuery('task')
  .where('active', true) //Returns the GlideQuery object to add more attributes to the query.
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
.groupBy('priority') //Returns the GlideQuery object to add more attributes to the query.
.aggregate('sum', 'reassignment_count') //Returns the GlideQuery object to add more attributes to the query.
.having('sum', 'reassignment_count', '>', 4) //Returns the GlideQuery object to add more attributes to the query.
.select() //Returns a stream of records wrapped in a Stream object.
.toArray(10); //Terminal method in the Stream class that executes the query and returns the result.

gs.info(JSON.stringify(query, null, 2));

Output:

[  
  {  
    "group":{  
      "priority":1  
    },  
    "sum":{  
      "reassignment_count":11  
    }  
  },  
  {  
    "group":{  
      "priority":3  
    },  
    "sum":{  
      "reassignment_count":6  
    }  
  },  
  {  
    "group":{  
      "priority":5  
    },  
    "sum":{  
      "reassignment_count":5  
    }  
  }  
]
GlideQuery - having(String aggregateType, String field, String operator, Number value)

Filters aggregate groups so that you can display only groups of results that match a specified condition.

Must use this method with the aggregate() or groupBy() methods.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aggregateType</td>
<td>String</td>
<td>The type of aggregation function to perform. Options include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• min: Returns the smallest value of all matching records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• max: Returns the largest value of all matching records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• sum: Returns the sum of all matching records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• avg: Returns the average of all matching records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• count: Returns the number of number of matching records.</td>
</tr>
<tr>
<td>field</td>
<td>String</td>
<td>Field on which to perform the operation.</td>
</tr>
<tr>
<td>operator</td>
<td>String</td>
<td>Numeric operator to use in the operation. Options include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &gt;: Greater than.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &lt;: Less than.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &gt;=: Greater than or equal to.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &lt;=: Less than or equal to.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• =: Equal to.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• !=: Not equal to.</td>
</tr>
<tr>
<td>value</td>
<td>Number</td>
<td>Number value to use in the operation.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQuery</td>
<td>The query object being built.</td>
</tr>
</tbody>
</table>
This example performs a query on the Task table that groups the records by priority, adds the numbers in the reassignment count field for each group, and returns groups with total reassignments greater than four.

```javascript
var query = new global.GlideQuery('task')
    .where('active', true) //Returns the GlideQuery object to add more attributes to the query.
    .groupBy('priority') //Returns the GlideQuery object to add more attributes to the query.
    .aggregate('sum', 'reassignment_count') //Returns the GlideQuery object to add more attributes to the query.
    .having('sum', 'reassignment_count', '>', 4) //Returns the GlideQuery object to add more attributes to the query.
    .select() //Returns a stream of records wrapped in a Stream object.
    .toArray(10); //Terminal method in the Stream class that executes the query and returns the result.

gs.info(JSON.stringify(query, null, 2));
```

Output:

```json
[
  {
    "group": {
      "priority": 1
    },
    "sum": {
      "reassignment_count": 11
    }
  },
  {
    "group": {
      "priority": 3
    },
    "sum": {
      "reassignment_count": 6
    }
  },
  {
    "group": {
      "priority": 5
    },
    "sum": {
      "reassignment_count": 5
    }
  }
]
```
GlideQuery - insert(Object keyValues, Object selectedFields)

Inserts a record and returns an Optional object containing the record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>keyValues</td>
<td>Object</td>
<td>Object containing name-value pairs to insert into the record. Unspecified fields will be null.</td>
</tr>
<tr>
<td>selectedFields</td>
<td>Array</td>
<td>Optional. Additional fields to return in the result. Default: The system always returns the sys_id.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional</td>
<td>Object used to interact with a single record.</td>
</tr>
</tbody>
</table>

This example shows how to insert a record based on a user's first and last name.

```javascript
var fred = new global.GlideQuery('sys_user')
    .insert({ first_name: 'Fred', last_name: 'Luddy' })
    .get();
gs.info(JSON.stringify(fred, null, 2));
```

**Output:**

```json
{
    "sys_id": "cf16eed0e82a9010f8778bda83d255d2",
    "first_name": "Fred",
    "last_name": "Luddy"
}
```

GlideQuery - insertOrUpdate(Object changes, Object selectedFields)

Updates an existing record, or inserts a new record if one does not already exist.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>changes</td>
<td>Object</td>
<td>Object containing name-value pairs to update or insert into the record.</td>
</tr>
<tr>
<td>selectedFields</td>
<td>Array</td>
<td>Optional. Additional fields to return in the result. Default: The system always returns the sys_id.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional</td>
<td>Object used to interact with a single record.</td>
</tr>
</tbody>
</table>

This example shows how to insert a new record that does not already exist in the system.

```javascript
// insert a new record
var user = new GlideQuery('sys_user')
  .insertOrUpdate({
    first_name: 'George',
    last_name: 'Griffey'
  })
  .orElse(null);
```

This example shows how to update an existing record.

```javascript
// update existing record
var user = new global.GlideQuery('sys_user')
  .insertOrUpdate({
    sys_id: '2d0efd6c73662300bb513198caf6a72e',
    first_name: 'George',
    last_name: 'Griffey'
  })
  .orElse(null);
```

### GlideQuery - limit(Number limit)

Limits the number of records returned in a query.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>limit</td>
<td>Number</td>
<td>Number of records to return.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQuery</td>
<td>The query object being built.</td>
</tr>
</tbody>
</table>

This example shows how to limit the results returned to five records.

```javascript
var incidents = new global.GlideQuery('incident')
  .limit(5)
  .select('priority', 'description')
  .toArray(100);

gs.info(JSON.stringify(incidents, null, 2));
```

Output:

```json
[
  {
    "priority": 3,
    "description": "I am unable to connect to the email server. It appears to be down.",
    "sys_id": "1c741bd70b2322007518478d83673af3"
  },
  {
    "priority": 3,
    "description": "My computer is not detecting the headphone device. It could be an issue with the USB port.",
    "sys_id": "1c83270673202302728660c4cf6a7b9"
  },
  {
    "priority": 1,
    "description": "I can't remember my password and need to log in. Can someone reset my password asap? I am blocked on several urgent items until I can log in again.",
    "sys_id": "46b66a40a9fe198101f243dfbc79033d"
  },
  {
    "priority": 4,
    "description": "Currently running 10GR1 and need to upgrade to 10GR2."
  },
  {
    "priority": 3,
    "description": "I'm replacing my old phone with a Blackberry and require assistance to get it set up. I'd like to get the files and contacts transferred to the new phone and need help getting the two factor authentication app set up on it."
  }
]```
GlideQuery - max(String field)
Returns the aggregate maximum of a given field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Field on which to perform the operation.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional</td>
<td>Object used to interact with a single record.</td>
</tr>
</tbody>
</table>

This example shows how to return the maximum value, or highest alphanumerical value, of a given field.

```javascript
var name = new global.GlideQuery('sys_user')
    .max('last_name')
    .orElse('');

gs.info(JSON.stringify(name));
```

Output:

"Zortman"

GlideQuery - min(String field)
Returns the aggregate minimum of a given field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Field on which to perform the operation.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional</td>
<td>Object used to interact with a single record.</td>
</tr>
</tbody>
</table>

This example shows how to return the minimum value, or lowest alphanumeric value, of a given field.

```javascript
var name = new global.GlideQuery('sys_user')
    .min('last_name')
    .orElse('');
gs.info(JSON.stringify(name));
```

Output:

"Abel"

**GlideQuery - orderBy(String fields)**

Orders the returned result in ascending order by a given field.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>String</td>
<td>Comma-delimited fields to order the result by in ascending order.</td>
</tr>
</tbody>
</table>

This example shows how to order results in ascending order by record number.

```javascript
var query = new global.GlideQuery('incident')
    .orderBy('number')
    .limit(5)
    .select('priority', 'description')  //Returns a stream of records wrapped in a Stream object.
    .toArray(100);  //Terminal method in the Stream class that executes the query and returns the result.
gs.info(JSON.stringify(query, null, 2));
```
Output:

```json
[
  {
    "priority":1,
    "description":"User can't access email on mail.company.com.",
    "sys_id":"9c573169c611228700193229fff72400"
  },
  {
    "priority":1,
    "description":"User can't get to any of his files on the file server.",
    "sys_id":"9d385017c611228701d22104cc95c371"
  },
  {
    "priority":1,
    "description":"I just moved from floor 2 to floor 3 and my laptop cannot connect to any wireless network.",
    "sys_id":"e8caedc0a80164017df472f39eae1"
  },
  {
    "priority":1,
    "description":"User forgot their email password.",
    "sys_id":"9d3c1197c611228701cd1d94bc32d76d"
  },
  {
    "priority":1,
    "description":"Watcher daemon detected that the CPU was 100% busy for more than 10 minutes",
    "sys_id":"e8e875b0c0a80164009dc852b4d677d5"
  }
]
```

GlideQuery - orderByDesc(String fieldOrAggregate, String field)

Orders the returned result in descending order by a given field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldOrAggregate</td>
<td>String</td>
<td>If the query does not use the aggregate() method, pass the field to order the results by. If the query uses the aggregate() method, pass the type of aggregation function to perform. Options include:</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>• <strong>min</strong>: Returns the smallest value of all matching records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>max</strong>: Returns the largest value of all matching records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>sum</strong>: Returns the sum of all matching records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>avg</strong>: Returns the average of all matching records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>count</strong>: Returns the number of number of matching records.</td>
</tr>
<tr>
<td>field</td>
<td>String</td>
<td>Optional. Field to order the result by in descending order. Required for queries using the <code>aggregate()</code> method.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQuery</td>
<td>The query object being built.</td>
</tr>
</tbody>
</table>

This example shows how to order the result in descending order by number.

```javascript
var query = new global.GlideQuery('incident')
    .orderByDesc('number')
    .select('number', 'description')
    .limit(5)
    .toArray(100);

gs.info(JSON.stringify(query, null, 2));
```

**Output:**

```json
[
  {
    "number":"INC0010112",
    "description":null,
    "sys_id":"552c48888c033300964f4932b03eb092"
  },
  {
    "number":"INC0010111",
    "description":null,
```
This example shows how to order an aggregate result by the sum of child incidents.

```javascript
var aggQuery = new GlideQuery('incident')
    .aggregate('sum', 'child_incidents')
    .groupBy('category')
    .orderByDesc('sum', 'child_incidents')
    .select()
    .toArray(100);

gs.info(JSON.stringify(aggQuery, null, 2));
```

Output:

```json
[
  {
    "group":{
      "category":"hardware"
    },
    "sum":{
      "child_incidents":2
    }
  },
  {
    "group":{
      "category":"software"
    },
    "sum":{
      "child_incidents":3
    }
  }
]
```
GlideQuery - orWhere(String fieldOrQuery, String operator, Any value)

Adds an OR clause to a query that returns values based on a given condition.
## Note
Precede this method with the `where()`, `whereNull()`, or `whereNotNull()` methods.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldOrQuery</td>
<td>String or GlideQuery</td>
<td>Field or another GlideQuery object used in the where clause. If passing a field, you can dot-walk to a desired value. For example, <code>company.name</code>.</td>
</tr>
<tr>
<td>operator</td>
<td>String</td>
<td>Optional. Operator used in the OR clause. If you do not pass an argument, the system uses the = operator. You do not need to include a placeholder value.</td>
</tr>
<tr>
<td>value</td>
<td>Any</td>
<td>Value used in the OR clause.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQuery</td>
<td>The query object being built.</td>
</tr>
</tbody>
</table>

This example shows how to add a simple OR clause to a query.

```javascript
var query = new global.GlideQuery('sys_user')
    .where('failed_attempts', '>', 0)
    .orWhere('last_login', '<', '2019-04-15')
    .select()
    .toArray(100)

gs.info(JSON.stringify(query, null, 2));
```

### Output:

```
[
  {
    "sys_id":"005d50b536073005e0addeeffb2f4"
  },
  {
    "sys_id":"30ad318577ab2300454792718a10619e"
  },
  {
    "sys_id":"3883f4c073012302728660c4cf6a754"
  }
]
```
This example shows how to add an `orWhere` clause that contains a separate query.

```javascript
// active = true OR (title = 'Vice President' AND state = 'CA')
var query = new GlideQuery('sys_user')
  .where('active', true)
  .orWhere(new GlideQuery()
    .where('title', 'Vice President')
    .where('state', 'CA'))
  .select('name')
  .limit(5)
  .toArray(100)

gs.info(JSON.stringify(query, null, 2));
```

Output:

```json
[
  {
    "name": "survey user",
    "sys_id": "005d500b536073005e0adce6f7b12f4"
  },
  {
    "name": "Lucius Bagnoli",
    "sys_id": "02826bf03710200044e0bfc8bcbe5d3f"
  },
  {
    "name": "Jimmie Barninger",
    "sys_id": "02826bf03710200044e0bfc8bcbe5d55"
  }
]
```
GlideQuery - orWhereNotNull(String field)

Adds an OR clause that returns records that do not contain a null value in a given field.

ℹ️ **Note:** Precede this method with the `where()`, `whereNull()`, or `whereNotNull()` methods.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Field used in the query.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQuery</td>
<td>The query object being built.</td>
</tr>
</tbody>
</table>

This example shows how to query the User table and return results where the first and last names are not null.

```javascript
var query = new global.GlideQuery('sys_user')
  .whereNotNull('first_name')
  .orWhereNotNull('last_name')
  .select()
  .limit(5)
  .toArray(100)

gs.info(JSON.stringify(query, null, 2));
```

**Output:**

```json
[
  {
    "sys_id":"005d500b536073005e0dadeeff7b12f4"
  }
]```
GlideQuery - orWhereNull(String field)

Adds an OR clause to a query that returns records that contain a null value in a given field.

⚠️ Note: Precede this method with the `where()`, `whereNull()`, or `whereNotNull()` methods.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Field used in the query.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQuery</td>
<td>The query object being built.</td>
</tr>
</tbody>
</table>

This example shows how to query the User table and return records where the first or last names are null.

```javascript
var query = new global.GlideQuery('sys_user')
  .whereNull('last_name')
  .orWhereNull('first_name')
  .select()
  .toArray(100)

gs.info(JSON.stringify(query, null, 2));
```
Output:

```json
[
  {
    "sys_id": "1195569be81a1010f8778bda83d25585"
  },
  {
    "sys_id": "5136503cc611227c0183e96598c4f706"
  }
]
```

**GlideQuery - select(String fields)**

Returns the results of the query as a Stream object containing the specified fields.

ℹ️ **Note:** Use a terminal method in the `Stream` class to get the result of the query. For more information, see `Stream`.

You can append a flag to a field name to return the field's metadata instead of the field's value. For example, using the field name `company$DISPLAY` returns the display value of a company field. Possible flags include:

- **DISPLAY**: Returns the display value of a field.
- **CURRENCY_CODE**: Returns the currency code of a currency field. For example, `USD`.
- **CURRENCY_DISPLAY**: Returns the currency display value of a currency field. For example, `¥123.45`.
- **CURRENCY_STRING**: Returns the currency string of a currency field. For example, `JPY;123.45`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| fields | String or Array of Strings | Optional. Fields to display in the result. You can provide any number of fields as arguments, dot-walk to a desired value, or use a flag. For example:  
```java
select('first_name', 'location.city', 'company$DISPLAY');
```
or  
```java
select(['first_name', 'location.city',  
         'company$DISPLAY']);
```
|                          | Default: The system always returns the `sys_id`. |
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stream</td>
<td>Object used to interact with a stream of items such as records.</td>
</tr>
</tbody>
</table>

This example shows how to select fields to display from the query and use `$DISPLAY` to return the display value of a field.

```javascript
var stream = new global.GlideQuery('sys_user').select('first_name', 'last_name', 'company$DISPLAY').toArray(5);
gs.info(JSON.stringify(stream, null, 2));
```

**Output:**

```json
[
  {
    "first_name": "survey",
    "last_name": "user",
    "company$DISPLAY": "",
    "sys_id": "005d500b536073005e0addeeff7b12f4"
  },
  {
    "first_name": "Lucius",
    "last_name": "Bagnoli",
    "company$DISPLAY": "ACME Japan",
    "sys_id": "02826bf03710200044e0bf8bcbe5d3f"
  },
  {
    "first_name": "Jimmie",
    "last_name": "Barninger",
    "company$DISPLAY": "ACME South America",
    "sys_id": "02826bf03710200044e0bf8bcbe5d55"
  },
  {
    "first_name": "Melinda",
    "last_name": "Carleton",
    "company$DISPLAY": "ACME UK",
    "sys_id": "02826bf03710200044e0bf8bcbe5d5e"
  },
  {
    "first_name": "Jewel",
    "last_name": "Barnes"
  }
]
```
GlideQuery - selectOne(String fields)

Returns the result of the query as an Optional object containing specified fields.

Use this method when returning a single record, or to test if a record exists. If returning multiple records, use the select() method to return a Stream object.

You can append a flag to a field name to return the field's metadata instead of the field's value. For example, using the field name \texttt{company$DISPLAY} returns the display value of a company field. Possible flags include:

- \texttt{DISPLAY}: Returns the display value of a field.
- \texttt{CURRENCY\_CODE}: Returns the currency code of a currency field. For example, \texttt{USD}.
- \texttt{CURRENCY\_DISPLAY}: Returns the currency display value of a currency field. For example, \texttt{¥123.45}.
- \texttt{CURRENCY\_STRING}: Returns the currency string of a currency field. For example, \texttt{JPY;123.45}.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>String or Array of Strings</td>
<td>Optional. Fields to display in the result. You can provide any number of fields as arguments, dot-walk to a desired value, or use a flag. For example: selectOne('first_name', 'location.city', 'company$DISPLAY'); or selectOne(['first_name', 'location.city', 'company$DISPLAY']); Default: The system always returns the sys_id.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional</td>
<td>Object used to interact with a single record.</td>
</tr>
</tbody>
</table>

This example shows how to return a single record as an Optional object and display specified fields.

```javascript
var user = new global.GlideQuery('sys_user')
  .where('zip', '12345')
  .whereNotNull('last_name')
  .selectOne('first_name', 'last_name', 'company$DISPLAY')
  .get();

gs.info(JSON.stringify(user, null, 2));
```

Output:

```json
{
  "first_name": "Abel",
  "last_name": "Tuter",
  "company$DISPLAY": "ACME South America",
  "sys_id": "62826bf03710200044e0bfc8bcbe5df1"
}
```

**GlideQuery - sum(String field)**

Returns the aggregate sum of a given numeric field.

You can only use this method on fields of the following types:

- Integer
- Long
- Floating Point Number
- Double
- Currency

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Field on which to perform the operation.</td>
</tr>
</tbody>
</table>
This example shows how to return the sum of all faults in the cmdb_ci table.

```javascript
var totalFaults = new global.GlideQuery('cmdb_ci')
    .sum('fault_count')
    .orElse(0);

gs.info(JSON.stringify(totalFaults));
```

Output:

```
10
```

**GlideQuery - toGlideRecord()**

Returns a GlideRecord object that represents the current query. Returns a GlideAggregate object if the query uses the GlideQuery.aggregate() method.

After transforming the query, use the query() method in the GlideRecord or GlideAggregate classes to query the database.
GlideQuery - update(Object changes, Object selectedFields)

Updates an existing record that matches the defined conditions.

Before using this method, call the where() method to specify the conditions that a record must meet to be updated.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>changes</td>
<td>Object</td>
<td>Object containing name-value pairs to update in the record. Names must match fields in the table.</td>
</tr>
<tr>
<td>selectedFields</td>
<td>Array</td>
<td>Optional. Additional fields to return in the result. Default: The system always returns the sys_id.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional</td>
<td>Object used to interact with a single record.</td>
</tr>
</tbody>
</table>

This example shows how to update a record with new values.

```javascript
var updateRecord = new global.GlideQuery('sys_user')
    .where('sys_id', '0a826bf03710200044e0bfc8bcbe5d7a')
    .update({ city: 'Los Angeles' });
```

GlideQuery - updateMultiple(Object changes)

Updates all existing records that match the defined conditions. Returns the number of records updated.

Before using this method, call the where() method to specify the conditions that the records must meet to be updated.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>changes</td>
<td>Object</td>
<td>Object containing name-value pairs to update in the record. Names must match fields in the table.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing the number of records that were updated. Keys include:</td>
</tr>
<tr>
<td></td>
<td>• rowCount: Number of rows updated in the table.</td>
</tr>
</tbody>
</table>

This example shows how to update any records that fit a defined criteria.

```javascript
var update = new global.GlideQuery('sys_user')
  .where('active', false)
  .where('last_name', 'Griffey')
  .updateMultiple({ active: true });

gs.info(JSON.stringify(update));
```

Output:

```
{"rowCount":1}
```

**GlideQuery - where(String fieldOrQuery, String operator, Any value)**

Adds a Where clause to the query that returns values based on a given condition.

ℹ️ **Note:** Do not precede this method with the `orWhere()`, `orWhereNull()`, or `orWhereNotNull()` methods.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldOrQuery</td>
<td>String or</td>
<td>Field or another GlideQuery object used in the where clause. If passing a</td>
</tr>
<tr>
<td></td>
<td>GlideQuery</td>
<td>field, you can dot-walk to a desired value. For example, <code>company.name</code>.</td>
</tr>
<tr>
<td>operator</td>
<td>String</td>
<td>Optional. Operator used in the where clause. If you do not pass an argument,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the system uses the = operator.</td>
</tr>
<tr>
<td>value</td>
<td>Any</td>
<td>Value used in the where clause.</td>
</tr>
</tbody>
</table>
This example shows how to return records from the User table where active is true and the user last logged on after a specified date.

```javascript
var query = new global.GlideQuery('sys_user')
  .where('active', true)
  .where('last_login', '>', '2016-04-15')
  .limit(5)
  .select()
  .toArray(100)

gs.info(JSON.stringify(query, null, 2));
```

Output:

```
[
  {
    "sys_id":"b0f31e5673500010c2e7660c4cf6a711"
  },
  {
    "sys_id":"8ff5b254b33213005e3de13516a8dcf7"
  },
  {
    "sys_id":"d999e5fc77e72300454792718a10611d"
  },
  {
    "sys_id":"30ad318577ab2300454792718a10619e"
  },
  {
    "sys_id":"3883f4c0730123002728660c4cf6a754"
  }
]
```

This example shows how to return records from the Incident table where active is true and where either the priority or the severity is 1.

```javascript
// active = true AND (priority = 1 OR severity = 1)
var query = new GlideQuery('incident')
  .where('active', true)
  .where(new GlideQuery()
    .where('priority', 1)
    .or
    .where('severity', 1)
  )
  .select()
  .toArray(100)

gs.info(JSON.stringify(query, null, 2));
```
.orWhere('severity', 1))
.limit(5)
.select()
.toArray(100)

gs.info(JSON.stringify(query, null, 2));

Output:
[
  {
    "sys_id":"b0f31e5673500010c2e7660c4cf6a711"
  },
  {
    "sys_id":"8ff5b254b33213005e3de13516a8dcf7"
  },
  {
    "sys_id":"d999e5fc77e72300454792718a10611d"
  },
  {
    "sys_id":"30ad318577ab2300454792718a10619e"
  },
  {
    "sys_id":"3883f4c0730123002728660c4cf6a754"
  }
]

**GlideQuery - whereNotNull(String field)**

Returns records that do not contain a null value in a given field.

**Note:** Do not precede this method with the `orWhere()`, `orWhereNull()`, or `orWhereNotNull()` methods.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Field used in the query.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQuery</td>
<td>The query object being built.</td>
</tr>
</tbody>
</table>
This example shows how to query the User table and return results where the first_name field is not null.

```javascript
var query = new global.GlideQuery('sys_user').whereNotNull('first_name').select().limit(5).toArray(100);

gs.info(JSON.stringify(query, null, 2));
```

Output:

```
[
  {
    "sys_id":"005d500b536073005e0addee7b12f4"
  },
  {
    "sys_id":"02826bf03710200044e0bfc8bcbe5d3f"
  },
  {
    "sys_id":"02826bf03710200044e0bfc8bcbe5d55"
  },
  {
    "sys_id":"02826bf03710200044e0bfc8bcbe5d5e"
  },
  {
    "sys_id":"02826bf03710200044e0bfc8bcbe5d64"
  }
]
```

**GlideQuery - whereNull(String field)**

Returns records that contain a null value in a given field.

**Note:** Do not precede this method with the `orWhere()`, `orWhereNull()`, or `orWhereNotNull()` methods.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Field used in the query.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQuery</td>
<td>The query object being built.</td>
</tr>
</tbody>
</table>

This example shows how to query the User table and return records where the first or last names are null.

```javascript
var query = new global.GlideQuery('sys_user')
    .whereNull('last_name')
    .orWhereNull('first_name')
    .select()
    .toArray(100)

gs.info(JSON.stringify(query, null, 2));
```

Output:

```javascript
[
    {
        "sys_id":"1195569be81a1010f8778bda83d25585"
    },
    {
        "sys_id":"5136503cc611227c0183e96598c4f706"
    }
]
```

**GlideQuery - withAcls()**

Executes the query using the GlideRecordSecure API to securely query the database while honoring ACLs.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQuery</td>
<td>The query object being built.</td>
</tr>
</tbody>
</table>

This example shows how to execute a secure query using ACLs.
var users = new global.GlideQuery('sys_user')
    .withAcls()
    .limit(5)
    .orderByDesc('first_name')
    .select('first_name')
    .toArray(100);

gs.info(JSON.stringify(users, null, 2));

Output:

```
[
  {
    "first_name": "Zane",
    "sys_id": "16826bf03710200044e0bfc8bcbe5dbc"
  },
  {
    "first_name": "Zackary",
    "sys_id": "8a826bf03710200044e0bfc8bcbe5d69"
  },
  {
    "first_name": "Yvette",
    "sys_id": "4e826bf03710200044e0bfc8bcbe5d57"
  },
  {
    "first_name": "Winnie",
    "sys_id": "f682abf03710200044e0bfc8bcbe5d1d"
  },
  {
    "first_name": "Wilmer",
    "sys_id": "42826bf03710200044e0bfc8bcbe5d7b"
  }
]
```

**GlideQueryCondition - Global**

The GlideQueryCondition API provides additional AND or OR conditions that can be added to the current condition, allowing you to build complex queries.

Build complex queries such as:

```
category='hardware' OR category='software' AND priority='2' AND priority='1'
```

In the case of addCondition(), an implied AND is added.

This class has no constructor. A GlideQueryCondition object is returned by the following methods:
• addActiveQuery()
• addInactiveQuery()
• addJoinQuery()
• addNotNullQuery()
• addNullQuery()
• addQuery()

If there is a complicated set of AND and OR queries, a single encoded query containing all conditions simplifies the query creation. To simplify the query creation, create a query in a list view, right-click the query, and select **Copy query**. It creates a single encoded query string to return your result set. Use that string as a parameter in an `addEncodedQuery()` call.

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an `insert()`, `update()`, `deleteRecord()`, or `deleteMultiple()` method on bad query results can result in data loss.

You can set the `glide.invalid_query.returns_no_rows` system property to true to have queries with invalid encoded queries return no records.

**GlideQueryCondition - addCondition(String name, String oper, Object value)**

Adds an AND condition to the current condition.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of a field.</td>
</tr>
<tr>
<td>oper</td>
<td>String</td>
<td>(Optional) The operator for the query. If you do not specify an operator, the condition uses an equals operator.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>The value to query on.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>A reference to a GlideQueryCondition that was added to the GlideRecord.</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('incident');
var qc = now_GR.addQuery('category', 'Hardware');
qc.addCondition('category', 'Network');
now_GR.addQuery('number', 'INC0000003');
now_GR.next();
now_GR.number;
gs.info(now_GR.getEncodedQuery());
```

**Scoped equivalent**

To use the `addCondition()` method in a scoped application, use the corresponding scoped method: **Scoped GlideQueryCondition - addCondition(String name, String oper, Object value).**

**GlideQueryCondition - addOrCondition(String name, String oper, Object value)**

Appends a 2-or-3 parameter OR condition to an existing GlideQueryCondition.

`addOrCondition()` works in conjunction with any of the **Scoped GlideRecord - addQuery(String name, String operator, Object value)** methods to OR the specified query parameters to the query previously constructed using `addQuery()`.

`addOrCondition()` is typically called with three parameters; table field, operator, and comparison value. It can be called with only two parameters, table field and comparison value, such as `qc.addOrCondition('category', 'software');`. The operator in this case is assumed to be "equal to".

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Field name</td>
</tr>
<tr>
<td>oper</td>
<td>String</td>
<td>(Optional) Query operator. The available values are dependent on the data type of the value parameter. Numbers:</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>• =</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• !=</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &gt;=</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &lt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &lt;=</td>
</tr>
</tbody>
</table>

Strings (must be in upper case):

• =
• !=
• IN
• STARTSWITH
• ENDSWITH
• CONTAINS
• DOESNOTCONTAIN

<table>
<thead>
<tr>
<th>Value</th>
<th>Object</th>
<th>Value on which to query (not case-sensitive).</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Note: All passed in arrays must contain a minimum of two elements. Single element arrays are not supported.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>A reference to a GlideQueryCondition that was added to the GlideRecord.</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('incident');
var qc = now_GR.addQuery('category', 'Hardware');
qc.addOrCondition('category', 'Network');
now_GR.addQuery('number','INC0000003');
now_GR.next();
now_GR.number;
gs.info(now_GR.getEncodedQuery());
```
To group AND/OR statements such as \((\text{state} < 3 \text{ OR } \text{state} > 5) \text{ AND } (\text{priority} = 1 \text{ OR } \text{priority} = 5)\) use code similar to the following:

```javascript
var myObj = new GlideRecord('incident');
var q1 = myObj.addQuery('state', '<', 3);
q1.addOrCondition('state', '>', 5);
var q2 = myObj.addQuery('priority', 1);
q2.addOrCondition('priority', 5);
myObj.query();
```

**Scoped equivalent**

To use the `addOrCondition()` method in a scoped application, use the corresponding scoped method: `Scoped GlideQueryCondition - addOrCondition(String name, String oper, Object value)`.

**GlideQueryCondition - Scoped**

The scoped GlideQueryCondition API provides additional AND or OR conditions that can be added to the current condition, allowing you to build complex queries.

Build complex queries such as:

```
category='hardware' OR category='software' AND priority='2' AND priority='1'
```

In the case of `addCondition()`, an implied AND is added.

This class has no constructor. A GlideQueryCondition object is returned by the following methods:

- `addActiveQuery()`
- `addInactiveQuery()`
- `addJoinQuery()`
- `addNotNullQuery()`
- `addNullQuery()`
- `addQuery()`

If there is a complicated set of AND and OR queries, a single encoded query containing all conditions simplifies the query creation. To simplify the query creation, create a query in a list view, right-click the query, and select **Copy query**. It creates a single encoded query string to return your result set. Use that string as a parameter in an `addEncodedQuery()` call.

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as
including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an `insert()`, `update()`, `deleteRecord()`, or `deleteMultiple()` method on bad query results can result in data loss.

You can set the `glide.invalid_query.returns_no_rows` system property to true to have queries with invalid encoded queries return no records.

**Scoped GlideQueryCondition - addCondition(String name, String oper, Object value)**

Adds an AND condition to the current condition.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of a field.</td>
</tr>
<tr>
<td>oper</td>
<td>(Optional) The operator for the query. If you do not specify an operator, the condition uses an equals operator.</td>
</tr>
<tr>
<td>value</td>
<td>The value to query on.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>A reference to a GlideQueryCondition that was added to the GlideRecord.</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('incident');
var qc = now_GR.addQuery('category', 'Hardware');
qc.addCondition('category', 'Network');
now_GR.addQuery('number','INC0000003');
now_GR.next();
now_GR.number;
gs.info(now_GR.getEncodedQuery());
```

**Scoped GlideQueryCondition - addOrCondition(String name, String oper, Object value)**

Appends a two-or-three parameter OR condition to an existing GlideQueryCondition.
addOrCondition() works in conjunction with any of the `Scoped GlideRecord.addQuery(String name, String operator, Object value)` methods to OR the specified query parameters to the query previously constructed using `addQuery()`.

`addOrCondition()` is typically called with three parameters: table field, operator, and comparison value. It can be called with only two parameters, table field and comparison value, such as `qc.addOrCondition('category', 'software');`. The operator in this case is assumed to be "equal to".

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Field name</td>
</tr>
</tbody>
</table>
| oper | String| (Optional) Query operator. The available values are dependent on the data type of the value parameter. Numbers:  
  • =  
  • !=  
  • >  
  • >=  
  • <  
  • <=  
  Strings (must be in upper case):  
  • =  
  • !=  
  • IN  
  • STARTSWITH  
  • ENDSWITH  
  • CONTAINS  
  • DOESNOTCONTAIN |
| value| Object| Value on which to query (not case-sensitive). |

**Note:** All passed in arrays must contain a minimum of two elements. Single element arrays are not supported.
### GlideRecord - Client

GlideRecord is used for database operations. The client-side GlideRecord API enables the use of some GlideRecord functionality in client-side scripts, such as client scripts and UI policy scripts.

A GlideRecord contains both records and fields. Queries made with the client-side GlideRecord are executed on the server. Therefore, a request is made from the client browser to obtain the record data.

The client-side GlideRecord API is not supported in scoped applications. Instead, create a script include and use the GlideAjax API, or use the REST APIs. In addition, the client-side GlideRecord API applies ACLs based on the credentials of the user executing the script. To execute the code on the server without ACLs, use the GlideAJAX API.

### Client side GlideRecord - addOrderBy(String column)

Adds a column to order by in the query.

---

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>A reference to a GlideQueryCondition that was added to the GlideRecord.</td>
</tr>
</tbody>
</table>
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>column</td>
<td>String</td>
<td>The column by which to order the result set.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### Client side GlideRecord - `addQuery(String name, Object value)`

Adds a query to return records where the specified field name is equal to a specified value (or is in a list of values).

If you are familiar with SQL, this method is similar to the "where" clause. You can create one or more queries for a single filter by calling this method multiple times; for this method the queries are AND'ed. Once you define all of the desired queries, call the `Client side GlideRecord - query(Function responseFunction)` to execute the specified query clause (filter).

To perform an operation other than AND, use either the `addQuery(String name, Object operator, Object value)` method or the `setEncodedQuery()` method.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field to check.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>Value on which to query.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### Example showing multiple queries.

```javascript
var now_GR = new GlideRecord('incident');
now_GR.addQuery('priority', 4); // Priority is 4 – Low and,
now_GR.addQuery('state', 3); // State is On Hold
now_GR.query(response);
```
```javascript
function response(result) {
    while(result.next()) {
        // Print all INC with priority 4 – Low AND state is On Hold
        console.log(result.getValue('number'));
    }
}
```

Example showing how to pass a string object.

```javascript
var now_GR = new GlideRecord('incident');
now_GR.addQuery('short_description', new String('USB device not working')); // string
now_GR.addQuery('priority', 4); // number
now_GR.query(response);
```

`Client side GlideRecord - addQuery(String name, Object operator, Object value)`

Adds a filter to return records where the field meets the specified condition (field, operator, value).

If you are familiar with SQL, this method is similar to the “where” clause. You can create one or more queries for a single filter by calling this method multiple times. Once you define all of the desired queries, call the `Client side GlideRecord - query(Function responseFunction)` to execute the specified query clause (filter).

To create more complex queries, use the `setEncodedQuery()` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field to check.</td>
</tr>
<tr>
<td>operator</td>
<td>Object</td>
<td>Query operator. The available values are dependent on the data type of the value parameter.</td>
</tr>
</tbody>
</table>
**Parameters (continued)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

**Numbers:**
- `=`
- `!=`
- `>`
- `>=`
- `<`
- `<=`

**Strings (must be in upper case):**
- `=`
- `!=`
- `IN`
- `NOT IN`
- `STARTSWITH`
- `ENDSWITH`
- `CONTAINS`
- `DOES NOT CONTAIN`
- `INSTANCEOF`

<table>
<thead>
<tr>
<th>value</th>
<th>Object</th>
<th>Value on which to query (not case-sensitive).</th>
</tr>
</thead>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void</code></td>
<td></td>
</tr>
</tbody>
</table>

Example showing how to add multiple queries to a filter.

```javascript
var now_GR = new GlideRecord('incident');
now_GR.addQuery('priority', '<=', 2); // Priority is 2 or higher and,
now_GR.addQuery('short_description', 'CONTAINS', 'crash'); // Short description contains the word crash
now_GR.query(response);
```
function response(result) {
    while(result.next()) {
        // Print all INC with priority of 2 or higher AND short description contains "crash"
        console.log(result.getValue('number'));
    }
}

Example showing how to pass in an array to verify multiple conditions in a single query.

var priorities = [4,2];
var now_GR = new GlideRecord('incident');
now_GR.addQuery('priority', 'IN', priorities);
now_GR.query(response);

function response(result) {
    while(result.next()) {
        console.log(result.getValue('number'));
    }
}

Client side GlideRecord - deleteRecord(Function responseFunction)
Deletes the current record and calls the specified response function when complete.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>responseFunction</td>
<td>Function</td>
<td>Response function for the Ajax callback.</td>
</tr>
</tbody>
</table>

This example deletes a record and then calls the response function response to log an alert message.

var recordGR = new GlideRecord('incident');
if (recordGR.get('99ebb4156fa831005be8883e6b3ee4b9')) {
    recordGR.deleteRecord(response);
}
function response(result) {
    alert('Deleted record sys_id: ' + result.getValue('sys_id'));
}

Output

Deleted record sys_id: 99ebb4156fa831005be8883e6b3ee4b9

**Client side GlideRecord - get(Object sys_id)**

Executes a GlideRecord query for a record with the specified sys_id. This method is expected to be used to query for single records, so a next operation is performed before returning.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>Object</td>
<td>The sys_id of the record to be found.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if one or more matching records was found. False if no records were found.</td>
</tr>
</tbody>
</table>

**Client side GlideRecord - getEncodedQuery()**

Retrieves the query condition of the current result set as an encoded query string.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The encoded query as a string.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Client side GlideRecord - getLimit()
Returns the limit for records to be returned by the GlideRecord query.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The limit for records to be returned by the GlideRecord query.</td>
</tr>
</tbody>
</table>

Client side GlideRecord - getTableName()
Retrieves the name of the table associated with this GlideRecord.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The table name</td>
</tr>
</tbody>
</table>

```javascript
var item = new GlideRecord('sc_request');
item.addQuery('sys_id', current.request);
item.query(itemResponse);

function itemResponse(item) {
    alert('The table is ' + item.getTableName());
}
```

Client side GlideRecord - GlideRecord(String tableName)
Creates an instance of the GlideRecord class for the specified table.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>The table to be used.</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('incident');
```

### Client side GlideRecord - hasNext()

Determines if there are any more records in the GlideRecord.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if there are more records in the query set.</td>
</tr>
</tbody>
</table>

### Client side GlideRecord - insert(Function responseFunction)

Inserts a new record using the field values that have been set for the current record.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>responseFunction</td>
<td>Function</td>
<td>Function to execute once the record is inserted.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the inserted record, or null if the record was not inserted.</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('incident');
now_GR.short_description = 'Learn about GlideRecord';
var recResponse = now_GR.insert(handleResponse);
```
function handleResponse(recResponse, answer) {
    // Answer will be the sys_id of the created record or null
    alert('Newly created sys_id is - ' + answer + ' exists');
}

**Client side GlideRecord - next()**

Moves to the next record in the GlideRecord.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>False if there are no more records in the query set.</td>
</tr>
</tbody>
</table>

```javascript
var rec = new GlideRecord('incident');
rec.query(recResponse);

function recResponse(rec) {
    while (rec.next()) {
        alert(rec.number + ' exists');
    }
}
```

**Client side GlideRecord - orderBy(String column)**

Specifies an orderBy column. May be called more than once to order by multiple columns.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>column</td>
<td>String</td>
<td>The column name to be used to order the result set.</td>
</tr>
</tbody>
</table>
function UpdateProjectWBS(project) {
    var count = 0;
    var child = new GlideRecord('pm_project_task');
    child.addQuery('parent', project.sys_id);
    child.orderBy('order');
    child.orderBy('number');
    child.query(childResponse);
    g_form.addInfoMessage(count + ' Project Tasks updated');
}

function childResponse(child) {
    var len = child.getRowCount().toString().length;
    var seq = 0;
    while (child.next()) {
        count += UpdateProjectTaskWBS(child, 1, ++seq, len, ' ');
    }
}

Client side GlideRecord - query(Function responseFunction)
Runs the query against the table based on the addQuery() filter. This queries the GlideRecord table as well as any references of the table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>responseFunction</td>
<td>Function</td>
<td>The response function for the Ajax callback.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
Client side GlideRecord - setEncodedQuery(String encodedQuery)
Adds a specified encoded query string to the current query clause.

This method enables you to specify complex filters (encoded query strings) in a single query call, unlike other client-side addQuery() methods. Once you define all of the desired queries, call the Client side GlideRecord - query(Function responseFunction) to execute the specified query clause (filter). For additional information on encoded query strings, refer to Encoded query strings.

If you call this method multiple times before calling the query() method, the queries are AND'ed together.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>encodedQuery</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('incident');
now_GR.setEncodedQuery('priority=4^ORstate=3');
now_GR.query(response);

function response(result) {
    while(result.next()) {
        console.log(result.getValue('number'));
    }
}
```
Client side GlideRecord - setLimit(Number maxQuery)

Sets the limit for how many records are in the GlideRecord.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxQuery</td>
<td>Number</td>
<td>The limit for the number of records to retrieve.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideRecord - Global**

The GlideRecord API is used for database operations.

The GlideRecord API is the primary means of interfacing with the database on the server-side code. A GlideRecord is an object that contains records from a single table. Use the API to instantiate a GlideRecord object and add query parameters, filters, limits, and ordering.

For information about GlideRecordSecure, which is a class inherited from GlideRecord that performs the same functions as GlideRecord, and also enforces ACLs, see the GlideServer APIs.

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an insert(), update(), deleteRecord(), or deleteMultiple() method on bad query results can result in data loss.

You can set the glide.invalid_query.returns_no_rows system property to true to have queries with invalid encoded queries return no records.

**Retrieve values from records**

In most cases, do not use dot-walking to get values from a record. Dot-walking retrieves the entire object instead of the field value. Retrieving the object uses much more storage and might cause undesirable results when used in arrays or in Service Portal.
Instead of retrieving the entire object, you can use one of the following methods to copy the field values:

- `getValue()`
- `getDisplayValue()`

If dot-walking through a `GlideElement` object is necessary, use the `toString()` method to retrieve values. For example, you might need the current caller's manager sys_id to set another reference field. The following example shows how to get the string value instead of the entire object:

```javascript
var mgr = current.caller_id.manager.toString();
```

See also:
- `GlideAggregate`
- `GlideElement`
- `GlideQuery - Scoped, Global`

**GlideRecord - addActiveQuery()**

Adds a filter to return active records.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>GlideQueryCondition</td>
<td>Filter to return active records.</td>
</tr>
</tbody>
</table>

```javascript
var inc = new GlideRecord('incident');
inc.addActiveQuery();
inc.query();
```

**Scoped equivalent**

To use the `addActiveQuery()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - addActiveQuery()`.

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.

Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
GlideRecord - addDomainQuery(Object glideRecord)

Changes the domain used for the query from the user's domain to the domain of the provided GlideRecord.

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an `insert()`, `update()`, `deleteRecord()`, or `deleteMultiple()` method on bad query results can result in data loss.

You can set the `glide.invalid_query.returns_no_rows` system property to true to have queries with invalid encoded queries return no records.

This function requires the Domain Support - Domain Extensions Installer (com.glide.domain.msp_extensions.installer) plugin.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>glideRecord</td>
<td>Object</td>
<td>GlideRecord from which to obtain the domain.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
//This example requires the Domain plugin be active, the Group table is the specified Domain table, and the ITIL user is in the Database Atlanta domain. //From any domain (using queryNoDomain()) look up the incidents that an ITIL user can only see //who is in the Database Atlanta domain, should expect all incidents with the global or the //Database Atlanta domain specified.
var domain = new GlideRecord('sys_user');
domain.addQuery('user_name', 'itil');
domain.queryNoDomain();
if (domain.next()) {
    var domainQuery = new GlideRecord('incident');
domainQuery.addDomainQuery(domain);
domainQuery.query();
gs.print('Number of Incidents for ITIL user: ' + domainQuery.getRowCount());
```
while (domainQuery.next())
    gs.print(domainQuery.number);
}

Scoped equivalent
This method is not available in scoped applications.

GlideRecord - addEncodedQuery(String query)
Adds an encoded query to other queries that may have been set.

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an insert(), update(), deleteRecord(), or deleteMultiple() method on bad query results can result in data loss.

You can set the glide.invalid_query.returns_no_rows system property to true to have queries with invalid encoded queries return no records.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>String</td>
<td>An encoded query string.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var queryString = "priority=1^ORpriority=2";
var now_GR = new GlideRecord('incident');

now_GR.addEncodedQuery(queryString);
now_GR.query();
while (now_GR.next()) {
    gs.addInfoMessage(now_GR.number);
}
Scoped equivalent

To use the `addEncodedQuery()` method in a scoped application, use the corresponding scoped method: Scoped GlideRecord - `addEncodedQuery(String query)`.

GlideRecord - addFunction(Object function)

Applies a pre-defined GlideDBFunctionBuilder object to a record.

Use the GlideDBFunctionBuilder scoped class to define a function. After the function is defined, use the `addFunction(Object function)` method to apply the function to a record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>function</td>
<td>Object</td>
<td>A GlideDBFunctionBuilder object that defines a SQL operation.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var functionBuilder = new GlideDBFunctionBuilder();
var myAddingFunction = functionBuilder.add();
myAddingFunction = functionBuilder.field('order');
myAddingFunction = functionBuilder.field('priority');
myAddingFunction = functionBuilder.build();

var now_GR = new GlideRecord('incident');
now_GR.addFunction(myAddingFunction);
now_GR.addQuery(myAddingFunction, '<', 5);
now_GR.query();
while(now_GR.next())
gs.log(now_GR.getValue(myAddingFunction));
```

Output:

```plaintext
*** Script: 1
*** Script: 4
*** Script: 3
*** Script: 1
```
GlideRecord - addInactiveQuery()

Adds a filter to return inactive records. Inactive records have the active flag set to false.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>Records where the active flag is false.</td>
</tr>
</tbody>
</table>

```javascript
var inc = new GlideRecord('incident');
inc.addInactiveQuery();
inc.query();
```

### Scoped equivalent

In scoped applications use the scoped method `Scoped GlideRecord - addQuery(String name, Object value)`.

GlideRecord - addJoinQuery(String table)

Adds a filter to return records based on a relationship in a related table.

For example, find all the users that are in the database group (users via `sys_user_grmember` table). Another example would be find all problems that have an assigned incident (problems via the `incident.problem_id` relationship).

This is not a true database join; rather, `addJoinQuery()` adds a subquery. So, while the result set is limited based on the join, the only fields that you have access to are those on the base table (those which are in the table with which the GlideRecord was initialized).
Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an `insert()`, `update()`, `deleteRecord()`, or `deleteMultiple()` method on bad query results can result in data loss.

You can set the `glide.invalid_query.returns_no_rows` system property to true to have queries with invalid encoded queries return no records.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>table</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>GlideQueryCondition</td>
</tr>
</tbody>
</table>

Find problems that have an incident attached. This example returns problems that have associated incidents. However, it won't pull values from the incidents that are returned as a part of the query.

```javascript
var prob = new GlideRecord('problem');
prob.addJoinQuery('incident');
prob.query();
```

Find active=false problems with associated incidents.

```javascript
// Look for Problem records
var now_GR = new GlideRecord('problem');

// That have associated Incident records
var grSQ = now_GR.addJoinQuery('incident');

// Where the Problem records are "active=false"
now_GR.addQuery('active', 'false');

// And the Incident records are "active=true"
grSQ.addCondition('active', 'true');

// Query
```
now_GR.query();

// Iterate and print results
while (now_GR.next()) {
    gs.print(now_GR.getValue('number'));
}

**Scoped equivalent**

To use the `addJoinQuery()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord – addJoinQuery(String joinTable, String primaryField, String joinTableField)**.

**GlideRecord - addJoinQuery(String table, String primaryField)**

Adds a filter to return records based on a relationship in a related table.

For example, find all the users that are in the database group (users via sys_user_grmember table). Another example would be find all problems that have an assigned incident (problems via the incident.problem_id relationship).

This is not a true database join; rather, `addJoinQuery()` adds a subquery. So, while the result set is limited based on the join, the only fields that you have access to are those on the base table (those which are in the table with which the GlideRecord was initialized).

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an `insert()`, `update()`, `deleteRecord()`, or `deleteMultiple()` method on bad query results can result in data loss.

You can set the `glide.invalid_query.returns_no_rows` system property to true to have queries with invalid encoded queries return no records.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>Table name</td>
</tr>
<tr>
<td>primaryField</td>
<td>String</td>
<td>If other than sys_id, the primary field.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>Records where the relationships match.</td>
</tr>
</tbody>
</table>

Find problems that have incidents using the open_by field at the join key instead of the sys_id.

```javascript
var now_GR = new GlideRecord('problem');
now_GR.addJoinQuery('incident', 'opened_by');
now_GR.query();
```

**Scoped equivalent**

To use the `addJoinQuery()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord – addJoinQuery(String joinTable, String primaryField, String joinTableField)`.

**GlideRecord - addJoinQuery(String table, String primaryField, String joinTableField)**

Adds a filter to return records based on a relationship in a related table.

For example, find all the users that are in the database group (users via `sys_user_grmember` table). Another example would be find all problems that have an assigned incident (problems via the incident.problem_id relationship).

This is not a true database join; rather, `addJoinQuery()` adds a subquery. So, while the result set is limited based on the join, the only fields that you have access to are those on the base table (those which are in the table with which the GlideRecord was initialized).

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an `insert()`, `update()`, `deleteRecord()`, or `deleteMultiple()` method on bad query results can result in data loss.

You can set the `glide.invalid_query_returns_no_rows` system property to true to have queries with invalid encoded queries return no records.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>Table name</td>
</tr>
<tr>
<td>primaryField</td>
<td>String</td>
<td>If other than sys_id, the primary field.</td>
</tr>
<tr>
<td>joinTableField</td>
<td>String</td>
<td>If other than sys_id, the field that joins the tables</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>Records where the relationships match.</td>
</tr>
</tbody>
</table>

Find problems that have incidents associated where the incident caller_id field value matches that of the problem opened_by field.

```javascript
var now_GR = new GlideRecord('problem');
now_GR.addJoinQuery('incident', 'opened_by', 'caller_id');
now_GR.query();
```

### Scoped equivalent

To use the `addJoinQuery()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord – addJoinQuery(String joinTable, String primaryField, String joinTableField)**.

### GlideRecord - addNotNullQuery(String fieldName)

Adds a filter to return records where the specified field is not null.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field name.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>GlideQueryCondition of records where the parameter field is not null.</td>
</tr>
</tbody>
</table>
var target = new GlideRecord('incident');
    target.addNotNullQuery('short_description');
    target.query(); // Issue the query to the database to get all records
    while (target.next()) {
        // add code here to process the incident record
    }

Scoped equivalent

To use the `addNotNullQuery()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - addNotNullQuery(String fieldName)`.

**GlideRecord - addNullQuery(String fieldName)**

Adds a filter to return records where the specified field is null.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field name.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>GlideQueryCondition of records where the specified field is null.</td>
</tr>
</tbody>
</table>

var target = new GlideRecord('incident');
    target.addNullQuery('short_description');
    target.query(); // Issue the query to the database to get all records
    while (target.next()) {
        // add code here to process the incident record
    }

Scoped equivalent

To use the `addNullQuery()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - addNullQuery(String fieldName)`.
GlideRecord - addQuery(String name, Object operator, Object value)

Provides the ability to build a request, which when executed, returns the rows from the specified table that match the request.

If you are familiar with SQL, this method is similar to the "where" clause. One or more addQuery() calls can be made in a single query; for this method the queries are AND'ed. If any of the query statements need to be OR'ed, use the class GlideQueryCondition - Global.

addQuery() is typically called with three parameters; table field, operator, and comparison value. It can be called with only two parameters, table field and comparison value, such as myObj.addQuery('category', 'Hardware');. The operator in this case is assumed to be "equal to".

For additional information on using queries, see Querying tables in script.

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an insert(), update(), deleteRecord(), or deleteMultiple() method on bad query results can result in data loss.

You can set the glide.invalid_query.returns_no_rows system property to true to have queries with invalid encoded queries return no records.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Table field name</td>
</tr>
</tbody>
</table>
| operator | Object   | Query operator. The available values are dependent on the data type of the value parameter. Numbers:  
|         |            | • =  
|         |            | • !=  
|         |            | • >  
|         |            | • >=  
|         |            | • <  
|         |            | • <=  
|         |            | Strings (must be in upper case):                |
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• =</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• !=</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• IN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• NOT IN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• STARTSWITH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• ENDSWITH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• CONTAINS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• DOES NOT CONTAIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• INSTANCEOF</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

value | Object | Value on which to query (not case-sensitive).

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>A reference to the GlideQueryCondition that was added to the GlideRecord.</td>
</tr>
</tbody>
</table>

var rec = new GlideRecord('incident');
rec.addQuery('active', true);
rec.addQuery('sys_created_on', '>', '2010-01-19 04:05:00');
rec.query();
while (rec.next()) {
    rec.active = false;
    gs.print('Active incident ' + rec.number + ' closed');
    rec.update();
}

Using the IN operator.

var que = new GlideRecord('incident');
que.addQuery('number', 'IN', 'INC00001,INC00002');
que.query();
while (que.next()) {
    //do something....
}

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Scoped equivalent

To use the `addQuery()` method in a scoped application, use the corresponding scoped method: Scoped GlideRecord - `addQuery(String name, String operator, Object value)`.

GlideRecord - `addValue(String field, Number value)`

Provides atomic add and subtract operations on a specified number field at the database level for the current GlideRecord object.

Typically, a GlideRecord object is written as one record in a database. Individual field values are stored as defined. For code that adds a value to a GlideRecord field, it simply saves the field to the database with the new value, rather than atomically incrementing it.

For example, when the following code is executed, the value of the `u_count` field in the database is 2.

```java
gs.print(now_now_GR.u_count); // "1"
now_GR.u_count += 1;
now_GR.update();
now_GR.get(now_now_GR.sys_id);
gs.print(now_now_GR.u_count); // "2"
```

If another user concurrently runs identical code, instead of the two operations each adding 1 to `u_count`, the net effect is that `u_count` only contains 2, with one operation's update actually being lost.

Conversely, the `addValue()` method performs the addition/subtraction in the database when the record is updated as an atomic operation. Two operations running concurrently each properly update the field.

```java
gs.print(now_now_GR.u_count); // "1"
now_GR.addValue("u_count", 1);
now_GR.update();
now_GR.get(now_GR); // The record must be reloaded from the database to observe the result
gs.print(now_now_GR.u_count); // "3", if executed concurrently with another user
```

⚠️ Note: The new value is not read back from the database unless explicitly done so.

Like `setValue()`, `addValue()` changes only take effect in the database after a subsequent call to `update()` or `insert()`. If `insert()` is called, the specified field is initialized with the `value` parameter passed into `addValue()`.

⚠️ Note: If `setValue()` is called for the specified field prior to calling `addValue()`, the `addValue()` method is not processed and an error message is logged.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>The name of the field in this GlideRecord to modify. If the associated field is not a numeric type, the operation is ignored.</td>
</tr>
<tr>
<td>value</td>
<td>Number</td>
<td>The amount to add to the value when the record is saved. To perform a subtraction operation, simply pass a negative value.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Shows a value being added.

```java
gs.print(now_GR.u_count); // "1"
now_GR.addValue("u_count", 1);
now_GR.update();
now_GR.get(now_GR.sys_id); // The record must be reloaded from the database to observe the result
gs.print(now_GR.u_count);
```

Output: 2

Shows a value being subtracted.

```java
gs.print(now_GR.u_count); // "4"
now_GR.addValue("u_count", -1);
now_GR.update();
now_GR.get(now_GR.sys_id); // The record must be reloaded from the database to observe the result
gs.print(now_GR.u_count);
```

Output: 3

**GlideRecord - applyEncodedQuery(String queryString)**

Sets the values of the specified encoded query terms and applies them to the current GlideRecord.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queryString</td>
<td>String</td>
<td>Encoded query to apply to the current GlideRecord.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
function createAcl(table, role) {
  gs.print("Checking security on table " + table);
  var now_GR = new GlideRecord("sys_security_acl");
  now_GR.addQuery("name", table);
  now_GR.addQuery("operation", "read");
  now_GR.query();
  var encQuery = now_GR.getEncodedQuery();
  if (now_GR.next()) {
    // existing acl found so use it
    createAclRole(now_GR.sys_id.toString(), role);
    return;
  } else {
    now_GR.initialize();
    now_GR.applyEncodedQuery(encQuery);
    var acl = now_GR.insert();
    gs.print("Added read access control on " + table);
    createAclRole(acl, role);
  }
}
```

### GlideRecord - applyTemplate(String template)

Apply a template record from the Template table [sys_template] to the current record. If the specified template is not found, no action is taken.

**Note:** This method automatically instantiates a `now_GR.insert()` method if a template has the **Next Related Child Template** field filled. For information, see Create templates for related task records.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>template</td>
<td>String</td>
<td>Name of a template from the Templates [sys_template] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var rec1 = new GlideRecord("incident");
rec1.initialize();
rec1.applyTemplate("my_incident_template");
rec1.insert();
```

**GlideRecord - autoSysFields(Boolean e)**

Enables or disables the update to the fields sys_updated_by, sys_updated_on, sys_mod_count, sys_created_by, and sys_created_on. This is often used for manually updating field values on a record while leaving historical information unchanged.

ℹ️ **Note:** This is not available for scoped apps, starting with the Fuji release. See the Scoped GlideRecord API Reference for a list of what APIs are available for scoped apps.

⚠️ **CAUTION:** Use caution if you use this method. When you use this method the sys_mod_count field will not be incremented, and other sys_ fields will not be updated. This can break functionality including, but not limited to, the Activity Formatter, History Sets, Notifications, and Metrics.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>e</td>
<td>Boolean</td>
<td>If false disables updates to sys_updated_by, sys_updated_on, sys_mod_count, sys_created_by, and sys_created_on.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var inc = new GlideRecord('incident');

// Change all Open(1) incidents to Active(2)
inc.addQuery('state', 1);
inc.query();

while (inc.next()) {
    inc.autoSysFields(false);  // Do not update sys_updated_by, sys_updated_on,
                               // sys_mod_count, sys_created_by, and sys_created_on
    inc.setWorkflow(false);    // Do not run any other business rules
    inc.setValue('state', 2);
    inc.update();
}
```

**GlideRecord - canCreate()**

Determines if the access control rules (which includes the user's role) permit inserting new records in this table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the user's roles permit creating of records in this table. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Creating permitted</td>
</tr>
<tr>
<td></td>
<td>• false: Creating is not permitted</td>
</tr>
</tbody>
</table>

This example shows whether records can be created on the benefit_plan table.
canCreateBenefitPlan : function() {
    var now_GR = new GlideRecord('benefit_plan');
    return now_GR.canCreate();
}

Scoped equivalent
To use the canCreate() method in a scoped application, use the corresponding scoped method: Scoped GlideRecord - canCreate().

GlideRecord - canDelete()
Determines if the access control rules (which includes the user's role) permit deletion of records in this table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean  | Flag that indicates whether the user's roles permit deleting of records in this table. Valid values:  
• true: Deleting permitted  
• false: Deleting is not permitted |

Example

```javascript
var att = new GlideRecord('sys_attachment');
att.get('${sys_attachment.sys_id}');
var sm = GlideSecurityManager.get();
var checkMe = 'record/sys_attachment/delete';
var canDelete = sm.hasRightsTo(checkMe, att);
gs.log('canDelete: ' + canDelete);
```

Scoped equivalent
To use the canDelete() method in a scoped application, use the corresponding scoped method: Scoped GlideRecord - canDelete().
**GlideRecord - canRead()**

Determines if the access control rules (which includes the user's role) permit reading this table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the user's roles permit reading of records in this table. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Reading permitted</td>
</tr>
<tr>
<td></td>
<td>• false: Reading is not permitted</td>
</tr>
</tbody>
</table>

This example shows whether records can be read from the benefit_plan table.

```javascript
canReadBenefitPlan: function() {
    var now_GR = new GlideRecord('benefit_plan');
    return now_GR.canRead();
}
```

**Scoped equivalent**

To use the `canRead()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - canRead()**.

**GlideRecord - canWrite()**

Determines if the access control rules (which includes the user's role) permit updates to records in this table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates whether the user’s roles permit writing of records in this table. Valid values:  
• true: Writing permitted  
• false: Writing is not permitted |

This example shows whether records can be written to the benefit_plan table.

canWriteBenefitPlan : function() {
    var now_GR = new GlideRecord('benefit_plan');
    return now_GR.canWrite();
}

 Scoped equivalent

To use the `canWrite()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - canWrite()`.

GlideRecord - changes()

Determines whether any of the fields in the record have changed.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates whether there are changes in the specified record. Possible values:  
• true: Fields in the specified record have changed.  
• false: No fields have changed. |

This example shows how to check if there are changes in an incident record.
var now_GR = new GlideRecord("incident");
now_GR.query();
now_GR.next();
if (now_GR.changes()) {
    gs.print("The incident record reported changes right after being read");
} else {
    gs.print("The incident record has not changed");
}

**Scoped equivalent**
To implement this functionality in a scoped application, add code similar to the following:

```javascript
var now_GR = new GlideRecord("incident");
now_GR.get("965c9e5347c12200e0ef563dbb9a7156");
now_GR.short_description = "test";
var elements = now_GR.getElements();
var hasChanged = false;
for(var i=0; i < elements.length;i++){
    var element = elements[i];
    hasChanged = hasChanged || element.changes();
    gs.info(element.getName() + " : " + element.changes());
}
gs.info(hasChanged);
```

**GlideRecord - deleteMultiple()**
Deletes multiple records according to the current "where" clause.

This method does not delete attachments.

Dot-walking is not supported for this method. When using the `deleteMultiple()` function on referenced tables, all the records in the table are deleted. Also, when using `deleteRecord()` to cascade delete, prior calls to `setWorkflow()` on the same GlideRecord object are ignored.

Do not use `deleteMultiple()` on tables with currency fields. Always delete each record individually. Also, do not use this method with the `chooseWindow()` or `setLimit()` methods when working with large tables.

<p>| Parameters |
|---|---|---|
| Name | Type | Description |
| None | | |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the record was successfully deleted. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Record was deleted.</td>
</tr>
<tr>
<td></td>
<td>• false: No record was found to delete.</td>
</tr>
</tbody>
</table>

This example shows how to delete a record from the Incident table.
var rec = new GlideRecord('incident');
rec.addQuery('active',false);
rec.query();
while (rec.next()) {
    gs.print('Inactive incident ' + rec.number + ' deleted');
    rec.deleteRecord();
}

Scoped equivalent

To use the `deleteRecord()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - deleteRecord()`.

**GlideRecord - find(String columnName, String value)**

Returns true if any record has a matching value in the specified column. If found, it also moves to the first record that matches, essentially executing `next()` until the record is returned.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columnName</td>
<td>String</td>
<td>Field name to search.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value to check for in the specified field.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether any record within the current table has a matching value in the specified field. Possible values:</td>
</tr>
<tr>
<td></td>
<td>- true: Matching field.</td>
</tr>
<tr>
<td></td>
<td>- false: No matching field.</td>
</tr>
</tbody>
</table>

This example shows how to find a record in the Incident table with "Critical" in the short description field.

```javascript
var now_GR = new GlideRecord("incident");
now_GR.query();
var shortDescription = "Critical";
if (now_GR.find("short_description", shortDescription)) {
    gs.print("An incident with the specified field value was found");
}
```
var recordID = now_GR.getValue("sys_id");
gs.print("Found in the following record: " + recordID);
} else {
    gs.print("An incident with the specified field value was not found");
}

Output
An incident with the specified field value was found
Found in the following record: 552c48888c033300964f4932b03eb092

GlideRecord - get(Object name, Object value)

Returns the specified record in the current GlideRecord object.

This method accepts either one or two parameters. If only a single parameter is passed in, the method assumes that it is the sys_id of the desired record. If not found, it then tries to match the value against the display value. If two parameters are passed in, the first is the name of the column within the GlideRecord to search. The second is the value to search for.

If multiple records are found, use `next()` to access the additional records.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Object</td>
<td>Optional. Name of the instantiated GlideRecord column to search for the specified value parameter. If only a single parameter is passed in, the method assumes that this parameter is sys_id.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>Value to match.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Indicates whether the requested record was located. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Record was found</td>
</tr>
<tr>
<td></td>
<td>• false: Record was not found</td>
</tr>
</tbody>
</table>

This example gets an incident record by passing in the sys_id.
```javascript
var grIncident = new GlideRecord('incident');
var returnValue = grIncident.get('99ebb4156fa831005be8883e6b3ee4b9');
gs.info(returnValue); // logs true or false
gs.info(grIncident.number); // logs Incident Number
```

This example gets an incident record by passing in the column in the record to search and the value to search for.

```javascript
var grIncident = new GlideRecord('incident');
var returnValue = grIncident.get('caller_id.name','Sylivia Wayland');
gs.info(returnValue); // logs true or false
gs.info(grIncident.number); // logs Incident Number
```

**Scoped equivalent**

To use the `get()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - get(Object name, Object value)**.

**GlideRecord - getAttribute(String fieldName)**

Returns the dictionary attributes on the specified field.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Field name for which to return the dictionary attributes</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Dictionary attributes</td>
</tr>
</tbody>
</table>

This example shows how to return the dictionary attributes for “tree_picker”.

```javascript
function doit() {
  var now_GR = new GlideRecord('sys_user');
  now_GR.query("user_name","admin");
  if (now_GR.next()) {
    gs.print("we got one");
    gs.print(now_GR.location.getAttribute("tree_picker"));
  }
}
```
Scoped equivalent

To use the `getAttribute()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - getAttribute(String fieldName).**

GlideRecord - `getClassDisplayValue()`

Returns the table's label.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

This example prints the label of the passed in table.

```java
// Display the incident table label
var now_GR = new GlideRecord("incident");
var value = now_GR.getClassDisplayValue();
gs.info("The table label is " + value + ";");
```

Output:

The table label is Incident.

Scoped equivalent

To use the `getClassDisplayValue()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - getClassDisplayValue().**

GlideRecord - `getDisplayValue()`

Retrieves the display value for the current record.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
<tr>
<td>Returns</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

This example writes the display value of a specified incident record into the log.

```javascript
var now_GR = new GlideRecord('incident');
now_GR.get('sys_id','<sys_id>');
gs.info(now_GR.getDisplayValue());
```

Output:

```
INC0000050
```

**Scoped equivalent**

To use the `getDisplayValue()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - getDisplayValue()`.

**GlideRecord - getED()**

Returns the element's descriptor.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to retrieve the name field for an incident GlideRecord.

```javascript
grInc = new GlideRecord('incident');
grInc.getActiveQuery();
grInc.query();
if (grInc.next())
    gs.info("Table Descriptor is - " + grInc.getED().getName());
```

Output:

```
"Table Descriptor is - " + grInc.getED().getName()
```
**Scoped equivalent**

To use the `getED()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - getED()**.

**GlideRecord - getElement(String fieldName)**

Retrieves the GlideElement for a specified field.

The value returned by this method is a complete GlideElement object. The results are the equivalent of dot-walking a field value. For example, `now_GR.getElement('short_description')` provides the same result as `nowGR.short_description`.

In most cases, do not use dot-walking to get values from a record. Dot-walking retrieves the entire object instead of the field value. Retrieving the object uses much more storage and might cause undesirable results when used in arrays or in Service Portal.

Instead of retrieving the entire object, you can use one of the following methods to copy the field values:

- `getValue()`
- `getDisplayValue()`

If dot-walking through a GlideElement object is necessary, use the `toString()` method to retrieve values. For example, you might need the current caller's manager sys_id to set another reference field. The following example shows how to get the string value instead of the entire object:

```java
var mgr = current.caller_id.manager.toString();
```

See also: **GlideElement - toString()**.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Column name for which to return the GlideElement object.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideElement</td>
<td>The GlideElement for the specified column of the current record. Each object describes a field in the current GlideRecord.</td>
</tr>
</tbody>
</table>

This example shows how to obtain the GlideElement object for a passed in table and compare them with the expected system metadata values.

```javascript
function compareElement(comment, table, element, /*{}*/ expected) {
    var now_GR = new GlideRecord(table);
    var el = now_GR.getElement(element);
    for (var n in expected) {
        assertEquals(comment + "::" + table + "." + element, expected[n], el.sys_meta[n]);
    }
}
```

Scoped equivalent

To use the `getElement()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - getElement(String fieldName)**.

**GlideRecord - getEncodedQuery()**

Retrieves the query condition of the current result set as an encoded query string.

For details, see **Encoded query strings**.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to obtain the current encoded query to use later to create a read ACL role.
function createAcl(table, role) {
    gs.print("Checking security on table "+table);
    var now_GR = new GlideRecord("sys_security_acl");
    now_GR.addQuery("name", table);
    now_GR.addQuery("operation", "read");
    now_GR.query();
    var encQuery = now_GR.getEncodedQuery();
    if (now_GR.next()) {
        // existing acl found so use it
        createAclRole(now_GR.sys_id.toString(), role);
        return;
    } else {
        now_GR.initialize();
        now_GR.applyEncodedQuery(encQuery);
        var acl = now_GR.insert();
        gs.print("   Added read access control on " + table);
        createAclRole(acl, role);
    }
}

Scoped equivalent

To use the getEncodedQuery() method in a scoped application, use the corresponding scoped method: Scoped GlideRecord - getEncodedQuery().

GlideRecord - getEscapedDisplayValue()

Retrieves the field value for the display field of the current record and adds escape characters for use in Jelly scripts.

⚠️ Note: For this method to work, a display value must have been set on the associated table. For information on how to do this, see Select a field as the table display value.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Escaped value of the current table's display field.</td>
</tr>
</tbody>
</table>

This example shows how to obtain the user_name escaped display value from the User [sys_user] table.

```javascript
var userName = "Michael J'ones";
var now_GR = new GlideRecord("sys_user");
if (now_GR.get("user_name", userName)) {
    gs.print("Deleting existing user=" + userName);
    now_GR.deleteRecord();
}
now_GR.initialize();

var name = userName.split(" ", 2);
var firstName = name[0];
var lastName = name[1];
now_GR.setValue("user_name", userName);
gs.print("Name: " + now_GR.getValue("user_name"));
now_GR.setValue("first_name", firstName);
now_GR.setValue("last_name", lastName);
var escapeValue = now_GR.getEscapedDisplayValue();
gs.print("Escaped name: " + escapeValue);
```

Output:

- Name: Michael J'ones
- Escaped name: Michael J\'ones

**GlideRecord - getFields()**

Retrieves a Java ArrayList of fields in the current record.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java ArrayList</td>
<td>Fields in the current record</td>
</tr>
</tbody>
</table>

// This can be run in "Scripts - Background" for demonstration purposes

// Get a single incident record
var grINC = new GlideRecord('incident');
grINC.query();
grINC.next();
gs.print('Using ' + grINC.getValue('number'));
gs.print('');

// getFields() returns a Java ArrayList
var fields = grINC.getFields();

// Enumerate GlideElements in the GlideRecord object that have values
gs.print('Enumerating over all fields with values:');
for (var i = 0; i < fields.size(); i++) {
    var glideElement = fields.get(i);
    if (glideElement.hasValue()) {
        gs.print(' ' + glideElement.getName() + '\t' + glideElement);
    }
}
gs.print('');

// Get a specific GlideElement: number
gs.print('Getting the number field:');
for (var i = 0; i < fields.size(); i++) {
    var glideElement = fields.get(i);
    if (glideElement.hasValue() && glideElement.getName() == 'number') {
        gs.print(' ' + glideElement.getName() + '\t' + glideElement);
    }
}

GlideRecord - getLabel()
Retrieves the field's label.
This example shows how to retrieve a label of a question field in the Requested Item table and print the label and its display name.

```javascript
template.print("Summary of Requested items:\n");
var now_GR = new GlideRecord("sc_req_item");
now_GR.addQuery("request", current.sysapproval);
now_GR.query();
while(now_GR.next()) {
    var nicePrice = now_GR.price.toString();
    if (nicePrice != '') {
        nicePrice = parseFloat(nicePrice);
        nicePrice = nicePrice.toFixed(2);
    }
    template.print(now_GR.number + ":  " + now_GR.quantity + " X " +
    now_GR.cat_item.getDisplayValue()
    + " at $" + nicePrice + " each \n");
    template.print("    Options:\n");
    for (key in now_GR.variables) {
        var now_V = now_GR.variables[key];
        if(now_V.getGlideObject().getQuestion().getLabel() != '') {
            template.space(4);
            template.print('     ' + now_V.getGlideObject().getQuestion().getLabel() + " = " +
            now_V.getDisplayValue() + "\n");
        }
    }
}
```

**Scoped equivalent**

To use the `getLabel()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - getLabel()`.
GlideRecord - getLink(Boolean noStack)

Retrieves the link for the current record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>noStack</td>
<td>Boolean</td>
<td>Flag that indicates whether to append the generated link to the end of the URL. For example: &amp;sysparm_stack=[tablename]_list.do?sysparm_query=active=true. Valid values: • true: Append link. • false: Do not append link. Default: false</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>URL of the current record.</td>
</tr>
</tbody>
</table>

This example checks for attachments and adds the link if any are found.

```javascript
var attachment_link = ''; 
var rec = new GlideRecord('sc_req_item'); 
rec.addQuery('sys_id', current.request_item); 
rec.query(); 
if(rec.next()){ 
  if(rec.hasAttachments()){ 
    attachment_link = gs.getProperty('glide.servlet.uri') + rec.getLink(); 
  } 
}
```

Scoped equivalent

To use the getLink() method in a scoped application, use the corresponding scoped method: Scoped GlideRecord - getLink(Boolean noStack).

GlideRecord - getLocation()

Retrieves the row number of the current record within the table.
This example shows how to check if a record within the Incident table is not retrieved properly by comparing the record count to the record location number.

```javascript
var now_GR = new GlideRecord("incident");
now_GR.addQuery("priority", "4");
now_GR.query();
for (x=0; x < 4; x++) {
    now_GR.next();
    gs.print("Location value: " + now_GR.getLocation());
}
```

Output:

```
Location value: 0
Location value: 1
Location value: 2
Location value: 3
```

**GlideRecord - getPlural()**

Retrieves the plural label of the GlideRecord table.

For example, if the table name is "Change Request," this method returns "Change Requests."
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The plural label of the GlideRecord's table.</td>
</tr>
</tbody>
</table>

This example shows checks the return values for `getLabel()` and `getPlural()` for the Incident table.

```javascript
var now_GR = new GlideRecord('incident');
gs.print(now_GR.getLabel());
if (now_GR.getLabel() != 'Incident')
    throw "GlideRecord.getLabel() returned unexpected value for table 'incident': " +
    now_GR.getLabel();

gs.print(now_GR.getPlural());
if (now_GR.getPlural() != 'Incidents')
    throw "GlideRecord.getPlural() returned unexpected value for table 'incident': " +
    now_GR.getPlural();
"success";
```

### Scoped equivalent

To use the `getPlural()` method in a scoped application, use the scoped `GlideRecord.getED()` method to obtain the field’s descriptor, and then use the scoped `GlideElementDescriptor.getPlural()` method.

#### GlideRecord - getRecordClassName()

Retrieves the class (table) name for the current record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Class or table name</td>
</tr>
</tbody>
</table>

```javascript
function TaskAssignmentFilter() {
    var classname = current.getRecordClassName();
}
var filter = "type=null";
if (classname == "incident" && current.category == "database") {
    filter = GetGroupFilter("database");
}
else {
    // append exclusion for 'catalog' to the filter
    var cat = new GlideRecord("sys_user_group_type");
cat.addQuery("name", "catalog");
cat.query();
if (cat.next()) {
    filter += "^ORtype!=" + cat.sys_id;
}
}
gs.log("TaskAssignmentFilter: " + filter);
return filter;

Scoped equivalent

To use the getRecordClassName() method in a scoped application, use the corresponding scoped method: Scoped GlideRecord - getRecordClassName().

GlideRecord - getRelatedLists()
Retrieves a list of names and display values of related lists associated with the current GlideRecord.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HashMap</td>
<td>Hash map with names and display values of related lists.</td>
</tr>
</tbody>
</table>

This example shows how to obtain the list of names and display values of related lists associated with the current GlideRecord, store them in an array, and then print their values.

```javascript
var now_GR = new GlideRecord('incident');
var c = now_GR.getRelatedLists().values().toArray();
```
var numElements = c.length;
for( var i = 0; i < numElements; ++i){
gs.print(i+": "+c[i]);
}

GlideRecord - getRelatedTables()
Retrieves a list of names and display values of tables that are referred to by the
current record.
Parameters
Name

Type

Description

None
Returns
Type

Description

HashMap

Hash map with names and display values of related tables.

This example shows how to obtain the list of names and display values of tables
related to the current GlideRecord, store them in an array, and then print their
values.
var now_GR = new GlideRecord('incident');
var c = now_GR.getRelatedTables().values().toArray();
var numElements = c.length;
for( var i = 0; i < numElements; ++i){
gs.print(i+": "+c[i]);
}

GlideRecord - getRowCount()
Retrieves the number of rows (records) in the current GlideRecord object.
Note: This method should not be used in a production environment as it
creates a heavy load on the system. Instead, use GlideAggregate - Global.
Parameters
Name

Type

Description

None
2495

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


This example shows how to check if there are any records in the Incident table:

```javascript
function onSubmit() {
    var g_list = GlideRecord("incident");
    if (g_list.getRowCount() == 0) {
        alert("You must add at least one incident");
        return false;
    } else
    return true;
}
```

### Scoped equivalent
To use the `getRowCount()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - `getRowCount()`**.

### GlideRecord - `getRowNumber()`
Retrieves the row number set by `saveLocation()` or `setLocation()`.

To get the current row number, use `getLocation()`.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Saved row number.</td>
</tr>
</tbody>
</table>

This example shows how to save a row location in the Users table and then obtain that row number:

```javascript
var now_GR = new GlideRecord('sys_user');
now_GR.addQuery();
now_GR.query();
```
now_GR.saveLocation(5);
var savedRow = now_GR.getRowNumber();
gs.print("Saved row: " + savedRow);

Output:
Saved row: 5

**GlideRecord - getTableName()**
Retrieves the table name associated with this GlideRecord.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A table name</td>
</tr>
</tbody>
</table>

This example shows how to write the current table name in the system log.

```javascript
gs.log('Table: ' + current.getTableName());
gs.log('Parent: ' + current.parent.sys_id);
var item = new GlideRecord('sc_req_item');
item.addQuery('sys_id', current.parent.sys_id);
item.query();
if(item.next()){
    for(var variable in item.variable_pool) {
        gs.log(variable);
        var answer = eval("item.variable_pool." + variable + ".getDisplayValue()");
        gs.log(answer);
    }
}
```

**Scoped equivalent**

To use the `getTableName()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - getTableName()**.
GlideRecord - getValue(String fieldName)
Retrieves the string value of a specified field.

Note: If the Column Level Encryption Enterprise plugin is enabled and the instance has access to the key, this method returns clear text values from encrypted fields. If the instance does not have access to the key, this method returns the encrypted value. For more information, see Column Level Encryption Enterprise.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of a field for which to retrieve its value.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the specified element. Returns null if the field is empty or the field does not exist.</td>
</tr>
</tbody>
</table>

Note: Boolean values return as "0" and "1" string values instead of false and true.

This example uses the getValue() method to obtain field values in two GlideRecords that are being compared.

```javascript
function() {
  var jobExists = true;
  var jobGr;
  while(jobExists){
    jobGr = new GlideRecord("sys_trigger");
    jobExists = jobGr.get("name", "BaselineAPI: Create Baseline Job");
  }
  var baselineGr = new GlideRecord('pm_project_baseline');
  baselineGr.get('baseline_name', 'Test Project Baseline');
  baselineGr.query();
  Assert.assertEquals(true, baselineGr.next());
  var projectGr = new GlideRecord('pm_project');
  projectGr.get(this._project);
  Assert.assertEquals(projectGr.getValue('cost'), baselineGr.getValue('cost'));
  Assert.assertEquals(projectGr.getValue('capex_cost'), baselineGr.getValue('capex_cost'));
```
Scoped equivalent

To use the `getValue()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - getValue(String name)`.

**GlideRecord - GlideRecord(String tableName)**

Creates an instance of the GlideRecord class for the specified table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Table to use.</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('incident');
```

**GlideRecord - hasAttachments()**

Determines if the current GlideRecord has any attachments.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the current record has attachments, false otherwise.</td>
</tr>
</tbody>
</table>

This example checks for attachments and creates a link if there are any.

```javascript
var attachment_link = '';
var rec = new GlideRecord('sc_req_item');
```
rec.addQuery('sys_id', current.request_item);
rec.query();
if(rec.next()){
    if(rec.hasAttachments()){
        attachment_link = gs.getProperty('glide.servlet.uri') + rec.getLink();
    }
}

**GlideRecord - hasNext()**

Determines if there are any more records in the GlideRecord.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if there are more records in the query set, otherwise false.</td>
</tr>
</tbody>
</table>

This example shows how to call different functions based on whether there are additional records in the current GlideRecord.

```java
if (now_GR.hasNext()) {
    dothis(); // found it, do it
} else {
    dothat(); // didn't find it
}
```

**Scoped equivalent**

To use the `hasNext()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - hasNext()**.

**GlideRecord - initialize()**

Creates an empty record within the current GlideRecord that is suitable for population before an insert.
This example initializes a new record in the to_do table, creates a record, and then inserts it into the table.

```javascript
var now_GR = new GlideRecord('to_do');
now_GR.initialize();
now_GR.name = 'first to do item';
now_GR.description = 'learn about GlideRecord';
now_GR.insert();
```

**Scoped equivalent**

To use the `initialize()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - initialize()`.

**GlideRecord - insert()**

Inserts a new record with the field values that have been set for the current record.

This example creates a new record in the to_do table, sets values for some record fields, and then inserts the record into the table.
var now_GR = new GlideRecord('to_do');
now_GR.initialize();
now_GR.name = 'first to do item';
now_GR.description = 'learn about GlideRecord';
now_GR.insert();

**Scoped equivalent**

To use the `insert()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - insert()**.

**GlideRecord - insertWithReferences()**

Inserts a new record and also inserts or updates any related records with the provided information.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

If a reference value is not specified (as below), then a new user record is created with the provided first_name and last_name, and the caller_id value is set to this newly created sys_user record. The result is a new sys_user record with the provided first_name and last_name and a new incident record with the provided short_description and caller_id.

```javascript
var inc = new GlideRecord('incident');
inc.initialize();
inc.short_description = 'New incident 1';
inc.caller_id.first_name = 'John';
inc.caller_id.last_name = 'Doe';
inc.insertWithReferences();
```

If a caller_id value is specified, then that caller_id is updated with the provided first_name and last_name. The result is a newly created incident record with values set for short_description and caller_id.
```javascript
var inc = new GlideRecord('incident');
inc.initialize();
inc.short_description = 'New incident 1';
inc.caller_id.setDisplayValue('David Loo');
inc.caller_id.first_name = 'John';
inc.caller_id.last_name = 'Doe';
inc.insertWithReferences();
```

### GlideRecord - instanceOf(String className)
Checks a table for the type/class of table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>className</td>
<td>String</td>
<td>Name of a type or class of record.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if table is an instance of the specified class.</td>
</tr>
</tbody>
</table>

This example checks if the passed in group/user is of the sys_user table/class type.

```javascript
function(groupOrUser, label, grantedBy) {
  var labelCheck;
  var labelRecord;
  if (typeof label === "string") {
    labelRecord = new GlideRecord('label');
    labelRecord.addQuery('name', label);
    labelRecord.query();
    labelRecord.next();
  } else {
    labelRecord = label;
  }
  if (groupOrUser.instanceOf('sys_user')) {
    labelCheck = new GlideRecord('label_user_m2m');
    labelCheck.addQuery('user', groupOrUser.sys_id);
  } else {
    labelCheck = new GlideRecord('label_group_m2m');
    labelCheck.addQuery('group', groupOrUser.sys_id);
  }
}
```
```javascript
} 
labelCheck.addQuery('label', labelRecord.sys_id);
if (grantedBy !== "undefined") {
    labelCheck.addQuery('granted_by', grantedBy.sys_id);
}
labelCheck.query();
return labelCheck.hasNext();

GlideRecord - isNewRecord()
Determines whether the current record has been inserted into the database.

This method returns true only if the `newRecord()` method has been called. This method is useful for scripted ACL, and in the condition of UI actions, but should not be used in background scripts.

⚠️ **Note:** This method returns true for any new record during a business rule, or if the `newRecord()` method is used to initialize a record with default values and a unique ID (sys_id). In all other cases, it returns false.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates whether the current record is new (has not been inserted into the database.) Possible values:  
- true: Record is new.  
- false: Record is not new. |

This example checks if the new record has been inserted into the sys_user table.

```javascript
var now_GR = new GlideRecord("sys_user");
now_GR.newRecord();
now_GR.setValue("user_name", "John Smith");
gs.print("Is this a new record: "+ now_GR.isNewRecord());

var now_GR2 = new GlideRecord("sys_user");
```
GlideRecord - isValid()

Determine if the current GlideRecord table exists.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the table exists (valid), otherwise false.</td>
</tr>
</tbody>
</table>

This example checks if the Incident table exists.

```java
var testTable = new GlideRecord('incident');
gs.print(testTable.isValid());
```

Scoped equivalent

To use the isValid() method in a scoped application, use the corresponding scoped method: Scoped GlideRecord - isValid().

GlideRecord - isValidField(String fieldName)

Determine if the specified field is defined in the current GlideRecord table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of a field to check.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the field is defined in the table, otherwise false.</td>
</tr>
</tbody>
</table>

This example checks if the `sys_class_name` field exists in the `cmds_software_instance` table.

```javascript
var gobj = new GlideRecord('sys_db_object');
gobj.addQuery('name','cmdb_software_instance');
gobj.query();
if (!gobj.next()) {
  var si = new GlideRecord('cmdb_software_instance');
  si.query();
  var count = 0;
  while (si.next()) {
    if (!si.isValidField('sys_class_name')) {
      var ci = new GlideRecord('cmdb_ci');
      ci.name = si.name;
      ci.setNewGuidValue(si.sys_id);
      ci.sys_class_name = 'cmdb_software_instance';
      ci.insert();
      count++;
    }
  }
}
```

Scoped equivalent

To use the `isValidField()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - isValidField(String columnName)**.

**GlideRecord - isValidRecord()**

Determines if the current record is valid.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag indicating whether the current record is valid.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Record is valid.</td>
</tr>
<tr>
<td></td>
<td>• false: Past the end of the record set.</td>
</tr>
</tbody>
</table>

This example checks if the record identifier by the passed in sys_id is a valid record in the idea table.

```javascript
function(sysId) {
    this.log('Inside acceptIdea sysId = ' + sysId);
    var now_GR = new GlideRecord('idea');
    now_GR.get(sysId);
    this.log('now_GR.isValidRecord() = ' + now_GR.isValidRecord());
    if(now_GR.isValidRecord()) {
        now_GR.setValue('state', 2);
        this.log('Updating gliderecord');
        this.log('gliderecord table name' + now_GR.getTableName());
        now_GR.update();
    }
}
```

**Scoped equivalent**

To use the `isValidRecord()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - isValidRecord()`.

**GlideRecord - newRecord()**

Creates a GlideRecord, sets the default values for the fields, and assigns a unique ID to the record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example checks if the new record has been inserted into the sys_user table.

```javascript
var now_GR = new GlideRecord("sys_user");
now_GR.newRecord();
now_GR.setValue("user_name", "John Smith");
gs.print("Is this a new record: " + now_GR.isNewRecord());

var now_GR2 = new GlideRecord("sys_user");
now_GR2.addQuery("user_name", "Abel Tutor");
now_GR2.query();
now_GR2.next();
gs.print("Is this a new record: " + now_GR2.isNewRecord());
```

**Scoped equivalent**

To use the `newRecord()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - newRecord()`.

**GlideRecord - next()**

Moves to the next record in the GlideRecord.

Use this method to iterate through the records returned by a GlideRecord query.

⚠️ **Note:** This method fails if there is a field in the table called "next". If that is the case, use the method `GlideRecord - _next()`.

⚠️ **Note:** The `if(myObj.next())` construct only processes the first record returned.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates if there is a &quot;next&quot; record in the GlideRecord. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Move to the next record was successful.</td>
</tr>
<tr>
<td></td>
<td>• false: No more records in the result set.</td>
</tr>
</tbody>
</table>

This example shows how to iterate through the Incident table using the `next()` method.

```javascript
var rec = new GlideRecord('incident');
rec.query();
while (rec.next()) {
    gs.print(rec.number + ' exists');
}
```

Scoped equivalent

To use the `next()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - next()`.

GlideRecord - `_next()`

Moves to the next record in the GlideRecord. Provides the same functionality as `next()`, intended to be used in cases where the GlideRecord has a column named `next`.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if there are more records in the query set.</td>
</tr>
</tbody>
</table>

```javascript
var rec = new GlideRecord('sys_template');
rec.query();
while (rec._next()) {
    ...
}
```
Scoped equivalent

To use the `_next()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - _next().**

GlideRecord - operation()

Determines if an operation is insert, update, or delete.

Knowing the operation enables using `current.operation()` to make a generic business rule which can handle each operation uniquely.

For information on using the global variable `current`, refer to **Global variables in business rules.**

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>The current operation. Possible values: • delete • insert • update</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to use this method in a business rule. The `operation()` method detects which operation triggered an event and performs a different action depending for update and insert operations.

```java
if(current.operation() == "update") {
    current.updates ++; }
if(current.operation() == "insert") {
    current.updates = 0; }
```

This example shows how to use the `operation()` method to pass the operation value into the `startFlow()` method.
function createNewTransferOrder() {
    var toGr = new GlideRecord("alm_transfer_order");
    toGr.initialize();
    // From Stockroom - Southern California Warehouse
    toGr.setValue('from_stockroom', fromStockroomSysId);
    // To Stockroom - San Diego North - Pickup/Dropoff
    toGr.setValue('to_stockroom', toStockroomSysId);
    var transferOrder = toGr.insert();

    if(toGr.get(transferOrder)) {
        // Start Transfer Order Workflow
        var wf = new Workflow();
        var context = wf.startFlow(toWorkflowSysId, toGr, toGr.operation(), {});
        createNewTransferOrderLine(transferOrder);
    }
}

Scoped equivalent

To use the operation() method in a scoped application, use the corresponding scoped method: Scoped GlideRecord - operation().

GlideRecord - orderBy(String fieldName)

Specifies a field name to use to order the query set. To order by multiple fields, call this method multiple times with different field values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Field on which to order the query set.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to order the query set by “order” and then by “number”.

function UpdateProjectWBS(project) {
    var count = 0;
    var child = new GlideRecord('pm_project_task');
    child.addQuery('parent', project.sys_id);
child.orderBy('order');
child.orderBy('number');
child.query();
var len = child.getRowCount().toString().length;
var seq = 0;
while (child.next()) {
    count += UpdateProjectTaskWBS(child, 1, ++seq, len, '');
}
gs.addInfoMessage(count + ' Project Tasks updated');

Scoped equivalent

To use the `orderBy()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - orderBy(String name)`.  

**GlideRecord - orderByDesc(String, fieldName)**

Specifies the field to use to order the query set in descending order.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of field to use to order the query results in descending order.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to order the query set in ascending or descending order.

```javascript
function(fiscalType, orderByDesc) {
    var now_GR = new GlideRecord('fiscal_period');
    now_GR.addQuery('fiscal_type', fiscalType);
    if (orderByDesc) {
        now_GR.orderByDesc('fiscal_start_date_time');
    } else {
        now_GR.orderBy('fiscal_start_date_time');
    }
    now_GR.setLimit(1);
}
```
Scoped equivalent

To use the `orderByDesc()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - orderByDesc(String name)`.

GladeRecord - query(String field, String value)

Runs a query against the table based on the filters specified by query methods such as `addQuery()` and `addEncodedQuery()`.

This method queries the GlideRecord table as well as any references of the table. For more information, see Querying tables in script.

⚠️ Note: This method fails if there is a field in the table called "query". If that is the case, use the method `_query()`. To run queries in a domain-separated instance, use the method queryNoDomain().

See also:
- GlideAggregate - Global
- GlideQuery - Scoped, Global
- GlideQueryCondition - Global

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Optional - must also specify value parameter. Name of the field to search for the value specified in the value parameter.</td>
</tr>
</tbody>
</table>

⚠️ Note: This method is typically run without arguments, but you can specify a name-value pair to filter records containing the specified values. If the parameters are specified, the "name=value" condition is added to the query.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>Optional - must also specify field parameter. Value to search for in the specified field parameter.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to scan the System Dictionary [sys_dictionary] for tables with a query field.

```javascript
var tableArr = [];

var now_GR = new GlideRecord('sys_dictionary');
now_GR.addQuery('element', 'query');
now_GR.setLimit(6);

now_GR.query();

while(now_GR.next()){  
    tableArr.push(now_GR.name.getValue());
}

gs.info('The following tables have a field column called "query":');

for (i = 0; i < tableArr.length; i++) {
    gs.info(tableArr[i]);
}

Output:

The following tables have a field column called "query":
  cmdb_convert_bulk_services
  cmdb_multisource_query_status
  cmdb_qb_result_base
  cmdb_qb_table_mapping
  discovery_probes_cim_query
  kb_feedback
```
Scoped equivalent

To use the `query()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - query(String field, String value)`.

GlideRecord - `_query(String field, String value)`

Runs a query against the table based on the filters specified by query methods such as `addQuery()` and `addEncodedQuery()`. This method is intended to be used on tables in which there is a column named "query", which might cause errors running the `query()` method.

This method queries the GlideRecord table as well as any references of the table. For more information, see Querying tables in script.

See also:
- GlideAggregate - Global
- GlideQuery - Scoped, Global
- GlideQueryCondition - Global

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Optional - must also specify value parameter. Name of the field to search for the value specified in the value parameter.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Optional - must also specify field parameter. Value to search for in the specified field parameter.</td>
</tr>
</tbody>
</table>

Note: This method is typically run without arguments, but you can specify a name-value pair to filter records containing the specified values. If the parameters are specified, the "name=value" condition is added to the query.

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
The following example shows how query the Knowledge Feedback [kb_feedback] table and list KB articles with comments that include Excel.

```javascript
var rec = new GlideRecord('kb_feedback');
rec.addQuery('comments', 'CONTAINS', 'Excel');
rec._query();
while (rec.next()) {
    gs.info(rec.getDisplayValue('article') + " comment: " + rec.getValue('comments'));
}
```

Output:

```
KB0000005 comment:
    Can you please add the version of Excel this applies to? All?

KB0000005 comment:
    Does this work for all Excel versions? OSX and Windows alike?
```

**Scoped equivalent**

To use the `_query()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - _query(String field, String value)**.

**GlideRecord - queryNoDomain(String field, String value)**

Used in domain-separated instances. Similar to `query()`, runs the query against the table based on the filters specified by `addQuery()` and `addEncodedQuery()`, but ignores domains.

This method queries the GlideRecord table as well as any references of the table. For more information, see [Querying tables in script](#).

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Optional - must also specify value parameter. Name of the field to search for the value specified in the value parameter. <strong>Note:</strong> This method is typically run without arguments, but you can specify a name-value pair to filter records containing the specified values. If the parameters are specified, the &quot;name=value&quot; condition is added to the query.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>Optional - must also specify field parameter. Value to search for in the specified field parameter.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to query the incident table for all domains.

```javascript
var rec = new GlideRecord('incident');
rec.queryNoDomain();
while (rec.next()) {
  gs.print(rec.number + ' exists');
}
```

**GlideRecord - restoreLocation()**

Sets the current record to be the record that was saved with `saveLocation()`. If `saveLocation()` has not been called, the current record is set to the first record of the GlideRecord.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how save the current record location, move to the next record, and then return back to the saved location.

```javascript
function() {
  var now_GR = new GlideRecord("incident");
```
now_GR.query();
now_GR.next();
var firstID = now_GR.getValue("sys_id");
now_GR.next();
var restoreID = now_GR.getValue("sys_id");
now_GR.saveLocation();
now_GR.next();
var lastID = now_GR.getValue("sys_id");
now_GR.restoreLocation();
now_GR.next();
var thisID = now_GR.getValue("sys_id");
var progress = "first id=" + firstID
+ "\n\t last id=" + lastID
+ "\n\t saved id=" + restoreID
+ "\n\t restored id=" + thisID;
gs.print("Progress:\n\t" + progress);
}

**GlideRecord - saveLocation()**

Saves the current row number so you can return to this location using the `restoreLocation()` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how save the current record location, move to the next record, and then return back to the saved location.

```javascript
function() {
  var now_GR = new GlideRecord("incident");
  now_GR.query();
  now_GR.next();
  var firstID = now_GR.getValue("sys_id");
  now_GR.next();
```
var restoreID = now_GR.getValue("sys_id");
now_GR.saveLocation();
now_GR.next();
var lastID = now_GR.getValue("sys_id");
now_GR.restoreLocation();
now_GR.next();
var thisID = now_GR.getValue("sys_id");
var progress = "first id=" + firstID
 + "\n\t last id=" + lastID
 + "\n\t saved id=" + restoreID
 + "\n\t restored id=" + thisID;
gs.print("Progress:\\n" + progress);
}

GlideRecord - setAbortAction(Boolean b)

Sets a flag to indicate if the next database action (insert, update, delete) is to
be aborted.

Use in an onBefore business rule to prevent the database action from being
done. The business rule continues to run after setAbortAction() is called. Calling
setAbortAction() does not stop subsequent business rules from executing. Calling
this method only prevents the database action from occurring.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>True to abort next action, or false to allow the next action.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

if (!(current.u_date1.nil()) && (!current.u_date2.nil())) {
  var start = current.u_date1.getGlideObject().getNumericValue();
  var end = current.u_date2.getGlideObject().getNumericValue();
  if (start > end) {
    gs.addInfoMessage('start must be before end');
    current.u_date1.setError('start must be before end');
    current.setAbortAction(true);
  }
Scoped equivalent

To use the `setAbortAction()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - setAbortAction(Boolean b)**.

**GlideRecord - setDisplayValue(String name, Object value)**

Sets the specified field to the specified display value.

For a reference field this is the display value for the table. For a date/time this is the time in the caller's current timezone.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field for which to set the display value.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>Display value for the specified field.</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('incident');
now_GR.get('46f09e75a9fe198100f4f6d366d17b');
now_GR.setDisplayValue('opened_at','2011-02-13 4:30:00');
now_GR.update();
```

**GlideRecord - setForceUpdate(Boolean force)**

Updates the record even if fields have not changed.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>force</td>
<td>Boolean</td>
<td>True to update even if fields have not changed, otherwise false.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to force a record in the itfm_cost_model table to update even without changes.

```javascript
function() {
    gs.log("Update Script: Upgrade_cost_model_data.js started");
    var now_GR = new GlideRecord("itfm_cost_model");
    now_GR.initialize();
    now_GR.addNullQuery("source_table");
    now_GR.query();
    while (now_GR.next()) {
        now_GR.setForceUpdate(true);
        var sourceTable = getSourceTable(now_GR.getValue('clone_from'));
        if (sourceTable) {
            now_GR.setValue("source_table", sourceTable);
            now_GR.setWorkflow(false);
            now_GR.update();
        }
    }
}
```

GlideRecord - setLimit(Number limit)

Sets the maximum number of records to return in the GlideRecord from a query.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>limit</td>
<td>Number</td>
<td>Limit for records to fetch.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example sets the number of records returned in the query to a maximum of ten.
```javascript
var now_GR = new GlideRecord('incident');
now_GR.orderByDesc('sys_created_on');
now_GR.setLimit(10);
now_GR.query();
```

**Scoped equivalent**

To use the `setLimit()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - setLimit(Number maxNumRecords)`.

**GlideRecord - setLocation(Number rowNumber)**

Sets the current row location to the specified value.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rowNumber</td>
<td>Number</td>
<td>Row number to set as the current row.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to set the record position in the Incident table

```javascript
verifySetLocation : function() {
    var now_GR = new GlideRecord("incident");
    now_GR.query();
    var recordIDs = new Array();
    while(now_GR.next()) {
        recordIDs.push(now_GR.getValue("sys_id"));
    }
    for (var ix = recordIDs.length - 1; ix >= 0; ix--) {
        now_GR.setLocation(ix);
        Assert.assertEquals(recordIDs[ix], now_GR.getValue("sys_id"),
            "The entry retrieved for the index "+ ix + " is not valid";
    }
}
```
**GlideRecord - setNewGuid()**

Generates a new GUID and sets it as the unique ID (sys_id) for the current record.

This function applies only to new records. You cannot change the GUID for an existing record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the current record.</td>
</tr>
</tbody>
</table>

This example creates a new task record and stores the ID in the task_rel_task table.

```javascript
var task = new GlideRecord ('task');
var tsk_id = task.setNewGuid();

task.description = "Request: " + current.request.number;
task.description = task.description + "\n" + "Requested by: " +
current.request.u_requested_by.name;
task.description = task.description + "\n" + "Requested for: " +
current.request.u_requested_for.name;
task.description = task.description + "\n" + "Item: " + current.cat_item.name;

var now_GR = new GlideRecord ('task_rel_task');
//link the incident to the request
now_GR.parent = current.request;
now_GR.child = tsk_id;
now_GR.insert();
```

**GlideRecord - setNewGuidValue (String guid)**

Generates a new GUID and sets it as the unique ID for the current record, when inserting a new record.
This example generates a new GUID and sets it as the unique ID for the newly created `cmdb_software_instance` record.

```javascript
var gobj = new GlideRecord('sys_db_object');
gobj.addQuery('name','cmdb_software_instance');
gobj.query();
if (!gobj.next()) {
  var si = new GlideRecord('cmdb_software_instance');
  si.query();
  var count = 0;
  while (si.next()) {
    if (!si.isValidField('sys_class_name')) {
      var ci = new GlideRecord('cmdb_ci');
      ci.name = si.name;
      ci.setNewGuidValue(si.sys_id);
      ci.sys_class_name = 'cmdb_software_instance';
      ci.insert();
      count++;
    }
  }
}
```

**Scoped equivalent**

To use the `setNewGuidValue()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - setNewGuidValue(String guid)`.

**GlideRecord - setQueryReferences(Boolean queryReferences)**

Enables or disables using the reference field's display name when querying a reference field.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queryReferences</td>
<td>Boolean</td>
<td>Flag indicating what time of data to generate. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Generate a string of display names.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Generate a string of sys_ids.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var grByName = new GlideRecord(this.TEST_TABLE);
grByName.addQuery("caller_id", userDisplayName);
grByName.setQueryReferences(true);
grByName.query();
if (grByName.hasNext()) {
    while(grByName.next()) {
        gs.print("Incident caller_id=" + grByName.caller_id);
    }
} else {
    gs.print("NO RESULTS");
}
```

GlideRecord - setUseEngines(Boolean e)

Disables or enables the running of any engines (approval rules / assignment rules).

⚠️ **Warning:** Disabling the running of engines can have a significant impact on your ServiceNow® instance and how it operates. Ensure that you thoroughly test this change before deploying it to production.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>e</td>
<td>Boolean</td>
<td>Flag that indicates whether to enable or disable the running of any engines.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Disable engines.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Enable engines.</td>
</tr>
</tbody>
</table>
This example shows how to disable engines and business rules before executing a query.

```javascript
function auditOAuthInboundRequestsUsage()
{
    var now_GR = new GlideRecord("oauth_credential");
    var oauthClient = now_GR.addJoinQuery("oauth_entity", "peer", "sys_id");
    now_GR.addQuery("type", "access_token");
    now_GR.addQuery("expires", ">", nowDateTime());
    now_GR.addNullQuery("oauth_requestor_profile");
    oauthClient.addCondition("active", "true");
    oauthClient.addCondition("type", "client");
    now_GR.setUseEngines(false);
    now_GR.setWorkflow(false);
    now_GR.query();
    return now_GR.hasNext();
}
```

**GlideRecord - setValue(String name, Object value)**

Sets the specified field to the specified value.

Normally a script would do a direct assignment, for example, `now_GR.category = value`. However, if in a script the element name is a variable, then you can use `now_GR.setValue(elementName, value)`. When setting a value, ensure the data type of the field matches the data type of the value you enter.

If the value parameter is null, the record is not updated, and an error is not thrown.

**Note:** This method cannot be used on journal fields, use `GlideElement.setJournalEntry()` instead.
Note: If the Column Level Encryption Enterprise plugin is enabled and the instance has access to the key, this method can insert encrypted data into encrypted fields. If the instance does not have access to the key, this method returns an error. For more information, see Column Level Encryption Enterprise.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field whose value you want to set.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>Value to set in the specified field.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to set the value of the state field in the planned_task table.

function() {
    var topTask = new GlideRecord('planned_task');
    topTask.setValue('state', -5);
    var childTask = new GlideRecord('planned_task');
    childTask.setValue('state', -5);
    var s = new PlannedTaskStateManagement(childTask, topTask);
    var state = s/manageStateChange(false);
    Assert.assertEquals("-5", state);
}

Scoped equivalent

To use the setValue() method in a scoped application, use the corresponding scoped method: Scoped GlideRecord - setValue(String name, Object value).

GlideRecord - setWorkflow(Boolean e)

Enables or disables the running of business rules that might normally be triggered by subsequent actions. If the e parameter is set to false, an insert/update will not be audited. Auditing only happens when the parameter is set to true for a GlideRecord operation.
**Note:** The `setWorkflow()` method is ignored when subsequently using either the `deleteProblem()` or `deleteMultiple()` methods to cascade delete.

**Warning:** Disabling the running of business rules can have a significant impact on your ServiceNow® instance and how it operates. Ensure that you thoroughly test this change before deploying it to production.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| e    | Boolean | Flag that indicates whether to enable or disable the running of business rules. Valid values:  
|      |         | • true: Enable business rules  
|      |         | • false: Disable business rules |

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
doit('name1','name2');

function doit(username1,username2) {

    var usr1 = new GlideRecord('sys_user');
    var usr2 = new GlideRecord('sys_user');
    var num = 0;

    if (usr1.get('user_name',username1) && usr2.get('user_name',username2)) {
        var ref;
        var dict = new GlideRecord('sys_dictionary');
        dict.addQuery('reference','sys_user');
        dict.addQuery('internal_type','reference');
        dict.query();
        while (dict.next()) {
            num = 0;
            ref = new GlideRecord(dict.name.toString());
            ref.addQuery(dict.element,usr1.sys_id);
            ref.query();
            while (ref.next()) {
```
ref.setValue(dict.element.toString(),usr2.sys_id);
ref.setWorkflow(false);
ref.update();
num++;
}
if (num > 0) {
    gs.print(dict.element + ' changed from ' + usr1.user_name +
    ' to ' + usr2.user_name + ' in ' + num + ' ' + dict.name + ' records');
}


Scoped equivalent
To use the setWorkflow() method in a scoped application, use the corresponding scoped method: Scoped GlideRecord - setWorkflow(Boolean enable).

GlideRecord - update(Object reason)
Updates the GlideRecord with any changes that have been made. If the record does not exist, it is inserted.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reason</td>
<td>Object</td>
<td>Optional. Reason for the update. The reason appears in the audit record.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_id of the new or update record. Returns null if the update fails.</td>
</tr>
</tbody>
</table>

This example updates a record in the task_ci table.

```javascript
var now_GR = new GlideRecord('task_ci');
now_GR.addQuery();
now_GR.query();
var count = now_GR.getRowCount();
if (count > 0) {
    var allocation = parseInt(10000 / count) / 100;
```
while (now_GR.next()) {
    now_GR.u_allocation = allocation;
    now_GR.update();
}

Scoped equivalent

To use the `update()` method in a scoped application, use the corresponding scoped method: `Scoped GlideRecord - update(String reason)`.

### GlideRecord - updateMultiple()

Updates each GlideRecord in a stated query with a specified set of changes.

For information on adding journal entries, refer to the `setJournalEntry()` method.

⚠️ **Note:** To ensure expected results, use the `setValue()` method instead of direct assignments. That is, use `gr.setValue('<field_name>', '4')`; instead of `gr.<field_name> = 4`.

⚠️ **Note:** Do not use this method with the `chooseWindow()` or `setLimit()` methods when working with large tables.

This method sets new values and does not clear existing values. To clear an existing value, use the `setValue()` method and set the field to `null`.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to update the state of all active incidents to 4 - "Awaiting User Info".

```javascript
var now_GR = new GlideRecord('incident');
now_GR.addQuery('active', true);
now_GR.setValue('state', 4);
now_GR.updateMultiple();
```
Scoped equivalent

To use the `updateMultiple()` method in a scoped application, use the corresponding scoped method: **Scoped GlideRecord - updateMultiple().**

GlideRecord - `updateWithReferences(Object reason)`

Updates a record and also inserts or updates any related records with the information provided.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reason</td>
<td>Object</td>
<td>Reason for the updates. The reason is displayed in the audit record.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_id for the record being updated.</td>
</tr>
</tbody>
</table>

If processing an incident where the Caller ID is set to reference sys_user record 'John Doe,' then the following code would update John Doe's user record. If processing an incident where there is no Caller ID specified, then the following code would create a new sys_user record with the provided information (first_name, last_name) and set the Caller ID value to the newly created sys_user record.

```javascript
var inc = new GlideRecord('incident');
inc.get(inc_sys_id);  // Looking up an existing incident record where 'inc_sys_id'
                      // represents the sys_id of a incident record
inc.caller_id.first_name = 'John';
inc.caller_id.last_name = 'Doe';
inc.updateWithReferences();
```

GlideRecord - Scoped

Scoped GlideRecord is used for database operations.

The GlideRecord API is the primary means of interfacing with the database on the server-side code. A GlideRecord is an object that contains records from a single table. Use the API to instantiate a GlideRecord object and add query parameters, filters, limits, and ordering.
For information on a class that performs the same functions as GlideRecord and enforces ACLs, see Using GlideRecordSecure.

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an insert(), update(), deleteRecord(), or deleteMultiple() method on bad query results can result in data loss.

You can set the glide.invalid_query.returns_no_rows system property to true to have queries with invalid encoded queries return no records.

Retrieve values from records

In most cases, do not use dot-walking to get values from a record. Dot-walking retrieves the entire object instead of the field value. Retrieving the object uses much more storage and might cause undesirable results when used in arrays or in Service Portal.

Instead of retrieving the entire object, you can use one of the following methods to copy the field values:

• getValue()
• getDisplayValue()

If dot-walking through a GlideElement object is necessary, use the toString() method to retrieve values. For example, you might need the current caller's manager sys_id to set another reference field. The following example shows how to get the string value instead of the entire object:

```javascript
var mgr = current.caller_id.manager.toString();
```

See also:

• GlideAggregate
• GlideElement
• GlideQuery

Scoped GlideRecord - addActiveQuery()

Adds a filter to return active records.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>Filter to return active records.</td>
</tr>
</tbody>
</table>

```javascript
var inc = new GlideRecord('incident');
inc.addActiveQuery();
inc.query();
```

Scoped GlideRecord - `addEncodedQuery(String query)`

Adds an encoded query to other queries that may have been set.

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an `insert()`, `update()`, `deleteRecord()`, or `deleteMultiple()` method on bad query results can result in data loss.

You can set the `glide.invalid_query.returns_no_rows` system property to true to have queries with invalid encoded queries return no records.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>String</td>
<td>An encoded query string.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var queryString = "priority=1^ORpriority=2";
var now_GR = new GlideRecord('incident');
```
Scoped GlideRecord - addFunction(Object function)
Applies a pre-defined GlideDBFunctionBuilder object to a record.

Use the GlideDBFunctionBuilder scoped class to define a function. After the
function is defined, use the addFunction(Object function) method to apply the
function to a record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>function</td>
<td>Object</td>
<td>GlideDBFunctionBuilder object that defines a SQL operation.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var functionBuilder = new GlideDBFunctionBuilder();
var myAddingFunction = functionBuilder.add();
myAddingFunction = functionBuilder.field('order');
myAddingFunction = functionBuilder.field('priority');
myAddingFunction = functionBuilder.build();

// Query incidents in which order + priority is less than 5 and display that added value
var now_GR = new GlideRecord('incident');
now_GR.addFunction(myAddingFunction);
now_GR.addQuery(myAddingFunction, '<', 5);
now_GR.query();
while(now_GR.next())
    gs.info(now_GR.getValue(myAddingFunction));
```

Output:

1
4
3
Scoped GlideRecord – addJoinQuery(String joinTable, String primaryField, String joinTableField)

Adds a filter to return records based on a relationship in a table related to the current GlideRecord.

You can use this method to find all the users that are in the database group via the Group Member [sys_user_grmember] table, or to find all problems that have an assigned incident via the incident.problem_id relationship.

This is not a true database join; rather, addJoinQuery() adds a subquery. So, while the result set is limited based on the join, the only fields that you have access to are those on the base table (those which are in the table with which the GlideRecord was initialized.)

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an insert(), update(), deleteRecord(), or deleteMultiple() method on bad query results can result in data loss.

You can set the glide.invalid_query.returns_no_rows system property to true to have queries with invalid encoded queries return no records.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>joinTable</td>
<td>String</td>
<td>Name of table to use in the join, such as 'incident'.</td>
</tr>
<tr>
<td>primaryField</td>
<td>String</td>
<td>Optional. Name of the field in the GlideRecord to use to join the field specified in the joinTableField parameter. Default: sys_id</td>
</tr>
<tr>
<td>joinTableField</td>
<td>String</td>
<td>Optional. Name of the field in the table specified in joinTable to use to join the tables. Default: First field in the table specified in joinTable that is a reference field to the current GlideRecord table.</td>
</tr>
</tbody>
</table>
## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>Filter that lists records where the relationships match.</td>
</tr>
</tbody>
</table>

Find problems that have an incident attached. This example returns problems that have associated incidents. However, it won't pull values from the incidents that are returned as a part of the query.

```javascript
var prob = new GlideRecord('problem');
prob.addJoinQuery('incident');
prob.query();
```

Find inactive problems with associated incidents.

```javascript
// Look for Problem records that have associated Incident records
var now_GR = new GlideRecord('problem');
var grSQ = now_GR.addJoinQuery('incident');

// Where the Problem records are "active=false"
now_GR.addQuery('active', 'false');

// And the Incident records are "active=true"
grSQ.addCondition('active', 'true');

// Query
now_GR.query();

// Iterate and output results
while (now_GR.next()) {
    gs.info(now_GR.getValue('number'));
}
```

Find problems that have incidents associated where the incident `caller_id` field value matches that of the problem `opened_by` field.

```javascript
var now_GR = new GlideRecord('problem');
    now_GR.addJoinQuery('incident', 'opened_by', 'caller_id');
now_GR.query();
```

## Scoped GlideRecord - addNotNullQuery(String fieldName)

A filter that specifies records where the value of the field passed in the parameter is not null.
Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an `insert()`, `update()`, `deleteRecord()`, or `deleteMultiple()` method on bad query results can result in data loss.

You can set the `glide.invalid_query.returns_no_rows` system property to true to have queries with invalid encoded queries return no records.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The name of the field to be checked.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>A filter that specifies records where the value of the field passed in the parameter is not null.</td>
</tr>
</tbody>
</table>

```javascript
var target = new GlideRecord('incident');
target.addNotNullQuery('short_description');
target.query(); // Issue the query to the database to get all records where short_description is not null
while (target.next()) {
    // add code here to process the incident record
}
```

### Scoped GlideRecord - addNullQuery(String fieldName)

Adds a filter to return records where the value of the specified field is null.

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an `insert()`, `update()`, `deleteRecord()`, or `deleteMultiple()` method on bad query results can result in data loss.
You can set the `glide.invalid_query.returns_no_rows` system property to true to have queries with invalid encoded queries return no records.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The name of the field to be checked.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>The query condition added to the GlideRecord.</td>
</tr>
</tbody>
</table>

```javascript
var target = new GlideRecord('incident');
target.addNullQuery('short_description');
target.query();  // Issue the query to the database to get all records where
                 // short_description is null
while (target.next()) {
    // add code here to process the incident record
}
```

**Scoped GlideRecord - addQuery(String query)**

Adds a filter to return records using an encoded query string.

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an `insert()`, `update()`, `deleteRecord()`, or `deleteMultiple()` method on bad query results can result in data loss.

You can set the `glide.invalid_query.returns_no_rows` system property to true to have queries with invalid encoded queries return no records.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>String</td>
<td>An encoded query string.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>The query condition added to the GlideRecord.</td>
</tr>
</tbody>
</table>

```javascript
var rec = new GlideRecord('incident');
rec.addQuery('active=true');
rec.query();
while (rec.next()) {
    rec.setValue('active', false);
    gs.info('Active incident ' + rec.getValue('number') + ' closed');
    rec.update();
}
```

**Scoped GlideRecord - addQuery(String name, Object value)**

Build a search query and return the rows that match the request.

If you are familiar with SQL, this method is similar to the "where" clause. One or more `addQuery()` calls can be made in a single query; in this case the queries are AND’ed. If any of the query statements need to be OR’ed, use the `GlideQueryCondition` method `addOrCondition()`.

When `addQuery()` is called with only two parameters, table name and comparison value, such as `myObj.addQuery('category','Hardware')`, the operator is assumed to be "equal to".

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an `insert()`, `update()`, `deleteRecord()`, or `deleteMultiple()` method on bad query results can result in data loss.

You can set the `glide.invalid_query.returns_no_rows` system property to true to have queries with invalid encoded queries return no records.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Table field name. If you are not querying a table field, use these reserved variables:</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>123TEXTQUERY321</td>
<td></td>
<td>Adds a search term to the query. Use this option to return matching values from any field in the table. Use the term you want to query as the value.</td>
</tr>
<tr>
<td>123TEXTINDEXGROUP321</td>
<td></td>
<td>Adds a text index group to the query. Use the name of the text index group from the Text Index Groups [ts_index_group] table you want to query as the value. For more information about text index groups, see Configure multiple tables for indexing and searching.</td>
</tr>
</tbody>
</table>

value Object Value on which to query (not case-sensitive).

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>Query condition added to the GlideRecord.</td>
</tr>
</tbody>
</table>

This code example shows how to query the active records in the incident table and then set them all to inactive.

```javascript
var rec = new GlideRecord('incident');
rec.addQuery('active', true);
rec.query();
while (rec.next()) {
    rec.setValue('active', false);
    rec.update();
}
```

Adds a query for records in the kb_knowledge table and portal text index group.

```javascript
var now_GR = new GlideRecord("kb_knowledge");
now_GR.addQuery("123TEXTQUERY321", "email server");
now_GR.addQuery("123TEXTINDEXGROUP321", "portal");
now_GR.query();
```

Scoped GlideRecord - addQuery(String name, String operator, Object value)

Provides the ability to build a request, which when executed, returns the rows from the specified table, that match the request.
If you are familiar with SQL, this method is similar to the "where" clause. One or more `addQuery()` calls can be made in a single query; in this case the queries are AND'ed. If any of the query statements need to be OR'ed, use the `GlideQueryCondition` method `addOrCondition()`.

Always test queries on a sub-production instance prior to deploying them on a production instance. An incorrectly constructed encoded query, such as including an invalid field name, produces an invalid query. When the invalid query is run, the invalid part of the query condition is dropped, and the results are based on the valid part of the query, which may return all records from the table. Using an `insert()`, `update()`, `deleteRecord()`, or `deleteMultiple()` method on bad query results can result in data loss.

You can set the `glide.invalid_query.returns_no_rows` system property to true to have queries with invalid encoded queries return no records.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Table field name.</td>
</tr>
<tr>
<td>operator</td>
<td>String</td>
<td>Query operator. The available values are dependent on the data type of the value parameter. Numbers:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• =</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• !=</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &gt;=</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &lt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• &lt;=</td>
</tr>
<tr>
<td>Strings (must be in upper case):</td>
<td></td>
<td>• =</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• !=</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NOT IN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• STARTSWITH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ENDSWITH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CONTAINS</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Object</td>
<td>Value on which to query (not case-sensitive).</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideQueryCondition</td>
<td>The query condition that was added to the GlideRecord.</td>
</tr>
</tbody>
</table>

```javascript
var rec = new GlideRecord('incident');
rec.addQuery('active', true);
rec.addQuery('sys_created_on', '>', '2010-01-19 04:05:00');
rec.query();
while (rec.next()) {
    rec.setValue('active', false);
    gs.info('Active incident ' + rec.getValue('number') + ' closed');
    rec.update();
}
```

Using the IN operator.

```javascript
var now_GR = new GlideRecord('incident');
now_GR.addQuery('number', 'IN', 'INC00001,INC00002');
now_GR.query();
while (now_GR.next()) {
    //do something....
}
```

**Scoped GlideRecord - addValue(String field, Number value)**

Provides atomic add and subtract operations on a specified number field at the database level for the current GlideRecord object.

Typically, a GlideRecord object is written as one record in a database. Individual field values are stored as defined. For code that adds a value to a GlideRecord field, it simply saves the field to the database with the new value, rather than atomically incrementing it.
For example, when the following code is executed, the value of the u_count field in the database is 2.

```java
gs.info(now_GR.u_count); // "1"
now_GR.u_count += 1;
now_GR.update();
now_GR.get(now_GR.sys_id);
gs.info(now_GR.u_count); // "2"
```

If another user concurrently runs identical code, instead of the two operations each adding 1 to u_count, the net effect is that u_count only contains 2, with one operation's update actually being lost. Conversely, the `addValue()` method performs the addition/subtraction in the database when the record is updated as an atomic operation. Two operations running concurrently each properly update the field.

```java
gs.info(now_GR.u_count); // "1"
now_GR.addValue("u_count", 1);
now_GR.update();
now_GR.get(now_GR.sys_id); // The record must be reloaded from the database to observe the result
gs.info(now_GR.u_count); // "3", if executed concurrently with another user
```

⚠️ **Note:** The new value is not read back from the database unless explicitly done so.

Like `setValue()`, `addValue()` changes only take effect in the database after a subsequent call to `update()` or `insert()`. If `insert()` is called, the specified field is initialized with the `value` parameter passed into `addValue()`.

⚠️ **Note:** If `setValue()` is called for the specified field prior to calling `addValue()`, the `addValue()` method is not processed and an error message is logged.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>The name of the field in this GlideRecord to modify. If the associated field is not a numeric type, the operation is ignored.</td>
</tr>
<tr>
<td>value</td>
<td>Number</td>
<td>The amount to add to the value when the record is saved. To perform a subtraction operation, simply pass a negative value.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Shows a value being added.

```java
gs.info(now_GR.u_count); // "1"
now_GR.addValue("u_count", 1);
now_GR.update();
now_GR.get(now_GR.sys_id); // The record must be reloaded from the database to observe the result
gs.info(now_GR.u_count);
```

Output: 2

Shows a value being subtracted.

```java
gs.info(now_GR.u_count); // "4"
now_GR.addValue("u_count", -1);
now_GR.update();
now_GR.get(now_GR.sys_id); // The record must be reloaded from the database to observe the result
gs.info(now_GR.u_count);
```

Output: 3

### Scoped GlideRecord - applyEncodedQuery(String queryString)

Sets the values of the specified encoded query terms and applies them to the current GlideRecord.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queryString</td>
<td>String</td>
<td>Encoded query to apply to the current GlideRecord.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
```javascript
function createAcl(table, role) {
    gs.info("Checking security on table "+table);
    var now_GR = new GlideRecord("sys_security_acl");
    now_GR.addQuery("name", table);
    now_GR.addQuery("operation", "read");
    now_GR.query();
    var encQuery = now_GR.getEncodedQuery();
    if (now_GR.next()) {
        // existing acl found so use it
        createAclRole(now_GR.getValue('sys_id'), role);
        return;
    } else {
        now_GR.initialize();
        now_GR.applyEncodedQuery(encQuery);
        var acl = now_GR.insert();
        gs.info("Added read access control on "+table);
        createAclRole(acl, role);
    }
}
```

**Scoped GlideRecord - canCreate()**

Determines if the Access Control Rules, which include the user's roles, permit inserting new records in this table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the user's roles permit creating of records in this table. Valid values: • true: Creating permitted • false: Creating is not permitted</td>
</tr>
</tbody>
</table>

This code example writes whether the current user can create records in the Incident table in the system log.
var now_GR = new GlideRecord('incident');
gs.info(now_GR.canCreate());

Scoped GlideRecord - canDelete()
Determines if the Access Control Rules, which include the user's roles, permit deleting records in this table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates whether the user's roles permit deleting of records in this table. Valid values:  
- true: Deleting permitted  
- false: Deleting is not permitted |

This code example writes whether the current user can delete records in the Incident table in the system log.

var att = new GlideRecord('sys_attachment');
gs.info(att.canDelete());

Scoped GlideRecord - canRead()
Determines if the Access Control Rules, which include the user's roles, permit reading records in this table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean  | Flag that indicates whether the user's roles permit reading of records in this table. Valid values:  
  • true: Reading permitted  
  • false: Reading is not permitted |

This code example writes whether the current user can read records in the Incident table in the system log.

```javascript
var now_GR = new GlideRecord('incident');
gs.info(now_GR.canRead());
```

### Scoped GlideRecord - canWrite()

Determines if the Access Control Rules, which include the user's roles, permit editing records in this table.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean  | Flag that indicates whether the user's roles permit writing of records in this table. Valid values:  
  • true: Writing permitted  
  • false: Writing is not permitted |

This code example writes whether the current user can delete records in the Incident table in the system log.

```javascript
var now_GR = new GlideRecord('incident');
gs.info(now_GR.canWrite());
```
Scoped/Global GlideRecord - chooseWindow(Number firstRow, Number lastRow, Boolean forceCount)

Sets a range of rows to be returned by subsequent queries.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>firstRow</td>
<td>Number</td>
<td>First row to include. Because the index starts at 0, a value of 0 returns the first row.</td>
</tr>
<tr>
<td>lastRow</td>
<td>Number</td>
<td>0-based row number of the first row NOT to return. Behaves similar to Java's String.substring(a,b) method. For example, if lastRow = 4 and firstRow = 2, two records are returned (4-2).</td>
</tr>
</tbody>
</table>
| forceCount| Boolean| Optional. Flag that indicates whether to force a row count query. In most implementations of this call, the row count is performed. There are some outlying cases, such as text searches, were a row count is not performed. Setting this flag ensures that the row count occurs. Valid values:
  • true: Row count always occurs.
  • false: Row count occurs if implemented in normal execution of method. Default: false |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example show returning records 3 and 4 from the Incident table.

```javascript
var now_GR = new GlideRecord('incident');
now_GR.orderBy('number');
now_GR.chooseWindow(2, 4);
now_GR.query();
while (now_GR.next()) {
  gs.info(now_GR.getValue('number') + ' is within window');
}
```
Output

```nohighlight
*** Script: INC00000003 is within window
*** Script: INC00000004 is within window
```

**Scoped GlideRecord - deleteMultiple()**

Deletes multiple records that satisfy the query condition.

This method does not delete attachments.

Do not use `deleteMultiple()` on tables with currency fields, delete these records individually. Also, do not use this method with the `chooseWindow()` or `setLimit()` methods when working with large tables.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```nohighlight
var now_GR = new GlideRecord('incident');
now_GR.addQuery('active','false'); // To delete all inactive incidents
now_GR.query();
now_GR.deleteMultiple();
```

**Scoped GlideRecord - deleteRecord()**

Deletes the current record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean  | Flag that indicates whether the record was successfully deleted. Valid values:  
  • true: Record was deleted.  
  • false: No record was found to delete.                                                                                                                                 |

```javascript
var now_GR = new GlideRecord('incident');
// to delete one record
if (now_GR.get('99ebb4156fa831005be8883e6b3ee4b9'))
  now_GR.deleteRecord();
```

Scoped GlideRecord - get(Object name, Object value)

Returns the specified record in the current GlideRecord object.

This method accepts either one or two parameters. If only a single parameter is passed in, the method assumes that it is the sys_id of the desired record. If not found, it then tries to match the value against the display value. If two parameters are passed in, the first is the name of the column within the GlideRecord to search. The second is the value to search for.

If multiple records are found, use `next()` to access the additional records.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Object</td>
<td>Optional. Name of the instantiated GlideRecord column to search for the specified <code>value</code> parameter. If only a single parameter is passed in, the method assumes that this parameter is the sys_id or display value.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>Value to match.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Indicates whether the requested record was located:</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• true:</td>
<td>record was found</td>
</tr>
<tr>
<td>• false:</td>
<td>record was not found</td>
</tr>
</tbody>
</table>

This example shows how to obtain an incident record by passing in the sys_id.

```javascript
var grIncident = new GlideRecord('incident');
var returnValue = grIncident.get('99ebb4156fa831005be8883e6b3ee4b9');
gs.info(returnValue); // logs true or false
gs.info(grIncident.short_description); // logs Incident Short description
```

This example shows how to obtain an incident record by passing the field to search (caller_id.name) and the value to match within that field.

```javascript
var grIncident = new GlideRecord('incident');
var returnValue = grIncident.get('caller_id.name','Sylivia Wayland');
gs.info(returnValue); // logs true or false
gs.info(grIncident.getValue('number')); // logs Incident Number
```

**Scoped GlideRecord - getAttribute(String fieldName)**

Returns the dictionary attributes for the specified field.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Field name for which to return the dictionary attributes</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Dictionary attributes</td>
</tr>
</tbody>
</table>

```javascript
function doit() {
    var now_GR = new GlideRecord('sys_user');
    now_GR.query("user_name","admin");
    if (now_GR.next()) {
        gs.info("we got one");
    }
}
Scoped GlideRecord - getClassDisplayValue()

Returns the current table's label.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Label that identifies the table.</td>
</tr>
</tbody>
</table>

This example prints the label of the passed in table.

```javascript
// Display the incident table label
var now_GR = new GlideRecord("incident");
var value = now_GR.getClassDisplayValue();
gs.info("The table label is " + value + ".");
```

Output:

```
The table label is Incident.
```

Scoped GlideRecord - getDisplayValue()

Retrieves the display value for the current record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Display value for the current record.</td>
</tr>
</tbody>
</table>

This example writes the display value of a specified incident record into the log.

```javascript
var now_GR = new GlideRecord('incident');
now_GR.get('sys_id','<sys_id>');
gs.info(now_GR.getDisplayValue());
```

Output:

```
INC0000050
```

Scoped GlideRecord - getED()

Returns the element's descriptor.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideElementDescriptor</td>
<td>The element's descriptor.</td>
</tr>
</tbody>
</table>

This example shows how to retrieve the name field for an incident GlideRecord.

```javascript
var grInc = new GlideRecord('incident');
grInc.get('sys_id','ef43c6d40a0b5700c77f9bf387afe3');

var field = grInc.getElementById('priority');
var ed = field.getED();

var isEdge = ed.getLabel();
gs.info("Label is - " + isEdge);
```

Output:

```
Label is - Priority
```
Scoped GlideRecord - getElement(String fieldName)

Retrieves the GlideElement object for the specified field.

The value returned by this method is a complete GlideElement object. The results are the equivalent of dot-walking a field value. For example, `now_GR.getElement('short_description')` provides the same result as `nowGR.short_description`.

In most cases, do not use dot-walking to get values from a record. Dot-walking retrieves the entire object instead of the field value. Retrieving the object uses much more storage and might cause undesirable results when used in arrays or in Service Portal.

Instead of retrieving the entire object, you can use one of the following methods to copy the field values:

- `getValue()`
- `getDisplayValue()`

If dot-walking through a GlideElement object is necessary, use the `toString()` method to retrieve values. For example, you might need the current caller's manager sys_id to set another reference field. The following example shows how to get the string value instead of the entire object:

```javascript
var mgr = current.caller_id.manager.toString();
```

See also:

- GlideElement – `toString()`
- `getElements()`

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Column name for which to return the GlideElement object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideElement</td>
<td>The GlideElement for the specified column of the current record. Each object describes a field in the current GlideRecord.</td>
</tr>
</tbody>
</table>
The following example shows how to add a new incident record with details in the **Short Description** field.

```javascript
var elementName = 'short_description';
var now_GR = new GlideRecord('incident');
now_GR.newRecord();
now_GR.setValue(elementName, "My DB is not working");
now_GR.insert();

var sdesc = now_GR.getElement('short_description');
gs.info(sdesc.getValue());
```

Output:

```
My DB is not working
```

**Scoped GlideRecord - getElements()**

Returns an array of GlideElement objects. Each object describes a field in the current GlideRecord.

If dot-walking through a GlideElement object is necessary, use the `toString()` method to retrieve values. For example, you might need the current caller's manager sys_id to set another reference field. The following example shows how to get the string value instead of the entire object:

```javascript
var mgr = current.caller_id.manager.toString();
```

See also:

- GlideElement – toString()
- getElement()

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array of GlideElement objects. Each object describes a field in the current GlideRecord.</td>
</tr>
</tbody>
</table>
The following example displays the value of the name field for the five most recent records created in the Question [question] table.

```javascript
var now_GR = new GlideRecord('question');
var elementArr = now_GR.getElements();
now_GR.orderByDesc('sys_created_on');
now_GR.setLimit(5);
now_GR.query();

while (now_GR.next()){
    var qNames = now_GR.name.toString();
    elementArr.push(qNames);
    gs.info(qNames);
}
```

Output:

delivery_time_var
delivery_time
sequence
priority
assigned_group

Scoped GlideRecord - getEncodedQuery()

Retrieves the query condition of the current result set as an encoded query string.

For details, see [Encoded query strings](#).

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('incident');
now_GR.addQuery('active', true);
now_GR.addQuery('priority', 1);
```
now_GR.query();
var encodedQuery = now_GR.getEncodedQuery();
gs.info(encodedQuery);

Output: active=true\^priority=1

**Scoped GlideRecord - getLabel()**

Returns the field's label.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Field's label</td>
</tr>
</tbody>
</table>

```javascript
template.print("Summary of Requested items:\n");
var now_GR = new GlideRecord("sc_req_item");
now_GR.addQuery("request", current.sysapproval);
now_GR.query();
while(now_GR.next()) {
    var nicePrice = now_GR.price.toString();
    if (nicePrice !="") {
        nicePrice = parseFloat(nicePrice);
        nicePrice = nicePrice.toFixed(2);
    }
    template.print(now_GR.number + ":  " + now_GR.quantity + " X " +
    now_GR.cat_item.getDisplayValue()
    + " at $" + nicePrice + " each \n");
template.print("    Options:\n")
for (key in now_GR.variables) {
    var now_V = now_GR.variables[key];
    if(now_V.getGlideObject().getQuestion().getLabel() !="") {
        template.space(4);
        template.print('     ' +  now_V.getGlideObject().getQuestion().getLabel() + " = "+
        now_V.getDisplayValue() + "\n");
    }
}
Scoped GlideRecord - getLastErrorMessage()
Retrieves the last error message. If there is no last error message, null is returned.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The last error message as a string.</td>
</tr>
</tbody>
</table>

```
// Setup a data policy where short_description field in incident is mandatory
var now_GR = new GlideRecord('incident');
now_GR.insert(); // insert without data in mandatory field
var errormessage = now_GR.getLastErrorMessage();
gs.info(errormessage);
```

Output: Data Policy Exception: Short description is mandatory

Scoped GlideRecord - getLink(Boolean noStack)
Retrieves the link to the current record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| noStack| Boolean    | Flag indicating whether to append the sysparm_stack parameter to the returned link. This parameter specifies the page to visit after closing the current link. Valid values:  
  • true: Do not attach the sysparm_stack parameter.  
  • false: Attach the sysparm_stack parameter. |
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>If true, the sysparm_stack parameter is not appended to the link.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Link to the current record.</td>
</tr>
</tbody>
</table>

This example queries all Incident records with a priority of "1" and writes the servlet URI and the current record's link to the system log.

```javascript
var now_GR = new GlideRecord('incident');
now_GR.addActiveQuery();
now_GR.addQuery("priority", 1);
now_GR.query();
now_GR.next();
gs.info(gs.getProperty('glide.servlet.uri') + now_GR.getLink(false));
```

Output:

https://instance.service-now.com/incident.do?sys_id=46e2fee9a9fe19810049b49dee0daf58&sysparm_stack=incident_list.do?sysparm_query=active=true

---

**Scoped GlideRecord - getRecordClassName()**

Retrieves the class name for the current record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The class name.</td>
</tr>
</tbody>
</table>
var now_GR = new GlideRecord('incident');
var recordClassName = now_GR.getRecordClassName();
gs.info(recordClassName);

Output: incident

**Scoped GlideRecord - getRowCount()**

Retrieves the number of rows in the query result.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Number</td>
</tr>
</tbody>
</table>

var now_GR = new GlideRecord('incident');
now_GR.query();
gs.info("Records in incident table: " + now_GR.getRowCount());

**Scoped GlideRecord - getTableName()**

Retrieves the name of the table associated with the GlideRecord.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

var now_GR = new GlideRecord('incident');
gs.info(now_GR.getTableName());
Scoped GlideRecord - getUniqueValue()

Gets the primary key of the record, which is usually the sys_id unless otherwise specified.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The unique primary key as a String, or null if the key is null.</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('kb_knowledge');
now_GR.query();
now_GR.next();
var uniqueid = now_GR.getUniqueValue();
gs.info(uniqueid);
```

Scoped GlideRecord - getValue(String name)

Retrieves the string value of an underlying element in a field.

⚠️ **Note:** If the Column Level Encryption Enterprise plugin is enabled and the instance has access to the key, this method returns clear text values from encrypted fields. If the instance does not have access to the key, this method returns the encrypted value. For more information, see Column Level Encryption Enterprise.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the field to get the value from.</td>
</tr>
</tbody>
</table>
## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String | The string value of the underlying element. Returns null if the field is empty or the field does not exist. Boolean values return as "0" and "1" string values instead of false and true.

```javascript
var now_GR = new GlideRecord('incident');
now_GR.orderBy('number');
now_GR.query('active','true');
now_GR.next();
gs.info(now_GR.getValue('number'));
```

**Output:**

INC0000002

### Scoped GlideRecord - GlideRecord(String tableName)

Creates an instance of the GlideRecord class for the specified table.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>The table to be used.</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('incident');
```

### Scoped GlideRecord - hasNext()

Determines if there are any more records in the GlideRecord object.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if there are more records in the query result set.</td>
</tr>
</tbody>
</table>
Scoped GlideRecord - insert()

Inserts a new record using the field values that have been set for the current record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Unique ID of the inserted record, or null if the record is not inserted.</td>
</tr>
</tbody>
</table>

Scoped GlideRecord - initialize()

Creates an empty record suitable for population before an insert.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var grIncident = new GlideRecord('incident');
grIncident.initialize();
grIncident.setValue('short_description', 'New Incident');
grIncident.setValue('description', 'Incident description');
grIncident.insert();

Scoped GlideRecord - isActionAborted()
Checks to see if the current database action is to be aborted.

isActionAborted() is initialized (set to false) for new threads and by the next() method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean| Flag that indicates if the current database action is to be aborted. Valid values:  
• true: The current database action is to be aborted.  
• false: The current database action is not to be aborted. |

var now_GR = new GlideRecord('incident');

gs.info(now_GR.isActionAborted());

Output:

false

Scoped GlideRecord - isEncodedQueryValid(String query)
Verifies whether the specified encoded query is valid.

If the specified encoded query is valid, then the query is applied, just as if you had called addEncodedQuery(). If the specified encoded query is invalid, then sys_idNotValidnull is added as the encoded query.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>String</td>
<td>Encoded query to validate. See <a href="#">Encoded query strings</a>.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the specified encoded query is valid.</td>
</tr>
<tr>
<td>true</td>
<td>Passed-in encoded query is valid</td>
</tr>
<tr>
<td>false</td>
<td>Passed-in encoded query is not valid</td>
</tr>
</tbody>
</table>

This code example shows how to validate an encoded query and then execute logic if the query is valid.

```javascript
var now_GR = new GlideRecord('incident_sla');
var isValidQuery = now_GR.isEncodedQueryValid('inc_impact=1^taskslatable_active=true');
if (isValidQuery) {
    now_GR.query();
}
```

**Scoped GlideRecord - isNewRecord()**

Checks if the current record is a new record that has not yet been inserted into the database.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the record is new and has not been inserted into the database.</td>
</tr>
</tbody>
</table>
```javascript
var now_GR = new GlideRecord("x_app_table");
now_GR.newRecord(); // create a new record and populate it with default values
gs.info(now_GR.isNewRecord());
```

**Scoped GlideRecord - isValid()**

Determines if the current table is valid or if the record was successfully retrieved.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates if the table is valid or if the record was successfully retrieved. Possible values:  
|         | • true: Table is valid or the record was successfully retrieved.  
|         | • false: Table is invalid or the record was not successfully retrieved.     |

```javascript
var comment_GR = new GlideRecord('cf_comment');
var commentId = '99ebb4156fa831005be8883e6b3ee4b9';
comment_GR.get(commentId);
gs.info(comment_GR.isValid());
```

**Scoped GlideRecord - isValidField(String columnName)**

Determines if the specified field is defined in the current table.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columnName</td>
<td>String</td>
<td>The name of the field.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>--------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Boolean</td>
<td>True if the field is defined for the current table.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('incident');
now_GR.initialize();
gs.info(now_GR.isValidField("short_description"));
```

**Scoped GlideRecord - isValidRecord()**

Determines if a record was actually returned by the query/get record operation.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether a record was actually returned by the query/get operation. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Record returned by query/get operation.</td>
</tr>
<tr>
<td></td>
<td>• false: End of record set, no record returned.</td>
</tr>
</tbody>
</table>

```javascript
var rec = new GlideRecord('incident');
rec.query();
while (rec.next()) {
    gs.info(rec.number + ' exists');
}
gs.info(rec.isValidRecord());
```

**Scoped GlideRecord - newRecord()**

Creates a new GlideRecord record, sets the default values for the fields, and assigns a unique ID to the record.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord("x_app_table");
now_GR.newRecord();
gs.info(now_GR.isNewRecord());
```

Output: true

 Scoped GlideRecord - next()  
 Moves to the next record in the GlideRecord object.

ℹ️ Note: This method fails if there is a field in the table called "next". If that is the case, use the method Scoped GlideRecord - _next().

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates if there is a "next" record in the GlideRecord. Valid values:  
• true: Move to the next record was successful.  
• false: No more records in the result set. |

Example

```javascript
var rec = new GlideRecord('incident');
rec.query();
```
while (rec.next()) {
    gs.info(rec.getValue('number') + ' exists');
}

Output:

INC0010112 exists
INC0010114 exists
INC0010119 exists
INC0010127 exists

**Scoped GlideRecord - _next()**

Moves to the next record in the GlideRecord. Provides the same functionality as `next()`, use this method if the GlideRecord has a column named `next`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean| Flag that indicates whether there are more records in the query set. Possible values:  
  • true: More records in the query set.  
  • false: No more records in the query set. |

The following example shows how to list each record in the Incidents [incident] table.

```javascript
var rec = new GlideRecord('incident');
rec.query();
while (rec._next()) {
    gs.info(rec.getValue('number') + ' exists');
}
```

Output:

INC0000060 exists
INC0009002 exists
INC0000009 exists
Scoped GlideRecord - operation()

Determines if an operation is insert, update, or delete.

Knowing the operation enables using `current.operation()` to make a generic business rule which can handle each operation uniquely.

For information on using the global variable `current`, refer to Global variables in business rules.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The current operation. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• delete</td>
</tr>
<tr>
<td></td>
<td>• insert</td>
</tr>
<tr>
<td></td>
<td>• update</td>
</tr>
</tbody>
</table>

The following example shows how to use this method in a business rule. The `operation()` method detects which operation triggered an event and performs a different action depending for update and insert operations.

```java
if(current.operation() == "update") {
    current.updates ++; }
if(current.operation() == "insert") {
    current.updates = 0; }
```

Scoped GlideRecord - orderBy(String name)

Specifies an orderBy column.
Call this method more than once to order by multiple columns. Results are arranged in ascending order. To arrange records in descending order, see **Scoped GlideRecord - orderByDesc(String name)**.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Column name to use to order the records in this GlideRecord object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to order incident records in ascending order by Short Description.

```javascript
var queryString = "priority=2";
var now_GR = new GlideRecord('incident');
now_GR.orderBy('short_description'); // Ascending Order
now_GR.addEncodedQuery(queryString);
now_GR.query();
while (now_GR.next()) {
  gs.info(now_GR.getValue('short_description'));
}
```

Output:

- Can't launch 64-bit Windows 7 virtual machine
- Can't log into SAP from my laptop today
- Network storage unavailable
- Please remove the latest hotfix from my PC

**Scoped GlideRecord - orderByDesc(String name)**

Specifies a descending orderBy column.

Call this method more than once to order by multiple columns. Results are arranged in descending order. To arrange records in ascending order, see **Scoped GlideRecord - orderBy(String name)**.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Column name to use to order the records in a GlideRecord object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to order incident records in descending order by Short Description.

```javascript
var queryString = "priority=2";
var now_GR = new GlideRecord('incident');
now_GR.orderByDesc('short_description'); //Descending Order
now_GR.addEncodedQuery(queryString);
now_GR.query();
while (now_GR.next()) {
    gs.info(now_GR.getValue('short_description'));
}
```

Output:

Please remove the latest hotfix from my PC
Network storage unavailable
Can't log into SAP from my laptop today
Can't launch 64-bit Windows 7 virtual machine

**Scoped GlideRecord - query(String field, String value)**

Runs a query against the table based on the filters specified by query methods such as `addQuery()` and `addEncodedQuery()`.

This method queries the GlideRecord table as well as any references of the table. For more information, see [Querying tables in script](#).

ℹ️ **Note:** This method fails if there is a field in the table called "query". If that is the case, use the `_query()` method instead.

See also:
- GlideAggregate - Scoped
- GlideQuery - Scoped, Global
- GlideQueryCondition - Scoped

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Optional - must also specify value parameter. Name of the field to search for the value specified in the value parameter.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Optional - must also specify field parameter. Value to search for in the specified field parameter.</td>
</tr>
</tbody>
</table>

**Note:** This method is typically run without arguments, but you can specify a name-value pair to filter records containing the specified values. If the parameters are specified, the "name=value" condition is added to the query.

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to scan the System Dictionary [sys_dictionary] for tables with a query field.

```javascript
var tableArr = [];

var now_GR = new GlideRecord('sys_dictionary');
now_GR.addQuery('element', 'query');
now_GR.setLimit(6);

now_GR.query();

while(now_GR.next()){
    tableArr.push(now_GR.name.getValue());
}
```
gs.info('The following tables have a field column called "query":');

for (i = 0; i < tableArr.length; i++) {
    gs.info(tableArr[i]);
}

Output:

The following tables have a field column called "query":
- cmdb_convert_bulk_services
- cmdb_multisource_query_status
- cmdb_qb_result_base
- cmdb_qb_table_mapping
- discovery_probes_cim_query
- kb_feedback

**Scoped GlideRecord - _query(String field, String value)**

Runs a query against the table based on the filters specified by query methods such as `addQuery()` and `addEncodedQuery()`. This method is intended to be used on tables in which there is a column named “query”, which might cause errors running the `query()` method.

This method queries the GlideRecord table as well as any references of the table. For more information, see [Querying tables in script](#).

See also:
- [GlideAggregate - Scoped](#)
- [GlideQuery - Scoped, Global](#)
- [GlideQueryCondition - Scoped](#)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Optional - must also specify value parameter. Name of the field to search for the value specified in the value parameter.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>Optional - must also specify field parameter. Value to search for in the specified field parameter.</td>
</tr>
</tbody>
</table>

Note: This method is typically run without arguments, but you can specify a name-value pair to filter records containing the specified values. If the parameters are specified, the "name=value" condition is added to the query.

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how query the Knowledge Feedback [kb_feedback] table and list KB articles with comments that include Excel.

```javascript
var rec = new GlideRecord('kb_feedback');
rec.addQuery('comments', 'CONTAINS', 'Excel');
rec._query();
while (rec.next()) {
  gs.info(rec.getDisplayValue('article') + " comment: " + rec.getValue('comments'));}
```

Output:

```javascript
KB0000005 comment:
    Can you please add the version of Excel this applies to? All?
KB0000005 comment:
    Does this work for all Excel versions? OSX and Windows alike?
```

Scoped GlideRecord - setAbortAction(Boolean b)

Sets a flag to indicate if the next database action (insert, update, delete) is to be aborted. This is often used in business rules.

Use in an onBefore business rule to prevent the database action from being done. The business rule continues to run after setAbortAction() is called. Calling
setAbortAction() does not stop subsequent business rules from executing. Calling this method only prevents the database action from occurring.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>True to abort the next action. False if the action is to be allowed.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
// Often used in business rule to check whether the current operation should be aborted.
if (current.size > 16) {
    current.setAbortAction(true);
}
```

### Scoped GlideRecord - setLimit(Number maxNumRecords)

Sets the limit for number of records are fetched by the GlideRecord query.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxNumRecords</td>
<td>Number</td>
<td>The maximum number of records to fetch.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var now_GR = new GlideRecord('incident');
now_GR.orderByDesc('sys_created_on');
now_GR.setLimit(10);
now_GR.query(); // this retrieves latest 10 incident records created
```

### Scoped GlideRecord - setNewGuidValue(String guid)

Sets the sys_id value for the current record.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>guid</td>
<td>String</td>
<td>GUID to assign to the current record.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('incident');
now_GR.setValue('short_description', 'The third floor printer is broken');
now_GR.setNewGuidValue('eb4636ca6f6d31005be8883e6b3ee333');
now_GR.insert();
gs.info(now_GR.getValue('sys_id'));
```

Scoped GlideRecord - setValue(String name, Object value)
Sets the value of the field with the specified name to the specified value.

Normally the script does `now_GR.category = value`. However, if the element name is itself a variable then you can use `now_GR.setValue(elementName, value)`. When setting a value, ensure the data type of the field matches the data type of the value you enter.

⚠️ **Note:** If the Column Level Encryption Enterprise plugin is enabled and the instance has access to the key, this method can insert encrypted data into encrypted fields. If the instance does not have access to the key, this method returns an error. For more information, see Column Level Encryption Enterprise.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the field.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>Value to assign to the field.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var elementName = 'short_description';
var now_GR = new GlideRecord('incident');
now_GR.initialize();
now_GR.setValue(elementName, "My DB is not working");
now_GR.insert();
```

Scoped GlideRecord - `setWorkflow(Boolean enable)`

Enables or disables the running of business rules, script engines, and audit.

⚠️ **Warning:** Disabling the running of business rules, script engines, and audit can have a significant impact on your ServiceNow® instance and how it operates. Ensure that you thoroughly test this change before deploying it to production.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| enable | Boolean | Flag that indicates whether to enable or disable the running of business rules, script engines, and audit. Valid values:  
• true: Enable the running of business rules, script engines, and audit. (Default)  
• false: Disable the running of business rules, script engines, and audit. |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
//Enable business rules, scripts engines for x_app_table
var now_GR = new GlideRecord("x_app_table");
now_GR.setWorkflow(true);
```
Scoped GlideRecord - update(String reason)

Updates the GlideRecord with any changes that have been made. If the record does not already exist, it is inserted.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reason</td>
<td>String</td>
<td>Optional. Reason for the update. The reason appears in the audit record.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the new or updated record. Returns null if the update fails.</td>
</tr>
</tbody>
</table>

The following example shows how to update the Short Description field of an incident.

```javascript
var now_GR = new GlideRecord('incident');
now_GR.get('99ebb4156fa831005be8883e6b3ee4b9');
now_GR.setValue('short_description', 'Update the short description');
now_GR.update();
gs.info(now_GR.getElement('short_description'));
```

Output:

Update the short description.

Scoped GlideRecord - updateMultiple()

Updates each GlideRecord in a stated query with a specified set of changes.

This method does not support adding multiple journal entries.

ℹ️ Note: To ensure expected results, use the setValue() method instead of direct assignments. That is, use `gr.setValue('<field_name>', '4');` instead of `gr.<field_name> = 4`.

ℹ️ Note: Do not use this method with the chooseWindow() or setLimit() methods when working with large tables.

This method sets new values and does not clear existing values. To clear an existing value, use the setValue() method and set the field to `null`.
This example shows how to update the state of all active incidents to 4 - "Awaiting User Info".

```javascript
var now_GR = new GlideRecord('incident');
now_GR.addQuery('active', true);
now_GR.setValue('state', 4);
now_GR.updateMultiple();
```

### GlideRecordUtil - Global

A utility class for working with GlideRecords

The GlideRecordUtil class is available in server-side scripts.

#### GlideRecordUtil - getCIGR(String sys_id)

Returns the GlideRecord object for the specified configuration item (CI) using just the sys_id of the CI.

Use this method to quickly obtain a specific CI without knowing its associated class/table.
The following example returns the GlideRecord object for the specified CI using just the sys_id of the CI.

```javascript
var now_GR = new GlideRecordUtil().getCIGR("2dfd7c8437201000deeabfc8bcb65d56");
```

**GlideRecordUtil - getFields(GlideRecord gr)**

Returns an array of all the fields in the specified GlideRecord.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>gr</td>
<td>GlideRecord</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>Array</td>
<td>Field names for the specified GlideRecord.</td>
</tr>
</tbody>
</table>

```javascript
var queryString = "priority=1^ORpriority=2";
var now_GR = new GlideRecord('incident');
now_GR.addEncodedQuery(queryString);
now_GR.query();
now_GR.next();

var gRU = new GlideRecordUtil();
var fieldList = gRU.getFields(now_GR);
gs.info(fieldList);
```

Output: Line breaks added for presentation.

```javascript
sys_id,skills,closed_by,assigned_to,contract,category,escalation,state,reassignment_count,location,
time_worked,order,due_date,number,upon_approval,sys_tags,sla_due,follow_up,reopen_count,notify,business_stc,
caused_by,rejection_goto,assignment_group,comments_and_work_notes,incident_state,opened_at,
parent_incident,
business_service,wf_activity,calendar_duration,group_list,caller_id,comments,priority,sys_updated_by,
variables,delivery_task,resolved_at,sys_updated_on,parent,active,opened_by,expected_start,watch_list,
company,upon_reject,work_notes,sys_created_by,additional_assignee_list,approval_set,cmdb_ci,user_input,
```
GlideRecordUtil - getGR(String baseTable, String sys_id)

Returns a GlideRecord instance for the given table, positioned to the given sys_id, and of the right class (table).

This method is useful when you need to load a GlideRecord from a sys_id, but you don’t know what the actual table is (because it may be extended from the base table). This method always returns a GlideRecord of the correct type.

**base_table**: the name of the base table that the specified sys_id is in.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseTable</td>
<td>String</td>
<td>The name of the base table containing the sys_id.</td>
</tr>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The sys_id of the desired record.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The GlideRecord for the specified sys_id.</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecordUtil().getGR("cmdb_ci_computer", "2dfd7c8437201000deeabfc8bcbe5d56");
```

GlideRecordUtil - getTables(String tableName)

Returns a Java ArrayList of the ancestors of the specified table name.

For example, given cmdb_ci_linux_server, this would return cmdb_ci_linux_server, cmdb_ci_server, cmdb_ci_computer, cmdb_ci_hardware, cmdb_ci, and cmdbr.
The following example shows the ancestors of the cmdb_ci_linux_server table.

```plaintext
var tables = new GlideRecordUtil().getTables("cmdb_ci_linux_server");
gs.info("Tables returned: " + tables);
```

**Output:**

```
Tables returned: [cmdb_ci_linux_server, cmdb_ci_server, cmdb_ci_computer, cmdb_ci_hardware,
                 cmdb_ci, cmdb]
```

**GlideRecordUtil - mergeToGR(Object hashMap, GlideRecord now_GR, Object ignore)**

Sets the fields in the specified GlideRecord with the field values contained in the specified hashmap, unless that field name is in the ignore hashmap.

```
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hashMap</td>
<td>Object</td>
<td>An Object instance (being used as a hashmap), with properties named for fields and containing the fields' value.</td>
</tr>
<tr>
<td>GlideRecord</td>
<td>GR</td>
<td>The GlideRecord instance to receive the field values.</td>
</tr>
<tr>
<td>ignore</td>
<td>Object</td>
<td>An optional hashmap of field names to ignore.</td>
</tr>
</tbody>
</table>
```
This example updates a computer record's name and os fields, but does not update the sys_created_by field:

```javascript
var now_GR = new GlideRecordUtil().getGR("cmdb_ci_computer", "2dfd7c8437201000deeabfc8bce5d56");
var obj = {"name": "xyz", "os": "windows 2000", "sys_created_by": "sleek.lin"};
var ignore = {"sys_created_by": true};
new GlideRecordUtil().mergeToGR(obj, now_GR, ignore);
now_GR.update();
```

**GlideRecordUtil - populateFromGR(Object hashMap, GlideRecord now_GR, Object ignore)**

Populates the given hashmap from the given GlideRecord instance. Each field in the GlideRecord becomes a property in the hashmap.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hashMap</td>
<td>Object</td>
<td>An object being used as a hashmap.</td>
</tr>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>A GlideRecord instance positioned to a valid record.</td>
</tr>
<tr>
<td>ignore</td>
<td>Object</td>
<td>An optional hashmap of file names not to populate.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var objectToPopulate = {};
var now_GR = new GlideRecordUtil().getGR("cmdb_ci_computer", "2dfd7c8437201000deeabfc8bce5d56");
var ignore = {"sys CREATED ON": true, "sys UPDATED BY": true};
new GlideRecordUtil().populateFromGR(objectToPopulate, now_GR, ignore);
// Now the objectToPopulate contains field/value pairs from the computer GlideRecord
```

**GlideRecordV3 - Client**

GlideRecord is used for database operations. Client-side GlideRecord enables the use of some GlideRecord functionality in client-side scripts, such as client scripts and UI policy scripts.

A GlideRecord contains both records and fields.
Queries made with the client-side GlideRecord are executed on the server. Therefore, a request is made from the client browser to obtain the record data. Client-side GlideRecord is not supported in scoped applications. Instead, create a script include and use GlideAjax, or use the REST APIs.

**GlideRecordV3 - addOrderBy(String column)**

Adds a column to order by in the query.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>column</td>
<td>String</td>
<td>The column by which to order the result set.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideRecordV3 - addQuery(String fieldName, Object value)**

Adds a filter to return records where the field is equal to the value (or is in a list of values).

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field to be checked.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>The value or list of values on which to query.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideRecordV3 - addQuery(String fieldName, Object operator, Object value)**

Adds a filter to return records where the field meets the specified condition (field, operator, value).
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field to be checked.</td>
</tr>
<tr>
<td>operator</td>
<td>Object</td>
<td>An operator for the query.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>The value to use.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### GlideRecordV3 - deleteRecord(Function responseFunction)

Deletes the current record.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>responseFunction</td>
<td>Function</td>
<td>The response function.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the record was deleted. False if no record was found to delete.</td>
</tr>
</tbody>
</table>

### GlideRecordV3 - get(String sysId)

Get a record by sysId.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysId</td>
<td>String</td>
<td>The sysId of the record for which to search.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Boolean</td>
<td>True if one or more matching records was found. False if no records were found.</td>
<td></td>
</tr>
</tbody>
</table>

**GlideRecordV3 - getEncodedQuery()**

Retrieves all query conditions as an encoded query string.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>An encoded query string containing all conditions that have been added to the query.</td>
</tr>
</tbody>
</table>

**GlideRecordV3 - getTableName()**

Gets the name of the table associated with the GlideRecord.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The table name.</td>
</tr>
</tbody>
</table>

**GlideRecordV3 - GlideRecord(String tableName)**

Creates an instance of the GlideRecord class for the specified table.
var now_GR = new GlideRecord('incident');

**GlideRecordV3 - hasNext()**
Determines if there are any more records in the GlideRecord.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if there are more records in the query set.</td>
</tr>
</tbody>
</table>

**GlideRecordV3 - insert(Function responseFunction)**
Inserts a new record using the field values that have been set for the current record.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>responseFunction</td>
<td>Function</td>
<td>The response function.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_id of the inserted record, or null if the record was not inserted.</td>
</tr>
</tbody>
</table>

**GlideRecordV3 - next()**
Moves to the next record in the GlideRecord.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>False if there are no more records in the query set.</td>
</tr>
</tbody>
</table>

**GlideRecordV3 - orderBy(String column)**

Specifies an orderBy column. May be called more than once to order by multiple columns.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>column</td>
<td>String</td>
<td>The column to add to sort the result set.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideRecordV3 - query()**

Performs a query using the current query conditions. Takes zero or more parameters. Parameters may be in any order. Any function is considered to be a response function. Any pair of literals is considered a query pair (field : value).

Do not make synchronous query calls. Performing a query without a response function makes the call synchronous, which means that the display will wait for the query response before continuing.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of a field to query. (optional)</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>responseFunction</td>
<td>Function</td>
<td>The function called when the query results are available. (optional)</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The field value to query for. (optional)</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
// synchronous call syntax (no response function): DO NOT USE
query();

// asynchronous call syntax
// performs query with current conditions, calls responseFunction when done
query(responseFunction)

// synchronous call syntax (no response function): DO NOT USE
// adds "category=hardware" to current query conditions and performs query
query('category', 'hardware')

// asynchronous call syntax
// adds "category=hardware" to current query conditions, performs query, and calls responseFunction when done
query('category', 'hardware', responseFunction)

// asynchronous call example
// adds "user_name=abel.tuter" to current query conditions, performs query, and calls defined response function when done
function onLoad() {
    var now_GR = new GlideRecord("sys_user");
    now_GR.query("user_name", "abel.tuter", function(now_GR) {
        if (now_GR.next()) {
            alert("You can access fields by name from the client API, just like in the server API:\n            now_GR.name = " + now_GR.name);
            alert("You can also access fields using getValue():\n            now_GR.getValue("email") = " + now_GR.getValue("email"));
            if (now_GR.getDisplayValue) {
```
alert("In Service Portal, Mobile, and Agent Workspace, you can
access a field's display value:\n\n\nnow_GR.getDisplayValue("company\") = " +
now_GR.getDisplayValue("company");
} else {
    alert("On the desktop, you cannot access a field's display value, but can get its
sys_id:\nnow_GR.company = " + now_GR.company);
}
    alert("You cannot dot-walk in the client API:\nnow_GR.company.name = " +
now_GR.company.name);
}
});

GlideSchedule - Scoped

The scoped GlideSchedule API provides methods for performing operations on
GlideSchedule objects, such as adding new schedule segments to a schedule,
determining if a datetime is within the schedule, or setting the schedule
timezone.

Scoped GlideSchedule - add(GlideDateTime startDate, GlideDuration offSet)

Adds a new schedule segment to the current schedule.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>GlideDateTime</td>
<td>The starting date of the new schedule segment.</td>
</tr>
<tr>
<td>offSet</td>
<td>GlideDuration</td>
<td>The time offset of the new schedule segment.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDateTime</td>
<td>The schedule updated with the new schedule segment.</td>
</tr>
</tbody>
</table>

var startDate = new GlideDateTime('2014-01-02');
var days = 2;
var dur = new GlideDuration(60 * 60 * 24 * 1000 * days);
var schedule = new GlideSchedule();
var end = schedule.add(startDate, dur);
gs.info(end);
Scoped GlideSchedule - duration(GlideDateTime startDate, GlideDateTime endDate)
Determines the elapsed time in the schedule between two date time values using the timezone of the schedule or, if that is not specified, the timezone of the session.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>GlideDateTime</td>
<td>The starting datetime.</td>
</tr>
<tr>
<td>endDate</td>
<td>GlideDateTime</td>
<td>The ending datetime.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDuration</td>
<td>The difference between the starting and ending datetime.</td>
</tr>
</tbody>
</table>

```javascript
var startDate = new GlideDateTime('2014-10-16 02:00:00');
var endDate = new GlideDateTime('2014-10-18 04:00:00');
var schedule = new GlideSchedule();

schedule.load('090eeeca0a0a0b260077e1dfa71da828'); // loads "8-5 weekdays excluding holidays" schedule
var duration = schedule.duration(startDate, endDate);
gs.info(duration.getDurationValue()); // gets the elapsed time in schedule
```

Scoped GlideSchedule - getName()
Retrieves the schedule name.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the current schedule.</td>
</tr>
</tbody>
</table>

```java
sys_id = '04e664654a36232701a2247dcd8fc4cf'; // sys_id for "Application" schedule record
var sched = new GlideSchedule(sys_id);
gs.info(sched.getName());
```

**Scoped GlideSchedule - GlideSchedule()**

Instantiates an empty GlideSchedule object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Scoped GlideSchedule - GlideSchedule(String sysID, String timeZone)**

Instantiates a GlideSchedule object and loads the schedule information. If a timezone is not specified or is nil, the current session timezone is used.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysID</td>
<td>String</td>
<td>The system ID for the schedule.</td>
</tr>
<tr>
<td>timeZone</td>
<td>String</td>
<td>The time zone. (Optional)</td>
</tr>
</tbody>
</table>

```java
var schedule = new GlideSchedule('090eecae0a0a0b260077edfa71da828', 'US/Pacific');
```

**Scoped GlideSchedule - isInSchedule(GlideDateTime time)**

Determines if the specified date and time is within the current schedule.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>GlideDateTime</td>
<td>Date and time value to check.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the specified date and time is within the schedule. Valid values: • true: Date and time is within the schedule. • false: Date and time are outside of the schedule.</td>
</tr>
</tbody>
</table>

```java
var glide = new GlideRecord('cmn_schedule');
glide.addQuery('type', 'blackout');
glide.query();
if (glide.next()) {
    var sched = new GlideSchedule(glide.sys_id);
    var date = new GlideDateTime();
    date.setDisplayValue("2007-09-18 12:00:00");
    if (sched.isInSchedule(date))
        gs.info("Is in the schedule");
    else
        gs.info("Is NOT in the schedule");
}
```

Scoped GlideSchedule - isValid()

Determines if the current schedule is valid. A schedule is valid if it has at least one schedule span.

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the schedule is valid.</td>
</tr>
</tbody>
</table>

```java
var glide = new GlideRecord('cmn_schedule');
glide.addQuery('type', 'blackout');
glide.query();
if (glide.next()) {
    var sched = new GlideSchedule(glide.sys_id);
    var date = new GlideDateTime();
    date.setDisplayValue("2007-09-18 12:00:00");
    if (sched.isValid())
        gs.info("Is valid");
    else
        gs.info("Is NOT valid");
}
```
else
gs.info("Is not valid");
}

Scoped GlideSchedule - load(String sysID, String timeZone, String excludeSpanID)
Loads a schedule with the schedule information.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysID</td>
<td>String</td>
<td>The system ID of the schedule.</td>
</tr>
<tr>
<td>timeZone</td>
<td>String</td>
<td>(Optional) The timezone. If a timezone is not specified, or is nil, the session timezone is used for the schedule.</td>
</tr>
<tr>
<td>excludeSpanID</td>
<td>String</td>
<td>Any span to exclude.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var x = new GlideSchedule();
x.load('08fcd0830a0a0b2600079f56b1adb9ae');
```

Scoped GlideSchedule - setTimeZone(String timeZone)
Sets the timezone for the current schedule.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeZone</td>
<td>String</td>
<td>The timezone.</td>
</tr>
</tbody>
</table>
This example sets the timezone for the schedule to US/Pacific.

```javascript
var schedule = new GlideSchedule();
schedule.setTimeZone('US/Pacific');
```

**Scoped GlideSchedule - whenNext(GlideDateTime time, String timeZone)**

Determines how much time (in milliseconds) until start time of the next schedule item.

This function is intended to be called when the GlideSchedule object (cmn_schedule table) is not currently in the schedule window. The whenNext() call returns duration (in ms) until the GlideSchedule object is within the schedule. This function does not return a meaningful value if called when the GlideSchedule object is within the schedule.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>GlideDateTime</td>
<td>Time to be evaluated</td>
</tr>
<tr>
<td>timeZone</td>
<td>String</td>
<td>Timezone</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of milliseconds until the start time of the next schedule item. Returns -1 if never.</td>
</tr>
</tbody>
</table>

```javascript
var startDate = new GlideDateTime('2014-10-25 08:00:00');
var glideSchedule = new GlideSchedule('08fcd0830a0a0b2600079f56b1ad9ae', 'UTC');
gs.info(glideSchedule.whenNext(startDate));
```

Output:

172800000
testScript();

function testScript() {
var now = new GlideDateTime(); //current date and time
var sched = new GlideSchedule("<sys_id>"); // Use a cmn_schedule sys_id
if (sched.isInSchedule(now)) {
    gs.info('We are in an active schedule window so whenNext() is not helpful');
} else {
    gs.info('Not currently in schedule so call whenNext()');
    var msUntilNext = sched.whenNext(new GlideDateTime(), 'US/Pacific');
    gs.info('Next schedule starts in '+msUntilNext+' milliseconds');
}
} // Output [schedule inactive]:
  \ *** Script: Not currently in schedule so call whenNext()
  \ *** Script: Next schedule starts in -1 milliseconds

Output:

[Scheduled for future] *** Script: Not currently in schedule *** Script: Next schedule starts in 332894000 milliseconds

**GlideScriptableInputStream - Scoped, Global**

A GlideScriptableInputStream object cannot be instantiated directly, but is used as an opaque object which is used to connect an input stream from GlideSysAttachment.getContentStream() with other streaming APIs, such as GlideTextReader, GlideDigest, and XMLDocument2.

See the scoped GlideSysAttachment API for methods that return a GlideScriptableInputStream object. The scoped GlideTextReader constructor requires a GlideScriptableInputStream object as an input parameter.

**GlideScriptedProcessor - Scoped**

ServiceNow processors are equivalent to Java servlets.

Processors provide a customizable URL endpoint that can execute arbitrary server-side JavaScript code and produce output such as TEXT, JSON, or HTML. The ScopedGlideScriptedProcessor APIs are used in processor scripts to access the processor (servlet) capabilities. There are no constructors for the ScopedGlideScriptedProcessor APIs. The methods are called using the global variable g_processor.

A useful global variable, g_target, is available in processor scripts. It contains the table name extracted from the URL.
The URL to a processor has the format: https://<instance name.servicenow.com>/
<path endpoint>.do?<parameter endpoint>=<value> where the path endpoint and
parameter endpoint are defined on the processor form.

Scoped GlideScriptedProcessor - redirect(String url)
Redirects to the specified URL.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>Destination URL</td>
</tr>
</tbody>
</table>

//Do whatever processing you need and redirect to the homepage
g_processor.redirect("/navpage.do")

Scoped GlideScriptedProcessor - writeJSON(Object o)
Encodes an object as a JSON string and writes it to the current URL.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>o</td>
<td>Object</td>
<td>The object to encode to a JSON string.</td>
</tr>
</tbody>
</table>

var map = {"key1":"value1","key2":"value2"};
g_processor.writeJSON(map);
Scoped GlideScriptedProcessor - writeOutput(String s)
Writes the specified string to the current URL.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>String</td>
<td>The string to write.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var name = g_request.getParameter("name");
g_processor.writeOutput("Hello " + name);
```

Scoped GlideScriptedProcessor - writeOutput(String contentType, String s)
Writes the specified string to the current URL in the specified character-encoding.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contentType</td>
<td>String</td>
<td>Sets the content type of the response sent to the client, if the response has not been committed, and may include a character-encoding specification.</td>
</tr>
<tr>
<td>s</td>
<td>String</td>
<td>The string to write.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var name = g_request.getParameter("name");
g_processor.writeOutput("text/plain", "Hello " + name);
```
GlideSecureRandomUtil - Scoped

The scoped GlideSecureRandomUtil API provides methods for generating integers, long values, and strings.

There is no constructor for this class. Methods are accessed through the static object GlideSecureRandomUtil. The GlideSecureRandomUtil class is available in both global and scoped applications.

Scoped GlideSecureRandomUtil - getSecureRandomInt()

Generates a pseudo-random integer.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The pseudo-randomly generated integer.</td>
</tr>
</tbody>
</table>

```java
gs.info(GlideSecureRandomUtil.getSecureRandomInt());
```

Output:

1976146969

Scoped GlideSecureRandomUtil - getSecureRandomIntBound(Number bound)

Generates a pseudo-random integer between 0 (inclusive) and the bound (exclusive) value that you pass into the method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bound</td>
<td>Number</td>
<td>The bound value.</td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>Number</td>
<td>The pseudo-randomly generated integer.</td>
<td></td>
</tr>
</tbody>
</table>

```
gs.info(GlideSecureRandomUtil.getSecureRandomIntBound(100));
```

Output:

55

**Scoped GlideSecureRandomUtil - getSecureRandomLong()**

Generates pseudo-random long value.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>Number</td>
</tr>
</tbody>
</table>

```
gs.info(GlideSecureRandomUtil.getSecureRandomLong());
```

Output:

792836514424092500

**Scoped GlideSecureRandomUtil - getSecureRandomString(Number length)**

Generates a random alpha-numeric String with the specified length.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>length</td>
<td>Number</td>
<td>The length of the string in number of characters.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The randomly generated string.</td>
</tr>
</tbody>
</table>

```java
gs.info(GlideSecureRandomUtil.getSecureRandomString(12));
```

Output:

```
1XzVI0sLfvIT
```

GlideSecurityUtils- Scoped, Global

Provides methods to work with URLs.

Access these methods using the static object GlideSecurityUtils. This class is available in scoped and global scripts.

GlideSecurityUtils - cleanURL(String url)

Removes suspicious encoding to prevent reflected or DOM based cross site scripting.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>The URL to be checked.</td>
</tr>
</tbody>
</table>

```java
myurl='javascript%3Aalert(1)';
var clean=GlideSecurityUtils.cleanURL(myurl);
gs.info(clean);
```

Output: null

GlideSecurityUtils - enforceRelativeURL(String url)

Removes the domain address from the URL, which leaves the page name and parameters.
### GlideSecurityUtils - `enforceRelativeURL(String url)`

**Description**

The URL to be turned into a relative URL.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>The URL to be turned into a relative URL.</td>
</tr>
</tbody>
</table>

```java
myurl='http://evildomain.com/test.do';
relativeURL=GlideSecurityUtils.enforceRelativeURL(myurl);
gs.info(relativeURL);
```

Output: test.do

### GlideSecurityUtils - `escapeScript(String script)`

**Description**

Add escape characters to a script to prevent cross-site scripting.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>script</td>
<td>String</td>
<td>The script to have escape characters added.</td>
</tr>
</tbody>
</table>

```java
theScript="<script> alert(1)</script>";
var escapedScript=GlideSecurityUtils.escapeScript(theScript);
gs.info(escapedScript);
```

Output: `&lt;script&gt; alert(1)&lt;/script&gt;`

### GlideSecurityUtils - `isURLWhiteListed(String url)`

**Description**

Check the specified URL against the system defined allow list.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>script</td>
<td>String</td>
<td>The script to have escape characters added.</td>
</tr>
</tbody>
</table>

```java
theScript="<script> alert(1)</script>";
var escapedScript=GlideSecurityUtils.escapeScript(theScript);
gs.info(escapedScript);
```

Output: `&lt;script&gt; alert(1)&lt;/script&gt;`
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>The URL to be checked against the URL allow list.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the specified URL is in the allow list.</td>
</tr>
</tbody>
</table>

```java
myURL="http://evil.com/badscript.do";
isWhitelisted=GlideSecurityUtils.isURLWhiteListed(myURL);
gs.info(isWhitelisted);
```

Output: false

### GlideServletRequest - Scoped

The GlideServletRequest API is used in processor scripts.

ServiceNow processors are equivalent to Java servlets. Processors provide a customizable URL endpoint that can execute arbitrary server-side JavaScript code and produce output such as TEXT, JSON, or HTML. The GlideServletRequest API is used in processor scripts to access the HttpServletRequest object. The GlideServletRequest object provides a subset of the HttpServletRequest APIs. The methods are called using the global variable `g_request`.

A useful global variable, `g_target`, is available in processor scripts. It contains the table name extracted from the URL.

The URL to a processor has the format: `https://<instance name.servicenow.com>/<path endpoint>.do?<parameter endpoint>=<value>` where the path endpoint and parameter endpoint are defined on the processor form.

### Scoped GlideServletRequest - getContentType()

Returns the MIME type of the body of the request.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The content type, returns null if the content type is not known.</td>
</tr>
</tbody>
</table>

```javascript
var contentType = g_request.getContentType();
```

**Scoped GlideServletRequest - getHeader(String name)**

Returns the header value.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the header to be retrieved.</td>
</tr>
</tbody>
</table>

```javascript
var headerValue = g_request.getHeader("host");
```

Output:

demonightlyus.service-now.com

**Scoped GlideServletRequest - getHeaderNames()**

Returns a comma-separated list of header names.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A comma-separated list of header names.</td>
</tr>
</tbody>
</table>
var headerList = g_request.getHeaderNames();

Output:

host, connection, cache-control, authorization, accept, user-agent, accept-encoding, accept-language,
cookie, x-forwarded-proto, x-forwarded-host, x-forwarded-for

Scoped GlideServletRequest - getHeaders(String name)

Returns the header values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Names of the headers to be retrieved.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The header values.</td>
</tr>
</tbody>
</table>

var headerValue = g_request.getHeaders("host");

Output:

demonightlyus.service-now.com

Scoped GlideServletRequest - getParameter(String name)

Returns the value of the parameter contained in the request URL.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the parameter to be retrieved. This can be the parameter endpoint from the processor form.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strings</td>
<td>The parameter value. Returns null if the parameter is not found.</td>
</tr>
</tbody>
</table>
var name = g_request.getParameter("x_snc_custom_x_snc_name");

**Scoped GlideServletRequest - getParameterNames()**

Returns an enumeration list of URL parameters that were used in the request URI.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Java enumerator object list of URL parameters used in the HTTP request URI.</td>
</tr>
</tbody>
</table>

var paramList = g_request.getParameterNames();

**Scoped GlideServletRequest - getQueryString()**

Returns the query string from the request.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The query string.</td>
</tr>
</tbody>
</table>

The code snippet is:

```javascript
var daString = g_request.getQueryString();
g_processor.writeOutput("The query string is: "+ daString);
```

Output: The request URL is: https://demonightlycloudedge.service-now.com/x_custom_app_customApp.do?x_custom_app_name=George&bell=rung
The query string is: x_custom_app_name=George&bell-rung

**GlideServletResponse - Scoped**

The **ScopedGlideServletResponse** API is used in processor scripts.

ServiceNow processors are equivalent to Java servlets. Processors provide a customizable URL endpoint that can execute arbitrary server-side JavaScript code and produce output such as TEXT, JSON, or HTML. The **ScopedGlideServletResponse** API is used in processor scripts to access the **HttpServletResponse** object. The **ScopedGlideServletResponse** object provides a subset of the **HttpServletResponse** APIs. The methods are called using the global variable `g_response`.

A useful global variable, `g_target`, is available in processor scripts. It contains the table name extracted from the URL.

The URL to a processor has the format: `https://<instance name.servicenow.com>/<path endpoint>.do?<parameter endpoint>=<value>` where the path endpoint and parameter endpoint are defined on the processor form.

**Scoped GlideServletResponse - sendRedirect(String location)**

Sends a temporary redirect to the client.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>location</td>
<td>String</td>
<td>The URL to receive the response.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Scoped GlideServletResponse - setContentType(String type)**

Sets the MIME type of the response

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>The MIME type.</td>
</tr>
</tbody>
</table>
void g_response.setContentType('text/html;charset=UTF-8');

Scoped GlideServletResponse - setHeader(String key, String value)
Sets a response header to the specified value.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Specifies the header.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The value to be assigned to the header. If the header exists, it is overwritten.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

void g_response.setHeader("host", "instance.service-now.com");

Scoped GlideServletResponse - setStatus(Number status)
Sets the status code for the response.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>Number</td>
<td>The status to be set.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
// set the status to okay
g_response.setStatus(200);

**GlideSession - Global**

The GlideSession API allows you to find information about the current session.

### GlideSession - clearClientData(String paramName)

Clears a session client value previously set with `putClientData()`.

This method is used in a client script to clear data values that were set by a server script using the `putClientData()` method.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramName</td>
<td>String</td>
<td>Name of the client data to clear.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var session = gs.getSession();
session.putClientData('custName', 'Harry');
var clientData = session.getClientData('custName');
gs.info(clientData);

session.clearClientData('custName');
clientData = session.getClientData('custName');
gs.info(clientData);
```

**Output:**

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Harry</td>
</tr>
<tr>
<td>null</td>
</tr>
</tbody>
</table>

### GlideSession - getClientData(String paramName)

Returns a session client value previously set with `putClientData()`.
This method is used in a client script to retrieve data values that were set by a server script that used the `putClientData()` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramName</td>
<td>String</td>
<td>Name of the client data to retrieve.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The client data as a string.</td>
</tr>
</tbody>
</table>

```javascript
var session = gs.getSession();
session.putClientData('test1', 'Harry');
var clientData = session.getClientData('test1');
gs.info(clientData);
```

**Output:**

```
Harry
```

### Scoped equivalent

To use the `getClientData()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSession - getClientData(String paramName)**.

### GlideSession - getLanguage()

Gets the session's language code.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The session's language code.</td>
</tr>
</tbody>
</table>
var session = gs.getSession();
var language = session.getLanguage();
gs.info(language);

Output:

en

Scoped equivalent

To use the `getLanguage()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSession - getLanguage()`.

**GlideSession - getRoles()**

Gets a list of roles for the current user.

The list of roles does not reflect any changes made during the current user session. To get the updated list of roles, the user must log out and log back in.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

`gs.info(gs.getSession().getRoles());`

Output:

admin,hr_fulfiller,itsa_fulfiller,security_admin

**GlideSession - getTimeZoneName()**

Gets the name of the session's time zone.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the session’s time zone.</td>
</tr>
</tbody>
</table>

```javascript
var session = gs.getSession();
var zoneName = session.getTimeZoneName();
gs.info(zoneName);
```

Output:

US/Pacific

Scoped equivalent

To use the `getTimeZoneName()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSession - getTimeZoneName()`.

GlideSession - isInteractive()

Determines if the current session is interactive.

An interactive session is one that involves an end-user interacting with a user interface that then retrieves information from a server. An example of this type of session is when a user logs in using the log-in screen or uses a form to query a data store. A non-interactive session is one that only involves programmatic interaction with a server such as a SOAP request to retrieve data.
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the session is interactive.</td>
</tr>
</tbody>
</table>

⚠️ **Note:** The `isInteractive()` method will always return a false value for all user sessions using the classic or current mobile applications.

```javascript
var interActive = gs.getSession().isInteractive();
gs.info(interActive);
```

Output:

false

**Scoped equivalent**

To use the `isInteractive()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSession - isInteractive()`.

### GlideSession - isLoggedIn()

Determines if the current user is currently logged in.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the current user is logged in.</td>
</tr>
</tbody>
</table>

```javascript
var session = gs.getSession();
var loggedIn = session isLoggedIn();
gs.info(loggedIn);
```

Output:

true
Scoped equivalent

To use the isLoggedIn() method in a scoped application, use the corresponding scoped method: Scoped GlideSession - isLoggedIn().

GlideSession - putClientData(String paramName, String paramValue)

Sets a session client value that can be retrieved with getClientData(). This method is used in a server side script that runs when a form is created.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramName</td>
<td>String</td>
<td>Name of the client parameter to set.</td>
</tr>
<tr>
<td>paramValue</td>
<td>String</td>
<td>Parameter value.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var session = gs.getSession();
session.putClientData('test1', 'Harry');
var clientData = session.getClientData('test1');
gs.info(clientData);
```

Output:

Harry

Scoped equivalent

To use the putClientData() method in a scoped application, use the corresponding scoped method: Scoped GlideSession - putClientData(String paramName, String paramValue).

GlideSession - Scoped

The scoped GlideSession API provides a way to find information about the current session.

There are no constructors for creating an instance of a scoped GlideSession object. Instead, use the getSession() method of the scoped GlideSystem API.
Scoped GlideSession - getClientData(String paramName)
Retrieves a session client value previously set with putClientData().
Use this method to retrieve data values that were set using the putClientData() method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramName</td>
<td>String</td>
<td>Name of the client data to retrieve.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The client data as a string.</td>
</tr>
</tbody>
</table>

```javascript
var session = gs.getSession();
session.putClientData('test1', 'Harry');
var clientData = session.getClientData('test1');
gs.info(clientData);
```

Output:

Harry

Scoped GlideSession - getClientIP()
Returns the client IP address.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The IP address.</td>
</tr>
</tbody>
</table>
var session = gs.getSession();
var addr = session.getClientIP();
gs.info(addr);

Output:
50.59.164.97

Scoped GlideSession - getCurrentApplicationId()

Returns the application currently selected in the application picker.

This method requires admin privileges.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The currently selected application.</td>
</tr>
</tbody>
</table>

var session = gs.getSession();
var appID = session.getCurrentApplicationId();
gs.info(appID);

Output:
ce05b9f32b840200c5244f74b4da1501

Scoped GlideSession - getCurrentDomainID()

Returns the sys_id of the current domain for the logged-in user session.

The identifier that is returned depends on the domain type and the instantiation of that domain.

- If the user is configured in the global domain, and does not use the domain picker to switch domains, the method returns null.
- If the user uses the domain picker to switch to the global domain, the method returns the string "global".
- For all other domains, the method returns the sys_id of that domain.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the session domain of the current logged-in user. This is the same information that appears in the domain picker.</td>
</tr>
</tbody>
</table>

This example shows the current sys_email record's domain being set to the user's session domain, if the user domain is global or null.

```javascript
// Set domain based on parent record's domain
setDomain();

// If the domain is global, set to user's domain
if (current.sys_domain == 'global' || current.sys_domain.nil())
    current.sys_domain = gs.getSession().getCurrentDomainID();

function setDomain()
{
    if (current.target_table.nil())
        return;

    var d = new GlideRecord(current.target_table);
    if (!d.isValid())
        return;

    if (!d.get('sys_id', current.instance))
        return;

    if (typeof(d.sys_domain) == 'object')
        current.sys_domain = d.sys_domain;

    Scoped GlideSession - getLanguage()
Returns the session's language code.
```
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The session's language code.</td>
</tr>
</tbody>
</table>

```javascript
var session = gs.getSession();
var language = session.getLanguage();
gs.info(language);
```

Output:
```
en
```

Scoped GlideSession - getSessionToken()

Returns the session token.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The session token.</td>
</tr>
</tbody>
</table>

```javascript
var session = gs.getSession();
var token = session.getSessionToken();
gs.info(token);
```

Output:
```
4284b5372b840200c5244f74b4da15f2c3476cf7fcb6572afa4ef9d5e6d307a5fd9e1da7
```
Scoped GlideSession - getTimeZoneName()

Returns the name of the session's time zone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the session's time zone.</td>
</tr>
</tbody>
</table>

```js
var session = gs.getSession();
var zoneName = session.getTimeZoneName();
gs.info(zoneName);
```

Output:

US/Pacific

 Scoped GlideSession - getUrlOnStack()

Returns the URL on the stack. Returns null if the stack is empty.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The URL. Returns null if the stack is empty.</td>
</tr>
</tbody>
</table>

```js
var session = gs.getSession();
var URL = session.getUrlOnStack();
gs.info(URL);
```

Output: line breaks added for clarity.
Scoped GlideSession - isImpersonating()

Returns true if the user is impersonating another user.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the user is impersonating another user; otherwise, returns false.</td>
</tr>
</tbody>
</table>

```javascript
var isImpersonator = gs.getSession().isImpersonating();
gs.info(isImpersonator);
```

Scoped GlideSession - isInteractive()

Returns true if the session is interactive.

An interactive session is one that involves an end-user interacting with a user interface that then retrieves information from a server. An example of this type of session is when a user logs in using the log-in screen or uses a form to query a data store. A non-interactive session is one that only involves programmatic interaction with a server such as a SOAP request to retrieve data.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the session is interactive.</td>
</tr>
</tbody>
</table>
var interActive = gs.getSession().isInteractive();
gs.info(interActive);

Scoped GlideSession - isLoggedIn()
Returns true if the user is logged in.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the user is logged in.</td>
</tr>
</tbody>
</table>

var session = gs.getSession();
var loggedIn = session.isLoggedIn();
gs.info(loggedIn);

Output:

true

Scoped GlideSession - putClientData(String paramName, String paramValue)
Sets a session client value that can be retrieved with getClientData(). This method is used in a server side script that runs when a form is created.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramName</td>
<td>String</td>
<td>Name of the client data to set.</td>
</tr>
<tr>
<td>paramValue</td>
<td>String</td>
<td>Value of the client data.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var session = gs.getSession();
session.putClientData('test1', 'Harry');
var clientData = session.getClientData('test1');
gs.info(clientData);

Output:
Harry

**GlideSPScriptable - Scoped**

Interact with data and perform record operations in Service Portal widgets. You access GlideSPScriptable methods by using the global `$sp` object.

 Scoped GlideSPScriptable - `canReadRecord(GlideRecord now_GR)`

Returns true if the user can read the specified GlideRecord.

If the record type is `kb_knowledge`, `sc_cat_item`, or `sc_category`, the method checks if the user can view the item.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>The GlideRecord to check.</td>
</tr>
</tbody>
</table>

Server script

```javascript
//Server script
data.items = [];
data.userName = gs.getUserDisplayName();
var now_GR = new GlideRecord("sc_cat_item");
now_GR.query();
while(now_GR.next() && data.items.length < 10) {
    if ($sp.canReadRecord(now_GR)) {
        data.items.push(now_GR.getDisplayValue("name"));
    }
}
```
//HTML template
<div class="panel panel-default">
    <div class="panel-heading">Hi, {{c.data.userName}}!</div>
    <div class="panel-body">
        Here are some things you can order:
        <ul><li ng-repeat="item in c.data.items">{{item}}</li></ul>
    </div>
</div>

Scoped GlideSPScriptable - canReadRecord(String table, String sysId)

Returns true if the user can read the specified GlideRecord.

If the record type is kb_knowledge, sc_cat_item, or sc_category, the method checks if the user can view the item.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>Name of the table to query.</td>
</tr>
<tr>
<td>sysId</td>
<td>String</td>
<td>Sys_id of the record to query.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the record is valid and readable.</td>
</tr>
</tbody>
</table>

Scoped GlideSPScriptable - canSeePage(String pageID)

Returns true if the currently logged in user has permission to view the specified page.

The system determines permission using roles and user criteria. For more information, see Configure page security by role and User criteria for Service Portal.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pageID</td>
<td>String</td>
<td>Page ID from the Pages [sp_page] table.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the currently logged in user has permissions to view the specified page. Values include:</td>
</tr>
<tr>
<td></td>
<td>• true: The user can view the page.</td>
</tr>
<tr>
<td></td>
<td>• false: Either the user cannot view the page, the given pageID is not valid, or the user is not logged in.</td>
</tr>
</tbody>
</table>

This example adds a condition to the Knowledge menu item in the SP Header Menu to only show the menu item to users who have permission to view the kb_view2 page.

GlideSPScriptable.canSeePage("kb_view2")

Scoped GlideSPScriptable - `getCatalogItem(String sysId)`

Returns a model and view model for a `sc_cat_item` or `sc_cat_item_guide`.

This method is deprecated. Use the `getCatalogItem(String sysId, Boolean isOrdering)` method instead. This method calls the `getCatalogItem(String sysId, Boolean isOrdering)` method with the `isOrdering` parameter set to false, which means that write roles security checking is done.

This method is a quick way to get the data necessary to render and order a catalog item using `<sp-model />`. If you just need to get a catalog item to show its picture or name, use GlideRecord to query the `sc_cat_item` table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysId</td>
<td>String</td>
<td>The sys_id of the catalog item (<code>sc_cat_item</code>) or order guide (<code>sc_cat_item_guide</code>).</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object containing the catalog item variable model, view, sections, pricing, and client scripts.</td>
</tr>
</tbody>
</table>
// Server script
(function() {
    var sys_id = $sp.getParameter("sys_id")
    data.catItem = $sp.getCatalogItem(sys_id);
})();

// Client script
function($http, spUtil) {
    var c = this;
    var submitting = false;
    c.getIt = function() {
        if (submitting) return;
        $http.post(spUtil.getURL('sc_cat_item'), c.data.catItem).success(function(response)
        {
            if (response.answer) {
                c.req = response.answer;
                c.req.page = c.req.table == 'sc_request' ? 'sc_request' : 'ticket';
            }
        });
    }
}

//SCSS
.img-bg {
    padding: 5px;
    background-color: $brand-primary;
}
.img-responsive {
    margin: 0 auto;
}
.cat-icon {
    display: block;
    margin: -40px auto 0;
}

// HTML template
<div class="col-sm-4">
    <div class="panel panel-default">
        <div class="img-bg">
            <img ng-src="{{::data.catItem.picture}}" class="img-responsive" />
            </div>
    </div>
</div>
**Scoped GlideSPScriptable - getCatalogItem(String sysId, Boolean isOrdering)**

Returns a model and view model for a `sc_cat_item` or `sc_cat_item_guide`.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysId</td>
<td>String</td>
<td>The sys_id of the catalog item (<code>sc_cat_item</code>) or order guide (<code>sc_cat_item_guide</code>).</td>
</tr>
</tbody>
</table>
| isOrdering | Boolean | When true, uses create roles security check. When false, uses write roles security check.  
When users are ordering an item or have it in their cart, check using the create roles.  
If users are not ordering, for example, somebody is looking at a requested item to see the variables associated with that item, then check using the write roles. |
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object containing the catalog item variable model, view, sections, pricing, and client scripts.</td>
</tr>
</tbody>
</table>

**Scoped GlideSPScriptable - getDisplayValue(String fieldName)**

Returns the display value of the specified field (if it exists and has a value) from either the widget's sp_instance or the sp_portal record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The display value from either the sp_instance or sp_portal record.</td>
</tr>
</tbody>
</table>

```javascript
//Server script
(function() {
    data.title = $sp.getDisplayValue("title");
    data.catalog = $sp.getDisplayValue("sc_catalog");
})();

//HTML template
<div>
    <h1>sp_instance.title: {{::data.title}}</h1>
    <h1>sp_portal.sc_catalog: {{::data.catalog}}</h1>
</div>
```

**Scoped GlideSPScriptable - getField(GlideRecord now_GR, String fieldName)**

Returns information about the specified field in the specified GlideRecord.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>The GlideRecord to check</td>
</tr>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The field to find information for</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object containing the field’s label, value, <code>displayValue</code>, and type. Returns null if the GlideRecord of field name are not valid, or if the field is not readable.</td>
</tr>
</tbody>
</table>

**Scoped GlideSPScriptable - `getFields(GlideRecord now_GR, String fieldNames)`**

Checks the specified list of field names, and returns an array of valid field names.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>The GlideRecord to check</td>
</tr>
<tr>
<td>fieldNames</td>
<td>String</td>
<td>A comma separated list of field names.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of valid fields.</td>
</tr>
</tbody>
</table>

**Scoped GlideSPScriptable - `getFieldsObject(GlideRecord now_GR, String fieldNames)`**

Checks the specified list of field names and returns an object of valid field names.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>The GlideRecord to check</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>A comma separated list of field names.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object containing valid field names.</td>
</tr>
</tbody>
</table>

Scoped GlideSPScriptable - getForm(String tableName, String sysId)

Return the form.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>The name of the table</td>
</tr>
<tr>
<td>sysId</td>
<td>String</td>
<td>The form’s sys_id</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The form</td>
</tr>
</tbody>
</table>

Scoped GlideSPScriptable - getKBCategoryArticles(String sys_id, Number limit)

Returns KB articles in the specified category and its subcategories.

To avoid performance issues, do not use this method to return articles in large categories or articles with inline images. Instead, use getKBArticleSummaries().

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysId</td>
<td>String</td>
<td>Sys_id of the KB article category.</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td>Maximum number of KB articles returned.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>The articles within the category and its subcategories with:</td>
</tr>
<tr>
<td></td>
<td>• A workflow_state of published.</td>
</tr>
<tr>
<td></td>
<td>• A valid_to date greater than or equal to the current date.</td>
</tr>
</tbody>
</table>

//Server script
(function() {
    data.kbs = $sp.getKBCategoryArticles("0ac1bf8bff0221009b20ffffffec", 5);
})();

//HTML template
<div>
articles: {{::data.kbs}}
</div>

Scoped GlideSPScriptable - getKBCategoryArticleSummaries(String sys_id, Number limit, Number maxChars)

Returns Knowledge Base article summaries in the specified category and its subcategories.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>Sys_id of the KB article category.</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td>Maximum number of KB articles returned.</td>
</tr>
<tr>
<td>maxChars</td>
<td>Number</td>
<td>Maximum number of characters to return from the article text. For full article text, set the value to -1.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>The articles within the category and its subcategories with:</td>
</tr>
<tr>
<td></td>
<td>• A workflow_state of published.</td>
</tr>
<tr>
<td></td>
<td>• A valid_to date greater than or equal to the current date.</td>
</tr>
</tbody>
</table>
// Server script
(function() {
    data.summary = $sp.getKBCategoryArticleSummaries("0ac1bf8bff0221009b20ff0fffffffec", 5, 200);
})();

// HTML template
<div>
    articles: {{::data.summary}}
</div>

Scoped GlideSPScriptable - getKBCount(String sys_id)

Returns the number of articles in the defined Knowledge Base.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>Sys_id of a Knowledge Base record.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of knowledge articles in the defined Knowledge Base with:</td>
</tr>
<tr>
<td></td>
<td>• A workflow_state of published.</td>
</tr>
<tr>
<td></td>
<td>• A valid_to date greater than or equal to the current date.</td>
</tr>
</tbody>
</table>

// Server script
(function() {
    data.count = $sp.getKBCount("a7e8a78bff0221009b20fffffff17");
})();

// HTML template
<div>
    articles: {{::data.count}}
</div>

Scoped GlideSPScriptable - getListColumns(String tableName, String view)

Returns a list of the specified table's columns in the specified view.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table</td>
</tr>
<tr>
<td>view</td>
<td>String</td>
<td>The view by which to filter the columns</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object containing the column names.</td>
</tr>
</tbody>
</table>

Scoped GlideSPScriptable - getMenuItems(String sysId)

Returns an array of menu items for the specified instance.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysId</td>
<td>String</td>
<td>sysId of the instance</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Menu items for the specified instance</td>
</tr>
</tbody>
</table>

Scoped GlideSPScriptable - getMenuHREF(GlideRecord page)

Returns the (?id=) portion of the URL based on the sp_menu type.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>page</td>
<td>GlideRecord</td>
<td>The page</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The href portion of the URL.</td>
</tr>
</tbody>
</table>
Scoped GlideSPScriptable - getParameter(String name)

Returns the value of the specified parameter.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the key from the query string or post body.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Returns the specified parameter as an object. Returns null if there is no request, JSON request, or widget.</td>
</tr>
</tbody>
</table>

Scoped GlideSPScriptable - getPortalRecord()

Returns the portal record from the Service Portals [sp_portal] table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The portal record for the current portal from the Service Portals [sp_portal] table.</td>
</tr>
</tbody>
</table>

```
//Server script
(function() {
    var portalGr = $sp.getPortalRecord();
    data.logo = portalGr.getDisplayValue("logo");
    data.homepage = portalGr.getDisplayValue("homepage.id");
})();

//HTML template
<div>
    <img ng-src="{{c.data.logo}}" />
```
Scoped GlideSPScriptable - getRecord(String table, String sys_id)

If parameters are provided, returns the GlideRecord identified by the provided table and Sys ID. If no parameters are provided, returns the record identified by the current URL.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>Optional. The table of the record to return. If no parameters are included, returns the table and Sys ID identified by the current URL.</td>
</tr>
<tr>
<td>sys_id</td>
<td>String</td>
<td>Optional. The Sys ID of the record to return. If no parameters are included, returns the record identified by the current URL.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>If parameters are provided, returns the record identified by the provided table and Sys ID.</td>
</tr>
<tr>
<td></td>
<td>If no parameters are provided, returns the record identified by the current URL.</td>
</tr>
<tr>
<td></td>
<td>Returns null if the widget is embedded by another widget, or if the record for the provided parameters is not found.</td>
</tr>
</tbody>
</table>

//Server script
(function(){
  var now_GR = $sp.getRecord();
data.tableLabel = now_GR.getLabel();
})();

//HTML template
<div class="panel-heading">
  <h4 class="panel-title">${{{data.tableLabel}} details}</h4>
</div>
Scoped GlideSPScriptable - `getRecordDisplayValues(Object data, GlideRecord from, String names)`

Copies display values for the specified fields into the data parameter.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Object</td>
<td>The display values for the specified fields are copied to this object.</td>
</tr>
<tr>
<td>from</td>
<td>GlideRecord</td>
<td>The GlideRecord to process.</td>
</tr>
<tr>
<td>names</td>
<td>String</td>
<td>A comma-separated list of field names.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Scoped GlideSPScriptable - `getRecordElements(Object data, GlideRecord from, String names)`

For the specified fields, copies the element's name, display value, and value into the data parameter.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Object</td>
<td>The element's name, display value, and value for the specified fields are copied to this object.</td>
</tr>
<tr>
<td>from</td>
<td>GlideRecord</td>
<td>The GlideRecord to process.</td>
</tr>
<tr>
<td>names</td>
<td>String</td>
<td>A comma-separated list of field names.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
Scoped GlideSPScriptable - getRecordValues(Object data, GlideRecord from, String names)

Copies values for the specified field names from the GlideRecord into the data parameter.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Object</td>
<td>The value for the specified fields are copied to this object.</td>
</tr>
<tr>
<td>from</td>
<td>GlideRecord</td>
<td>The GlideRecord to process.</td>
</tr>
<tr>
<td>names</td>
<td>String</td>
<td>A comma-separated list of field names.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Scoped GlideSPScriptable - getRecordVariables(GlideRecord now_GR, Boolean includeNilResponses)

Returns Service Catalog variables associated with a record in String format.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>The record to retrieve Service Catalog variables for. Must be a record with Service Catalog variables defined, such as a requested item [sc_req_item] record or an incident submitted through a record producer.</td>
</tr>
<tr>
<td>includeNilResponses</td>
<td>Boolean</td>
<td>Optional. If true, the API includes variables with no user-defined value.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Service Catalog variables associated with the record.</td>
</tr>
</tbody>
</table>
//Server script
(function() {
    var itemsGR = new GlideRecord("sc_req_item");
    itemsGR.get('585d1bc44f4f13008a959a211310c77d');

    data.scVars = $sp.getRecordVariables(itemsGR);

})();

//HTML template
<div>
    Requested item variables: {{::data.scVars}}
</div>

Output:

Requested item variables: [
    {
        "display_value":"MetroPCS",
        "name":"carrier",
        "visible_summary":true,
        "id":"585d1bc44f4f13008a959a211310c77c",
        "label":"Allocated carrier",
        "type":"5",
        "value":"metropcs",
        "visible_standalone":true,
        "visible_guide":true
    },
    {
        "display_value":"Unlimited",
        "name":"data_plan",
        "visible_summary":true,
        "id":"d05d1bc44f4f13008a959a211310c77c",
        "label":"Monthly data allowance",
        "type":"5",
        "value":"unlimited",
        "visible_standalone":true,
        "visible_guide":true
    },
    {
        "display_value":"12 Months",
        "name":"duration",
        "visible_summary":true,
        "id":"d85d1bc44f4f13008a959a211310c77c",
        "label":null,
        "type":null,
        "value":null,
        "visible_standalone":true,
        "visible_guide":true
    }
]
Scoped GlideSPScriptable - getRecordVariablesArray(GlideRecord now_GR, Boolean includeNilResponses)

Returns an array of Service Catalog variables associated with a record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>The record to retrieve Service Catalog variables for. Must be a record with Service Catalog variables defined, such as a requested item [sc_req_item] record or an incident submitted through a record producer.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>includeNilResponses</td>
<td>Boolean</td>
<td>Optional. If true, the API includes variables with no user-defined value.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Array of Service Catalog variables associated with the record.</td>
</tr>
</tbody>
</table>

//Server script
(function() {
    var itemsGR = new GlideRecord("sc_req_item");
    itemsGR.get('585d1bc44f4f13008a959a211310c77d');

    data.scVars = $sp.getRecordVariablesArray(itemsGR);

});

//HTML template
<div>
    Requested item variables: {{::data.scVars}}
</div>

Output:

Requested item variables: [
    {
        "display_value": "MetroPCS",
        "name": "carrier",
        "visible_summary": true,
        "id": "585d1bc44f4f13008a959a211310c77c",
        "label": "Allocated carrier",
        "type": "5",
        "value": "metropcs",
        "visible_standalone": true,
        "visible_guide": true
    },
    {
        "display_value": "Unlimited",
        "name": "data_plan",
        "visible_summary": true,
    },
]
Scoped GlideSPScriptable - getStream(String table, String sysId)

Gets the activity stream for the specified record. This method works on tables that extend the task table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>The table name</td>
</tr>
<tr>
<td>sysId</td>
<td>String</td>
<td>The sys_id of the record</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>If a table extending the task table is specified, contains the display_value, sys_id, short_description,number, entries, user_sys_id, user_full_name, user_login, label, table, and journal_fields properties; otherwise contains the table and sys_id properties.</td>
</tr>
</tbody>
</table>

ℹ️ Note: The user_login property contains the User ID of the current user. The user_sys_id and user_full_name properties reference the creator of the queried record.

Scoped GlideSPScriptable - getUserInitials()

Returns the user's initials.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The user's initials</td>
</tr>
</tbody>
</table>

Scoped GlideSPScriptable - getValue(String name)

Returns the named value of the JSON request, instance, or portal.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the JSON request, instance, or portal.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Value of the specified parameter. Null if the specified entity does not exist or has no such parameter.</td>
</tr>
</tbody>
</table>

//Server script
(function() {
    data.title = $sp.getValue("title");
    data.catalog = $sp.getValue("sc_catalog");
})();

//HTML template
<div>
    <h1>sp_instance.title: {{::data.title}}</h1>
    <h1>sp_portal.sc_catalog: {{::data.catalog}}</h1>
</div>

Scoped GlideSPScriptable - `getValue(Object data, String names)`
Copies values from the request or instance to the data parameter.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Object</td>
<td>Receives the parameter values.</td>
</tr>
<tr>
<td>names</td>
<td>String</td>
<td>Comma-separated string of field names.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
Scoped GlideSPScriptable - getVariablesArray(Boolean includeNilResponses)

Returns an array of Service Catalog variables associated with the record in the URL.

For example, if the URL includes the parameters
id=form&table=sc_req_item&sys_id=832e9620db4f330083766b984b9619cf, the API returns the variables associated with the given record in the Requested item [sc_req_item] table. Must be a record with Service Catalog variables defined, such as a requested item [sc_req_item] record or an incident submitted through a record producer.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>includeNilResponses</td>
<td>Boolean</td>
<td>Optional. If true, the API includes variables with no user-defined value.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Array of variables associated with the table.</td>
</tr>
</tbody>
</table>

//Server script
(function() {
  data.scVars = $sp.getVariablesArray();
})();

//HTML template
<div>
  Current record variables: {{::data.scVars}}
</div>

Output:

Current record variables: [
  {
    "display_value":"unlimited",
    "name":"data","visible_summary":true,"id":"472e5620db4f330083766b984b96198a",
    "label":"Data",
    "type":"6",
  }
]
Scoped GlideSPScriptable - `getWidget(String sysID, Object options)`

Gets a widget by id or sys_id, executes that widget's server script using the provided options, then returns the widget model.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysID</td>
<td>String</td>
<td>The widget sys_id or widget_id</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>An object to pass to the widget's server script. Refer to this object as <code>options</code> in your server script.</td>
</tr>
</tbody>
</table>

**Note:** Any options passed into this function will only be available in the embedded widget's server script on the **first execution** of that script. Any subsequent calls into the server script from the embedded widget will not contain the object properties passed in.

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>A widget model to be used with sp-widget.</td>
</tr>
</tbody>
</table>
//Server script
data.myWidget = $sp.getWidget('widget_id', {p1: param1, p2: param2});

//HTML
<sp-widget widget="c.data.myWidget"></sp-widget>

Scoped GlideSPScriptable - mapUrlToSPUrl(String url)
Transforms a URL requesting a list or form in the platform UI into the URL of the corresponding id=list or id=form Service Portal page.

Use this method to perform tasks such as redirecting a user after login to the correct Service Portal page when they request a platform UI form or list URL. Note that the id=list and id=form page targets are not customizable.

Note: Table, sys_id, and sysparm_query values are preserved from the original URL; sysparm_view is not.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>Platform UI URL</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Transformed Service Portal URL. If the passed-in URL does not request a list or a form in the platform UI, a null value is returned.</td>
</tr>
</tbody>
</table>

List example

```java
GlideSPScriptable().mapUrlToSPUrl("http://demo.service-now.com/task_list.do?sysparm_userpref_module=1523b8d4c611227b00be8216ec331b9a&sysparm_query=assigned_to=javascript:getMyAssignments()&sysparm_clear_stack=true")
```

Returns

```
id=list&table=task&filter=assigned_to=javascript:getMyAssignments()&sys_id=&v=
```

Form example

```java
GlideSPScriptable().mapUrlToSPUrl("incident.do?sys_id=12bc12bc12bc12bc12bc12bc12bc")
```
Returns

```plaintext
id=form&table=incident&filter=&sys_id=12bc12bc12bc12bc12bc12bc12bc12bc&v=
```

Scoped GlideSPScriptable - `userCanSeeSearchSourceByCriteria(String searchSourceID)`

Returns true if the current user can see a search source specified by the `searchSourceID`.

Returns true if the user is not restricted by user criteria. For more detail on how user criteria affects Service Portal assets, see [User criteria for Service Portal](#).

ℹ️ **Note:** This method does not evaluate role-based permissions of the search source, and should only be used if User Criteria is enabled.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>searchSourceID</code></td>
<td>String</td>
<td>Sys_id of the search source record.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if current user has read access to the search source.</td>
</tr>
</tbody>
</table>

Scoped GlideSPScriptable - `isUserCriteriaEnabled()`

Returns true if user criteria is enabled.

This method returns true if the Service Portal User Criteria Support plugin (com.glide.service-portal.user-criteria) is active and the Enable use of User Criteria records instead of Roles fields for Service Portal entitlements (glide.service_portal.user_criteria_enabled) system property is set to true.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if user criteria is enabled.</td>
</tr>
</tbody>
</table>

**GlideSPSearchAnalytics - Global**

Generates search analytics from custom ServiceNow search widgets.

By default, the Search Events [sys_search_event] and Search Source Events [sys_search_source_event] tables collect search data from base system search widgets. However, custom search widgets do not benefit from this feature. Use this API in the server script of custom search widgets in global environments to send search data to the search events tables. Generate relevant search suggestions for your users and monitor search analytics to understand what your users are searching for and whether they’re finding what they need.

This API collects the following search data:

- Searched terms
- Rank of clicked items
- Filters used to refine search results
- Table names and record identifiers from the first page of search results
- Search results clicked, including browser and location

Search Suggestions is a Now Platform feature. For more information, see Search Suggestions.

**GlideSPSearchAnalytics - publish(String payload)**

Sends search data to the Search Event [sys_search_event], Search Event per source [sys_search_source_event], and Search Signal Event [sys_search_signal_event] tables for search results, or to the Search Result Clicked [sys_search_result_clicked] and Search Signal Result Event [sys_search_signal_result_event] tables when a user clicks a search result.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>payload</td>
<td>String</td>
<td>JSON payload in String format containing the search data to send to the Search Event [sys_search_event], Search Event per source [sys_search_source_event], and Search Signal</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event [sys_search_signal_event] tables for search results, or to the Search Result Clicked [sys_search_result_clicked] and Search Signal Result Event [sys_search_signal_result_event] tables when a user clicks a search result. Structure the payload as key-value pairs according to the type of search data you would like to collect.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Collect data when the user searches for a term:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>◦ query: String. The query being searched.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>◦ results_per_source: Array. Number of results returned from each search source.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>number_of_results: Number. Number of results returned from the query for the specific search source.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ais_profile: String. Sys_id of the search profile applied to the search. Only applicable for searches made with the AI Search engine.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>search_results: Array. Results returned in the first page of search results.</td>
<td></td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record_id</td>
<td>String</td>
<td>Sys_id of the search result record.</td>
</tr>
<tr>
<td>table_name</td>
<td>String</td>
<td>Name of the table in which the search result record was found.</td>
</tr>
<tr>
<td>refinement_occurred</td>
<td>Boolean</td>
<td>True if the user filtered the search results; otherwise false.</td>
</tr>
</tbody>
</table>

- Collect data when a user refines their search results:
  - query: String. The query being searched.
  - refinement_occurred: Boolean. True if the user filtered the search results; otherwise false.

- Collect the rank of the search result that the user selected.
  - query: String. The query being searched.
  - click_rank: Number. Rank of the item the user selected. For example, if the user clicked the third item in a list of search results, the value is 3.

- Collect information on the search result that the user clicks. Data logs to the Search Result Clicked [sys_search_result_clicked] table.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>String</td>
<td>The query being searched.</td>
</tr>
<tr>
<td>page_id</td>
<td>String</td>
<td>Sys_id of the page record from the Pages [sp_page] table.</td>
</tr>
<tr>
<td>results_per_source</td>
<td>Array</td>
<td>Number of results returned from each search source.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>source_id</td>
<td>String</td>
<td>Sys_id of the search source record from the Search Sources [sp_search_source] table.</td>
</tr>
<tr>
<td>number_of_results</td>
<td>Number</td>
<td>Number of results returned from the query for the specific search source.</td>
</tr>
<tr>
<td>refinement_occurred</td>
<td>Boolean</td>
<td>True if the user filtered the search results; otherwise false.</td>
</tr>
<tr>
<td>click_rank</td>
<td>Number</td>
<td>Rank of the item the user selected. For example, if the user clicked the third item in a list of search results, the value is 3.</td>
</tr>
<tr>
<td>browser_info</td>
<td>String</td>
<td>Logs browser name, version, and platform.</td>
</tr>
<tr>
<td>location</td>
<td>Float</td>
<td>Logs latitude and longitude coordinates of the user location.</td>
</tr>
<tr>
<td>result_clicked_sys_id</td>
<td>String</td>
<td>Sys_id of the result the user selected.</td>
</tr>
<tr>
<td>label_description</td>
<td>String</td>
<td>Description of the result that the user selected.</td>
</tr>
<tr>
<td>source_table</td>
<td>String</td>
<td>Name of the source table containing the selected result.</td>
</tr>
<tr>
<td>ais_doc_id</td>
<td>String</td>
<td>Underscore-separated name of the source table and sys_id of the result record for each result the user selected. For example, for the result &quot;results&quot; where the user clicked the first result, the value is &quot;results/resultssysid&quot;.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>example, if the user clicked a result from the Knowledge [kb_knowledge] table, the value is of the form kb_knowledge_3020c9b14843210e9db4b5b08b9a712d. Only applicable for searches made with the AI Search engine.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Collect data when a user searches for a term and collect the rank of the search result that the user clicked:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>◦ query: String. The query being searched.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>◦ results_per_source: Array. Number of results returned from each search source.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>number_of_results: Number. Number of results returned from the query for the specific search source.</td>
<td></td>
</tr>
<tr>
<td>◦ ais_profile: String. Sys_id of the search profile applied to the search. Only applicable for searches made with the AI Search engine.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>◦ search_results: Array. Summary of records returned in the first page of search results.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>record_id: String. Sys_id of the search result record.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>table_name: String. Name of the table in which the search result record was found.</td>
<td></td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>click_rank</td>
<td>Number</td>
<td>Rank of the item the user selected. For example, if the user clicked the third item in a list of search results, the value is 3.</td>
</tr>
<tr>
<td>ais_doc_id</td>
<td>String</td>
<td>Underscore-separated name of the source table and sys_id of the result record for each result the user selected. For example, if the user clicked a result from the Knowledge [kb_knowledge] table, the value is of the form kb_knowledge_3020c9b14843210e9db4b5b08b9a712d. Only applicable for searches made with the AI Search engine.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

This example passes direct search data to the search analytics tables.

```javascript
var searchObject = new GlideSPSearchAnalytics();
var payload = {
    'query': 'how do i get a new computer',
    'portal_id': '81b75d3147032100ba13a5554ee4902b',
    'page_id': '87466b63c3223100c8b837659bba8feb',
    'results_per_source': [{
        'source_id': 'bf2c4c15b3a31300e64be12b86a8dcb7',
        'number_of_results': 2,
    }, {
        'source_id': 'c96eb1686721220023c82e08f585efff',
        'number_of_results': 1
    }],
    'refinement': false,
    'search_results': [{
        'record_id': 'KB Article 1',
        'table_name': 'knowledge'
    }, {  
```
This example dynamically collects direct search data from a custom search widget and passes it to the search analytics tables.

```javascript
var directSearchData = searchObject.publish(JSON.stringify(payload));
```

//client script
var payload = {
  query: this.data.query,
  portal_id: this.data.portalID,
  page_id: this.data.pageID,
  results_per_source: getSearchSources(),
  refinement_occurred: false,
};
var payloadObject = {
  action: "TestAction",
  payload: payload,
};
c.server.get(payloadObject);

//server script
if (input && input.action === "TestAction") {
  input.action = "";
  var textSearchAnalytics = new GlideSPSearchAnalytics().publish(JSON.stringify(input.payload));
  return;
}
```

This example collects direct search-results-clicked data from a custom search widget and passes it to the search analytics [sys_search_result_clicked] table.

```javascript
function setUserLocationCoords(cb) {
  var onSuccess = function(pos) {
    return cb({
      latitude: pos.coords.latitude,
      longitude: pos.coords.longitude,
    });
  };
}
```
var onError = function() {
    return cb({
        latitude: null,
        longitude: null,
    });
};
return window.navigator.geolocation.getCurrentPosition(onSuccess, onError, {
    enableHighAccuracy: true,
});

function trackSearchResultClicked(resultClicked, rank) {
    if (!rank || rank < 1) return;
    var c = $scope;
    var query = _.get(c.data, "q");
    var results = _.get(c.data, "results", []);
    var result = results.find(function(e) {
        return e.primary.trim() === resultClicked.trim();
    });
    var description = result.name != null ? result.name : result.primary;
    var sourceTable = result.table != null ? result.table : null;
    var payloadObject = {
        action: "TestAction",
        payload: {
            query: query,
            portal_id: c.portal.sys_id,
            page_id: $scope.page && $scope.page.id,
            results_per_source: getSearchSources(c),
            refinement_occurred: false,
            click_rank: rank,
            browser_info: $window.navigator.userAgent,
            location: {},
            result_clicked_sys_id: result.sys_id,
            label_description: description,
            source_table: sourceTable,
        },
    };
    setUserLocationCoords(function(coords) {
        payloadObject.payload.location.latitude = coords.latitude;
        payloadObject.payload.location.longitude = coords.longitude;
        $window.spSearchAnalytics = {
            query: query,
            refinement_occurred: false,
        };
    });
c.server.get(payloadObject);
});
}
// Event delegation for Updating click rank
$(".panel").on("click", ".result-item a", function(e) {
var index = $(e.currentTarget).parent().parent().data("index");
var resultClicked = e.currentTarget.innerText;
trackSearchResultClicked(resultClicked, index + 1);
});
// server script
if (input && input.action === "TestAction") {
input.action = "";
var textSearchAnalytics = new
GlideSPSearchAnalytics().publish(JSON.stringify(input.payload));
return;
}

GlideStringUtil - Scoped, Global
Provides string handling methods.
Access these methods using the static object GlideStringUtil. This class is
available in scoped and global scripts.
GlideStringUtil - dotToUnderBar(String sourceString)
Replaces periods with underscore characters.
Parameters
Name

Type

Description

sourceString

String

Text to process.

Returns
Type

Description

String

Text with periods replaced with underscores.

var filename="../../../../../../etc/passwd";
cleanFilename=GlideStringUtil.dotToUnderBar(filename);
gs.info(cleanFilename);

2656

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


Output:

```
__/__/__/__/__/__/etc/passwd
```

**GlideStringUtil - escapeAllQuotes(String sourceString)**
Removes quotes from a string.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceString</td>
<td>String</td>
<td>The string to be processed.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The string with quotes removed.</td>
</tr>
</tbody>
</table>

```java
mystring="let's escape some quotes";
escapeQuote=GlideStringUtil.escapeAllQuotes(mystring);
gs.info(escapeQuote);
```

Output:

```
lets escape some quotes
```

**GlideStringUtil - escapeForHomePage(String sourceString)**
Replaces problem characters with escape characters.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceString</td>
<td>String</td>
<td>Text to process.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Text with problem characters replaced with escape characters.</td>
</tr>
</tbody>
</table>
mystring="<test> string \n to escape";
escapedString=GlideStringUtil.escapeForHomePage(mystring);
gs.info(escapedString);

Output:
%3ctest%3e string \n to escape

**GlideStringUtil - escapeHTML(String htmlString)**
Replaces illegal characters with their escape codes.

Using this method removes illegal characters that might cause the UI to render improperly, or trigger a client side attack such as JavaScript or HTML injection.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>htmlString</td>
<td>String</td>
<td>Text to process.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Text with illegal characters replaced with their escape codes.</td>
</tr>
</tbody>
</table>

mydata='"<>\';
mydata=GlideStringUtil.escapeHTML(mydata);
gs.info(mydata);

Output:
"&quot;&amp;&lt;&gt;"

**GlideStringUtil - escapeNonPrintable(String sourceString)**
Replaces non-printable characters with their printable notation.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceString</td>
<td>String</td>
<td>Text to process.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>Text with non-printable characters replaced with printable notation.</td>
<td></td>
</tr>
</tbody>
</table>

```java
mystring="test \x09 non \x00 printable \x07 chars";
escapedString=GlideStringUtil.escapeNonPrintable(mystring);
gs.info(escapedString);
```

**Output:**

test 	 non   printable  chars

### GlideStringUtil - escapeQueryTermSeparator(String sourceString)
Replaces query term separators “^” with their escape sequence “^^”.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceString</td>
<td>String</td>
<td>Text to process.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Text with query term separators replaced with the escape characters.</td>
</tr>
</tbody>
</table>

```java
myquery="test^Test";
escapedQuery=GlideStringUtil.escapeQueryTermSeparator(myquery);
gs.info(escapedQuery);
```

**Output:**

test^^Test

### GlideStringUtil - escapeTicks(String sourceString)
Replaces quotes with escape characters by adding a backslash before each quote.
### GlideStringUtil - `getHTMLValue(String sourceString)`

**Description**

Replaces illegal HTML characters into HTML notation.

Using this method removes illegal characters that might cause the UI to render improperly, or trigger a client side attack such as JavaScript or HTML injection.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceString</td>
<td>String</td>
<td>Text to process.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Text with illegal characters replaced with HTML notation.</td>
</tr>
</tbody>
</table>

```java
define (string) mydata='&';
glideStringUtil.getHTMLValue(mydata);
gs.info(htmlvalue);
```

**Output:**

```
&amp;
```

© 2021 ServiceNow, Inc. All rights reserved.  
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.  
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**GlideStringUtil - getNumeric(String sourceString)**

Extracts numeric characters from a string.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceString</td>
<td>String</td>
<td>Text to process.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Text containing only numeric characters.</td>
</tr>
</tbody>
</table>

```java
mystring='123 test 456 String 789 cleaning';
onlyNumeric=GlideStringUtil.getNumeric(mystring);
gs.info(onlyNumeric);
```

**Output:**

```
123456789
```

**GlideStringUtil - isBase64(String sourceString)**

Validates whether the specified string is a valid base64 string.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceString</td>
<td>String</td>
<td>Text to process.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates whether the specified string is in valid base64 format. Possible values:  
  - true: Valid base64 formatted string.  
  - false: Invalid base64 formatted string. |
// (adding a *** to corrupt the base64 format)
base64="GethdTYehtsetB***";
isValid=GlideStringUtil.isBase64(base64);
gs.info(isValid);

Output:
false

GlideStringUtil - isEligibleSysID(String sourceString)
Validates whether the specified string is in valid sys_id format.

The sys_id format is a sequence of 32 hexadecimal characters where all the characters are in the range [0-9, a-f, A-F].

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceString</td>
<td>String</td>
<td>Text to process.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the specified string is in valid sys_id format. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Valid sys_id formatted string.</td>
</tr>
<tr>
<td></td>
<td>• false: Invalid sys_id formatted string.</td>
</tr>
</tbody>
</table>

sysID="62826bf03710200044e0bfc8bcbe5df1";
isElig=GlideStringUtil.isEligibleSysID(sysID);
gs.info(isElig);

Output:
true

GlideStringUtil - newLinesToBreaks(String sourceString)
Replaces the new line character, /n, with a break code, <br/>.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceString</td>
<td>String</td>
<td>Text to process.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Text with new line characters replaced with HTML break code.</td>
</tr>
</tbody>
</table>

mystring="new line break \n, this is after the break";
replaceNewLine=GlideStringUtil.newLinesToBreaks(mystring);
gs.info(replaceNewLine);

Output:
new line break <br/>, this is after the break

GlideStringUtil - normalizeWhitespace(String sourceString)
Replaces carriage returns, line feeds, and tabs with spaces, and then removes leading, trailing, and duplicate spaces.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceString</td>
<td>String</td>
<td>Text to process.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Text with carriage returns, line feeds, and tabs replaced with spaces, and then leading, trailing, and duplicate spaces removed.</td>
</tr>
</tbody>
</table>

mystring="test with \n (new line) and \t (tabulation)";
normalizedString=GlideStringUtil.normalizeWhitespace(mystring);
gs.info(normalizedString);

Output:
test with (new line) and (tabulation)
GlideStringUtil - unEscapeHTML(String htmlString)

Replaces escape characters with their respective character.

This method replaces these escape characters: &lt; &gt; &nbsp; &amp; &quot;.

⚠️ Note: In scoped applications call this method as unescapeHTML(String). In global applications call this method as unEscapeHTML(String).

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>htmlString</td>
<td>String</td>
<td>String to process.</td>
</tr>
</tbody>
</table>

This code example shows the method being called in a global application.

```java
mydata="&quot;&lt;&gt;&amp;';
unescaped=GlideStringUtil.unEscapeHTML(mydata);
gs.info(unescaped);
```

Output:

"<>"&

This code example shows the method being called in a scoped application.

```java
mydata="&quot;&lt;&gt;&amp;';
unescaped=GlideStringUtil.unescapeHTML(mydata);
gs.info(unescaped);
```

Output:

"<>"&

GlideSysAttachment - Scoped

The GlideSysAttachment API provides a way to handle attachments. Content is returned as a string, not as a byte array when `getcontent()` is called.
Content is returned as a GlideScriptableInputStream object when `getContentStream()` is called. The GlideScriptableInputStream contains the actual bytes not converted into a String.

**Scoped GlideSysAttachment - copy(String sourceTable, String sourceID, String targetTable, String targetID)**

Copies attachments from the source record to the target record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceTable</td>
<td>String</td>
<td>Name of the table with the attachments to be copied.</td>
</tr>
<tr>
<td>sourceID</td>
<td>String</td>
<td>Source table's sys_id.</td>
</tr>
<tr>
<td>targetTable</td>
<td>String</td>
<td>Name of the table on which to add the attachments.</td>
</tr>
<tr>
<td>targetID</td>
<td>String</td>
<td>Target table's sys_id.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Array of sys_ids of the attachments that were copied.</td>
</tr>
</tbody>
</table>

```glide
var attachment = new GlideSysAttachment();
var incidentSysID = 'ab1b30031b04ec101363ff37dc4bcfc';
var incGR = new GlideRecord('incident');
incGR.get(incidentSysID);
var copiedAttachments = attachment.copy('incident', incidentSysID, 'problem',
    incGR.getValue('problem_id'));
gs.info('Copied attachments: ' + copiedAttachments);
```

**Output**

```
Copied attachments: 6e4621df1bc420501363ff37dc4bcba2,a87769531b0820501363ff37dc4bcba2
```

**Scoped GlideSysAttachment - deleteAttachment(String attachmentID)**

Deletes the specified attachment.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attachmentID</td>
<td>String</td>
<td>Attachment's sys_id.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var attachment = new GlideSysAttachment();
var attachmentSysID = 'a87769531b0820501363ff37dc4bcba2';
attachment.deleteAttachment(attachmentSysID);
```

**Scoped GlideSysAttachment - getAttachments(String tableName, String sys_id)**

Returns a GlideRecord containing the matching attachment metadata such as name, type, or size.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table to which the attachment belongs; for example, incident.</td>
</tr>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The sys_id of record to which the attachment belongs.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>GlideRecord object containing the matching attachment metadata such as name, type, or size.</td>
</tr>
</tbody>
</table>

The following script lists attachment file names for a record with two attachments.

```javascript
var attachment = new GlideSysAttachment();

var agr = attachment.getAttachments('<table_name>', '<record_sys_id>');
```
while(agr.next())
gs.info(agr.getValue('file_name'));

Output:

*** Script: filename1.txt
*** Script: filename2.txt

Scoped GlideSysAttachment - getContent(GlideRecord sysAttachment)
Returns the attachment content as a string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysAttachment</td>
<td>GlideRecord</td>
<td>Attachment record.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Attachment contents as a string. Returns up to 5MB of data.</td>
</tr>
</tbody>
</table>

var attachment = new GlideSysAttachment();
var incidentSysID = 'ab1b30031b04ec101363ff37dc4bcfbc';

var agr = attachment.getAttachments('incident', incidentSysID);

if (agr.next()) {
    var attachmentContent = attachment.getContent(agr);
    gs.info('Attachment content: ' + attachmentContent);
}

Output

Attachment content: I am text in a txt file attached to a record.

Scoped GlideSysAttachment - getContentBase64(GlideRecord sysAttachment)
Returns the attachment content as a string with base64 encoding.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysAttachment</td>
<td>GlideRecord</td>
<td>Attachment record.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Attachment contents as a string with base64 encoding. Returns up to 5MB of data.</td>
</tr>
</tbody>
</table>

```javascript
var attachment = new GlideSysAttachment();
var incidentSysID = 'ab1b30031b04ec101363ff37dc4bcbfc';

var agr = attachment.getAttachments('incident', incidentSysID);

if (agr.next()) {
    var attachmentContent = attachment.getContentBase64(agr);
    gs.info('Attachment content base64 encoded: ' + attachmentContent);
}
```

Output

```
Attachment content base64 encoded: SSBhbSB0ZXh0Lg==
```

**Scoped GlideSysAttachment - getContentStream(String sysID)**

Returns a GlideScriptableInputStream object given the sys_id of an attachment.

You can use the GlideTextReader API to read the content stream.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysID</td>
<td>String</td>
<td>Attachment sys_id.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideScriptableInputStream</td>
<td>Stream that contains the attachment content.</td>
</tr>
</tbody>
</table>
```
var attachment = new GlideSysAttachment();
var attachmentSysID = '6e4621df1bc420501363ff37dc4bcb0';
var attachmentContentStream = attachment.getContentStream(attachmentSysID);
gs.info('Attachment content stream: ' + attachmentContentStream);
```

**Output**
```
Attachment content stream: com.glide.communications.GlideScriptableInputStream@14bd299
```

**Scoped GlideSysAttachment - GlideSysAttachment()**

Creates an instance of the GlideSysAttachment class.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Scoped GlideSysAttachment - write(GlideRecord record, String fileName, String contentType, String content)**

Attaches a specified attachment to the specified record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>GlideRecord</td>
<td>Record to which to attach the attachment.</td>
</tr>
<tr>
<td>fileName</td>
<td>String</td>
<td>Attachment file name.</td>
</tr>
<tr>
<td>contentType</td>
<td>String</td>
<td>Attachment content type.</td>
</tr>
<tr>
<td>content</td>
<td>String</td>
<td>Attachment content.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Attachment sys_id. Returns null if the attachment was not added.</td>
</tr>
</tbody>
</table>
rec.get('78271e1347c12200e0ef563dbb9a7109');
var fileName = 'example.txt';
var contentType = 'text/csv';
var content = 'The text that is stored inside my file';

var agr = attachment.write(rec, fileName, contentType, content);
gs.info('The attachment sys_id is: ' + agr);

Output:
The attachment sys_id is: 01271e4317c13311e0ef563dbb9abe34

Scoped GlideSysAttachment - writeBase64(GlideRecord now_GR, String fileName, String contentType, String content_base64Encoded)
Inserts an attachment for the specified record using base64 encoded content.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>Record to which the attachment is to be attached.</td>
</tr>
<tr>
<td>fileName</td>
<td>String</td>
<td>Attachment's file name.</td>
</tr>
<tr>
<td>contentType</td>
<td>String</td>
<td>Attachment's content type.</td>
</tr>
<tr>
<td>content</td>
<td>String</td>
<td>Attachment content in base64 format.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the attachment created.</td>
</tr>
</tbody>
</table>

var attachment = new GlideSysAttachment();

var rec = new GlideRecord('incident');
var incidentSysID = 'ab1b30031b04ec101363ff37dc4bcfbc';
rec.get(incidentSysID);
var fileName = 'example.txt';
var contentType = 'text/csv';
var base64Encodedcontent = 'SSBhbSB0ZXh0Lg==';
var agr = attachment.writeBase64(rec, fileName, contentType, base64Encodedcontent);

gs.info('The attachment sys_id is: ' + agr);

Output

The attachment sys_id is: 10cde9971b0820501363ff37dc4bcba6

**Scoped GlideSysAttachment - writeContentStream(GlideRecord now_GR, String fileName, String contentType, GlideScriptableInputStream inputStream)**

Inserts an attachment using the input stream.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>Record to which the attachment is to be attached.</td>
</tr>
<tr>
<td>fileName</td>
<td>String</td>
<td>Attachment's file name.</td>
</tr>
<tr>
<td>contentType</td>
<td>String</td>
<td>Attachment's content type.</td>
</tr>
<tr>
<td>content</td>
<td>GlideScriptableInputStream</td>
<td>Attachment content.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the attachment created.</td>
</tr>
</tbody>
</table>

Attaches a content stream from the sys_attachment table to a test_table record.

```javascript
function copyAttachmentToGlideRecord(conceptSysId) {

  // Get record from test_table using sys_id
  var targetGlideRecord = new GlideRecord("test_table");
  if (!targetGlideRecord.get(conceptSysId)) {
    throw ("Cannot find record created by test with sys_id: " + conceptSysId);
  }

  // Get record from sys_attachment table
  var sourceAttachmentGlideRecord = new GlideRecord('sys_attachment');
  sourceAttachmentGlideRecord.query();
  sourceAttachmentGlideRecord.next();
}
```
// Get field values from retrieved sys_attachment record
var fileName = sourceAttachmentGlideRecord.getValue('file_name');
var contentType = sourceAttachmentGlideRecord.getValue('content_type');
var sourceAttachmentSysId = sourceAttachmentGlideRecord.getValue('sys_id');

// Attach sys_attachment record content stream to test_table record
var gsa = new GlideSysAttachment();
gsa.writeContentStream(
    targetGlideRecord,
    fileName,
    contentType,
    gsa.getContentStream(sourceAttachmentSysId));
gs.info("Attachment created");
}

**GlideSysListControl - Scoped**

The scoped GlideSysListControl class allows you to determine if the New or Edit buttons are displayed.

**GlideSysListControl - getControlID()**

Returns the sys_id for the control.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

```javascript
var sysListCtrl = new GlideSysListControl("incident");
var controlID = sysListCtrl.getControlID();
gs.info(controlID);
```

Output:

```
91a807a60a0a3c74012113e28b47ca2e
```
**GlideSysListControl - GlideSysListControl(String tableName)**

Instantiates a GlideSysListControl object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table</td>
</tr>
</tbody>
</table>

**GlideSysListControl - isOmitEditButton()**

Returns true if the edit button is not displayed.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True when the edit button is not displayed.</td>
</tr>
</tbody>
</table>

```javascript
var sysListCtrl = new GlideSysListControl("incident");
var isOmitted = sysListCtrl.isOmitEditButton();
gs.info(isOmitted);
```

Output:

false

**GlideSysListControl - isOmitNewButton()**

Returns true when the New button is not displayed.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## GlideSystem - Global

The GlideSystem API, which is referred to by the variable name `gs` in any server-side JavaScript, provides a number of convenient methods to get information about the system, the current logged in user, and date/time information.

Many of the GlideSystem methods facilitate the easy inclusion of dates in query ranges, and are most often used in filters and reporting.

### GlideSystem - addErrorMessage(Object message)

Adds an error message for the current session.

Use `getErrorMessages()` to retrieve a list of error messages currently being shown.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Object</td>
<td>The message to add.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var sysListCtrl = new GlideSysListControl("incident");
var isOmitted = sysListCtrl.isOmitNewButton();
gs.info(isOmitted);
```

Output:

false
var user_password = request.getParameter("user_password");
var min_len = 8;
var rules = "Password must be at least " + min_len + 
  " characters long and contain a digit, an uppercase letter, and a lowercase letter.";
if (user_password.length() < min_len) {
  gs.addErrorMessage("TOO SHORT: " + rules);
  return false;
}
var digit_pattern = new RegExp("[0-9]", "g");
if (!digit_pattern.test(user_password)) {
  gs.addErrorMessage("DIGIT MISSING: " + rules);
  return false;
}
var upper_pattern = new RegExp("[A-Z]", "g");
if (!upper_pattern.test(user_password)) {
  gs.addErrorMessage("UPPERCASE MISSING: " + rules);
  return false;
}
var lower_pattern = new RegExp("[a-z]", "g");
if (!lower_pattern.test(user_password)) {
  gs.addErrorMessage("LOWERCASE MISSING: " + rules);
  return false;
}
return true; // password is OK
}  

Scoped equivalent

To use the addErrorMessage() method in a scoped application, use the corresponding scoped method: Scoped GlideSystem - addErrorMessage(String message).

GlideSystem - addInfoMessage(Object message)

Adds an info message for the current session.

Use getInfoMessages() to retrieve the list of info messages being shown. This method is not supported for asynchronous business rules and cannot be used within transform scripts.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Object</td>
<td>The message to add.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
if ((!current.u_date1.nil()) && (!current.u_date2.nil())) {
    var start = current.u_date1.getGlideObject().getNumericValue();
    var end = current.u_date2.getGlideObject().getNumericValue();
    if (start > end) {
        gs.addInfoMessage('start must be before end');
        current.u_date1.setError('start must be before end');
        current.setAbortAction(true);
    }
}
```

**Scoped equivalent**

To use the `addInfoMessage()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - addInfoMessage(String message)**.

**GlideSystem - addMessage(String type, Object message)**

Adds a message for the current session.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>Type of message, such as error or info.</td>
</tr>
<tr>
<td>message</td>
<td>Object</td>
<td>Message to add to the current session.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
GlideSystem - beginningOfLastMonth()

Gets the date and time for the beginning of last month in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of last month, in the format yyyy-mm-dd hh:mm:ss.</td>
</tr>
</tbody>
</table>

This example sets the date and time of the GlideDateTime object to the beginning of last month.

```javascript
var date = new GlideDate();
date.setValue(gs.beginningOfLastMonth());
var dateasint = date.toString().replace('-','');
gs.print(dateasint);
```

Scoped equivalent

To use the `beginningOfLastMonth()` method in a scoped application, use the corresponding scoped method: Scoped GlideSystem - beginningOfLastMonth().

GlideSystem - beginningOfLastWeek()

Returns the date and time for the beginning of last week in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of last week.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

This example sets the value of the current Glide date/time record to the beginning of last week.

```javascript
var gdt2 = new GlideDateTime(dt);
gdt2.setValue(gs.beginningOfLastWeek());
```

Scoped equivalent

To use the `beginningOfLastWeek()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - beginningOfLastWeek()`.

**GlideSystem - beginningOfNextWeek()**

Returns the date and time for the beginning of next week in GMT.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of next week.</td>
</tr>
<tr>
<td></td>
<td>Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

This example sets the value of the current Glide date/time record to the beginning of next week.

```javascript
var gdt2 = new GlideDateTime(dt);
gdt2.setValue(gs.beginningOfNextWeek());
```

Scoped equivalent

To use the `beginningOfNextWeek()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - beginningOfNextWeek()`.
GlideSystem - beginningOfNextMonth()

Returns the date and time for the beginning of next month in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of next month.</td>
</tr>
<tr>
<td></td>
<td>Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

This example sets the value of the current Glide date/time record to the beginning of next month.

```javascript
var gdt2 = new GlideDateTime(dt);
gdt2.setValue(gs.beginningOfNextMonth());
```

Scoped equivalent

To use the `beginningOfNextMonth()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - beginningOfNextMonth()`.

GlideSystem - beginningOfNextYear()

Returns the date and time for the beginning of next year in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The GMT beginning of next year, in the format yyyy-mm-dd hh:mm:ss.</td>
</tr>
</tbody>
</table>
This example sets the value of the current Glide date/time record to the beginning of next year.

```java
var gdt2 = new GlideDateTime(dt);
gdt2.setValue(gs.beginningOfNextYear());
```

**Scoped equivalent**

To use the `beginningOfNextYear()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - beginningOfNextYear()`.

### GlideSystem - beginningOfThisMonth()

Returns the date and time for the beginning of the current month in GMT.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of the current month.</td>
</tr>
<tr>
<td></td>
<td>Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

This example sets the value of the current Glide date/time record to the beginning of this month.

```java
var gdt2 = new GlideDateTime(dt);
gdt2.setValue(gs.beginningOfThisMonth());
```

**Scoped equivalent**

To use the `beginningOfThisMonth()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - beginningOfThisMonth()`.

### GlideSystem - beginningOfThisQuarter()

Returns the date and time for the beginning of the current quarter in GMT.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of the current quarter.</td>
</tr>
<tr>
<td></td>
<td>Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

This example sets the value of the current Glide date/time record to the beginning of the current quarter.

```javascript
var gdt2 = new GlideDateTime(dt);
gdt2.setValue(gs.beginningOfThisQuarter());
```

Scoped equivalent

To use the `beginningOfThisQuarter()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - beginningOfThisQuarter()`.

**GlideSystem - beginningOfThisWeek()**

Returns the date and time for the beginning of this week in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of the current week.</td>
</tr>
<tr>
<td></td>
<td>Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

This example sets the value of the current Glide date/time record to the beginning of the current week.
var gdt2 = new GlideDateTime(dt);
gdt2.setValue(gs.beginningOfThisWeek());

**Scoped equivalent**

To use the `beginningOfThisWeek()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - beginningOfThisWeek()`.

**GlideSystem - beginningOfThisYear()**

Returns the date and time for the beginning of this year in GMT.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of the current year. Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

This example sets the value of the current Glide date/time record to the beginning of the current year.

```javascript
var gdt2 = new GlideDateTime(dt);
gdt2.setValue(gs.beginningOfThisYear());
```

**Scoped equivalent**

To use the `beginningOfThisYear()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - beginningOfThisYear()`.

**GlideSystem - beginningOfToday()**

Retrieves the date and time for the beginning of today in GMT.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of the current day.</td>
</tr>
<tr>
<td></td>
<td>Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

This example sets the value of the current Glide date/time record to the beginning of the current day.

```javascript
var gdt2 = new GlideDateTime(dt);
gdt2.setValue(gs.beginningOfToday());
```

GlideSystem - beginningOfTomorrow()

Retrieves the (UTC) beginning of tomorrow adjusted for the timezone of the current session.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of tomorrow.</td>
</tr>
<tr>
<td></td>
<td>Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

```javascript
var today = new GlideDateTime(gs.beginningOfTomorrow()).getNumericValue();
```

GlideSystem - beginningOfYesterday()

Retrieves the date and time for the beginning of yesterday in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The GMT beginning of yesterday, in the format yyyy-mm-dd hh:mm:ss.</td>
</tr>
</tbody>
</table>

```javascript
var start = new GlideDateTime();
start.setValue(gs.beginningOfYesterday());
```

**GlideSystem - calDateDiff(String startDate, String endDate, Boolean numericValue)**

Calculate the difference between two dates using the default calendar.

Calendars are now legacy. If Schedules are being used, see the topic Calculate Duration Given a Schedule.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>String</td>
<td>Starting date to compare in the current user’s date format.</td>
</tr>
<tr>
<td>endDate</td>
<td>String</td>
<td>Ending date to compare in the current user’s date format.</td>
</tr>
<tr>
<td>numericValue</td>
<td>Boolean</td>
<td>Flag that indicates the format of the returned time value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Return value is formatted in number of seconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Return value is formatted ddd hh:mm:ss.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>If the numericValue parameter is true, returns the difference between the two dates as an integer number of seconds. If false, returns the difference between the two dates in the format ddd hh:mm:ss.</td>
</tr>
</tbody>
</table>
```
var endDateTime = gs.nowDateTime();
gs.print('--- Total records: ' + countRecordsTotal);
gs.print('--- End time: ' + endDateTime);
gs.print('Time diff: ' + gs.calDateDiff(startDateTime, endDateTime));
```

### GlideSystem - dateDiff(String startDate, String endDate, Boolean numericValue)

Calculates the difference between two dates.

This method expects the earlier date as the first parameter and the later date as the second parameter; otherwise, the method returns the difference as a negative value. Use `getDisplayValue()` to convert the strings to the expected format.

This method expects parameters in the user/system date time format, which may not be the same as the internal format. Using parameters in formats other than the user/system date time format may return invalid results.

If you are working with GlideDateTime objects use the GlideDateTime `subtract()` method instead of `dateDiff()`.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>String</td>
<td>The starting date to compare in the current user's date format.</td>
</tr>
<tr>
<td>endDate</td>
<td>String</td>
<td>The ending date to compare in the current user's date format.</td>
</tr>
<tr>
<td>numericValue</td>
<td>Boolean</td>
<td>If true, the return value will be formatted in number of seconds; if false, returns the difference between the two dates in the format ddd hh:mm:ss.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>If the numericValue parameter is true, returns the difference between the two dates as an integer number of seconds; if false, returns the difference between the two dates in the format ddd hh:mm:ss.</td>
</tr>
</tbody>
</table>

```java
// Given two date/times as DateTime objects
// Set the values this way to ensure a consistent input time
var datel = new GlideDateTime();
```
var date2 = new GlideDateTime();
date1.setDisplayValueInternal('2014-01-01 12:00:00');
date2.setDisplayValueInternal('2014-01-01 13:00:00');

// Determine the difference as number of seconds (returns a string)
// Use getDisplayValue() to convert the string to the format expected by dateDiff()
var diffSeconds = gs.dateDiff(date1.getDisplayValue(), date2.getDisplayValue(), true);

// JavaScript will coerce diffSeconds from a string to a number
// since diffSeconds is being compared to a number
var msg = (diffSeconds <= 0) ? ' is on or after ' : ' is before ';
gs.print(date1.getDisplayValue() + msg + date2.getDisplayValue());

GlideSystem - dateGenerate(String date, String range)
Generates a date and time for the specified date in GMT.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>String</td>
<td>Date to generate in GMT. Format: yyyy-mm-dd</td>
</tr>
<tr>
<td>range</td>
<td>String</td>
<td>Start, end, or a time. Format: 24-hour hh:mm:ss</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Date and time in the format yyyy-mm-dd hh:mm:ss. If range is start, the return value is yyyy-mm-dd 00:00:00. If range is end, the return value is yyyy-mm-dd 23:59:59.</td>
</tr>
</tbody>
</table>

This example shows using dateGenerate() to set the start date when querying records in the Incident table.

var tableData = new GlideRecord('incident');
tableData.addEncodedQuery("sys_created_onBETWEENjavascript:gs.dateGenerate('2015-10-07','00:00:00')@javascript:gs.daysAgoEnd(0)^priority=1^severityIN1,2");
tableData.query();
gs.addInfoMessage("Count: " +tableData.getRowCount());
Scoped equivalent

To use the `dateGenerate()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - dateGenerate(String date, String range).**

**GlideSystem - daysAgo(Number days)**

Returns a date and time for a certain number of days ago.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>days</td>
<td>Number</td>
<td>Number of days</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT of the specified number of days ago. Format: yyyy-mm-dd hh:mm:ss.</td>
</tr>
</tbody>
</table>

```javascript
function contractNoticeDue() {
    var now_GR = new GlideRecord("contract");
    now_GR.addQuery("u_contract_status", "Active");
    now_GR.query();
    while (now_GR.next()) {
        if ((now_GR.u_termination_date <= gs.daysAgo(-90)) && (now_GR.u_contract_duration == "Long")) {
            now_GR.u_contract_status = "In review";
        } else if ((now_GR.u_termination_date <= gs.daysAgo(-50)) && (now_GR.u_contract_duration == "Medium")) {
            now_GR.u_contract_status = "In review";
        } else if ((now_GR.u_termination_date <= gs.daysAgo(-10)) && (now_GR.u_contract_duration == "Short")) {
            now_GR.u_contract_status = "In review";
        }
    }
    now_GR.update();
}
```
Scoped equivalent
To use the `daysAgo()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - daysAgo(Number days)`.

GlideSystem - `daysAgoEnd(Number days)`
Returns a date and time for the end of the day a specified number of days ago.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>days</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

This example shows using `daysAgoEnd()` to set the end date when querying records in the Incident table.

```javascript
var tableData = new GlideRecord('incident');
tableData.addEncodedQuery("sys_created_on BETWEEN javascript:gs.dateGenerate('2015-10-07','00:00:00')@javascript:gs.daysAgoEnd(0)" +"priority=1" +"severity IN 1,2");
tableData.query();
gs.addInfoMessage("Count: " +tableData.getRowCount());
```

Scoped equivalent
To use the `daysAgoEnd()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - daysAgoEnd(Number days)`.

GlideSystem - `daysAgoLocal(Number days)`
Returns the date and time of the beginning of the day for the specified number of days ago. The returned date and time reflect the time zone of the current session (local time).

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>days</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Local date and time for the beginning of the day in the user-defined date time format. If the date time format is not modified from its initial value the format is yyyymm-dd hh:mm:ss.</td>
</tr>
</tbody>
</table>

```java
if (due_in == "1 Day") {
    dd = gs.daysAgoLocal(-1);
}
if (due_in == "1 Week") {
    dd = gs.daysAgoLocal(-7);
}
```

GlideSystem - daysAgoStart(Number days)

Returns a date and time for the beginning of the day a specified number of days ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>days</td>
<td>String</td>
<td>Integer number of days</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT start of the day in the format yyyymm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

```java
var now_GR = new GlideRecord('sysapproval_approver');
now_GR.addQuery('state', 'requested');
now_GR.addQuery('sys_updated_on', '<', gs.daysAgoStart(5));
now_GR.query();
```

Scoped equivalent

To use the daysAgoStart() method in a scoped application, use the corresponding scoped method: Scoped GlideSystem - daysAgoStart(Number days).
GlideSystem - endOfLastMonth()

Returns the date and time for the end of last month in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT of the end of last month. Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

This example shows how to query a report record from the last day of last month.

```javascript
var month = gs.endOfLastMonth();
var monthd = new GlideDateTime(month);
monthd.addDays(-1);

var queueLast = new GlideRecord('u_amazon_connect_phone_metrics');
queueMtd.addQuery('u_queue_name',queueNames[i]);
queueMtd.addEncodedQuery('u_time_range=MTD');
queueMtd.addEncodedQuery('u_report_date'+ monthd);
queueMtd.orderByDesc('u_report_date');
queueMtd.query();

if(queueMtd.next()){
    gs.info(queueMtd.u_report_date);
}
```

Scoped equivalent

To use the `endOfLastMonth()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - endOfLastMonth()**.

GlideSystem - endOfLastWeek()

Returns the date and time for the end of last week in GMT.
This example shows how to query records opened on the last week of the previous month.

```javascript
var inc = new GlideRecord('incident');
inc.addQuery('active=true^priority=1^opened_atONLast
month@javascript:gs.beginningOfLastMonth()@javascript:gs.endOfLastMonth()^opened_atONLast
week@javascript:gs.beginningOfLastWeek()@javascript:gs.endOfLastWeek()');
inc.query();
while(inc.next())
{
    gs.addInfoMessage(inc.getRowCount());
}
```

**Scoped equivalent**

To use the `endOfLastWeek()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - endOfLastWeek()**.

**GlideSystem - endOfLastYear()**

Returns the date and time for the end of last year in GMT.
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in format yyyy-mm-dd hh:mm:ss.</td>
</tr>
</tbody>
</table>

This example shows how to query records created for all records created in the last year.

```javascript
var inc = new GlideRecord('incident');
inc.addEncodedQuery('sys_created_onONLast
    year@javascript:gs.beginningOfLastYear()@javascript:gs.endOfLastYear()');
inc.query();
while(inc.next()){
    inc.assignment_group = "d625dccec0a8016700a222a0f7900d06";
    inc.update();
}
```

Scoped equivalent

To use the `endOfLastYear()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - endOfLastYear()**.

**GlideSystem - endOfNextMonth()**

Returns the date and time for the end of next month in GMT.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT data and time.</td>
</tr>
<tr>
<td></td>
<td>Format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

This example shows how to query KB knowledge records created since the beginning of today until the end of next month.

```javascript
var query = "kb_knowledge_base=e81c9a0ddbc15810c38f0763b99619c1^ORkb_knowledge_base=21302e89db055810c3"
```
Scoped equivalent

To use the `endOfNextMonth()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - endOfNextMonth()**.

**GlideSystem - endOfNextWeek()**

Returns the date and time for the end of next week in GMT.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

This example shows how to use `endOfNextWeek()` in a query to obtain all visitors in the past week.
var title = 'Visitors arriving this Week';
var visitorQuery = 'active=true^category=visitor_request^u_my_start_dateONThis
week@javascript:gs.beginningOfThisWeek()@javascript:gs.endOfNextWeek()';

Scoped equivalent
To use the endOfNextWeek() method in a scoped application, use the corresponding scoped method: Scoped GlideSystem - endOfNextWeek().

GlideSystem - endOfNextYear()
Returns the date and time for the end of next year in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT date and time for the end of next year.</td>
</tr>
<tr>
<td></td>
<td>Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

This example shows how to use endOfNextYear() to set the GlideDateTime object to the end of next year.

```javascript
setEndDate : function(dt) {
    var gdt2 = new GlideDateTime(dt);
    gdt2.setValue(gs.endOfNextYear());
}
```

Scoped equivalent
To use the endOfNextYear() method in a scoped application, use the corresponding scoped method: Scoped GlideSystem - endOfNextYear().

GlideSystem - endOfThisMonth()
Returns the date and time for the end of this month in GMT.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String  | GMT date and time for the end of this month.  
|         | Format yyyy-mm-dd hh:mm:ss                                   |

This example shows how to use `endOfThisMonth()` to set the GlideDateTime object to the end of the current month.

```javascript
setEndDate : function(dt) {
    var gdt2 = new GlideDateTime(dt);
    gdt2.setValue(gs.endOfThisMonth());
}
```

Scoped equivalent

To use the `endOfThisMonth()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - endOfThisMonth()`.

GlideSystem - endOfThisQuarter()

Returns the date and time for the end of this quarter in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String  | GMT date and time for the end of this quarter.  
|         | Format: yyyy-mm-dd hh:mm:ss                                   |

This example shows how to use `endOfThisQuarter()` to set the GlideDateTime object to the end of the current month.
setEndDate : function(dt) {
    var gdt2 = new GlideDateTime(dt);
    gdt2.setValue(gs.endOfThisQuarter());
}

Scoped equivalent

To use the endOfThisQuarter() method in a scoped application, use the corresponding scoped method: Scoped GlideSystem - endOfThisQuarter().

GlideSystem - endOfThisWeek()

Returns the date and time for the end of this week in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT date and time for the end of this week. Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

This example shows how to use endOfThisWeek() to set the GlideDateTime object to the end of the current month.

setEndDate : function(dt) {
    var gdt2 = new GlideDateTime(dt);
    gdt2.setValue(gs.endOfThisWeek());
}

Scoped equivalent

To use the endOfThisWeek() method in a scoped application, use the corresponding scoped method: Scoped GlideSystem - endOfThisWeek().

GlideSystem - endOfThisYear()

Returns the date and time for the end of this year in GMT.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT date and time for the end of this year. Format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

This example shows how to use `endOfThisYear()` to set the GlideDateTime object to the end of the current month.

```javascript
setEndDate : function(dt) {
    var gdt2 = new GlideDateTime(dt);
    gdt2.setValue(gs.endOfThisYear());
}
```

### Scoped equivalent

To use the `endOfThisYear()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - endOfThisYear()**.

### GlideSystem - endOfToday()

Retrieves the date and time for the end of today in GMT.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss.</td>
</tr>
</tbody>
</table>

```javascript
var beginToday = gs.beginningOfToday();
var endToday = gs.endOfToday();
var beginTomorrow = gs.beginningOfTomorrow();
```
```javascript
var endTomorrow = gs.endOfTomorrow();
var beginYesterday = gs.beginningOfYesterday();
var endYesterday = gs.endOfYesterday();

gs.info('beginningOfToday: ' + beginToday);
gs.info('endOfToday: ' + endToday);
gs.info('----------');
gs.info('beginningOfTomorrow: ' + beginTomorrow);
gs.info('endOfTomorrow: ' + endTomorrow);
gs.info('beginningOfYesterday: ' + beginYesterday);
gs.info('endOfYesterday: ' + endYesterday);
gs.info('----------');
var gdt1 = new GlideDateTime(beginToday);
var gdt2 = new GlideDateTime(beginTomorrow);
var dur = GlideDateTime.subtract(gdt1, gdt2); //the difference between gdt1 and gdt2

gs.info('BeginTomorrow: ' + dur.getDisplayValue());

var gdt1 = new GlideDateTime(endToday);
var gdt2 = new GlideDateTime(endTomorrow);
var dur = GlideDateTime.subtract(gdt1, gdt2); //the difference between gdt1 and gdt2

gs.info('EndTomorrow: ' + dur.getDisplayValue());

var gdt1 = new GlideDateTime(beginToday);
var gdt2 = new GlideDateTime(beginYesterday);
var dur = GlideDateTime.subtract(gdt2, gdt1); //the difference between gdt1 and gdt2

gs.info('BeginningYesterday: ' + dur.getDisplayValue());

var gdt1 = new GlideDateTime(endToday);
var gdt2 = new GlideDateTime(endYesterday);
var dur = GlideDateTime.subtract(gdt2, gdt1); //the difference between gdt1 and gdt2

gs.info('EndYesterday: ' + dur.getDisplayValue());
```

**GlideSystem - endOfTomorrow()**

Retrieves the date and time for the end of tomorrow in GMT.
### GlideSystem - endOfTomorrow()

Gets the date and time for the end of tomorrow in GMT.

```javascript
var tomorrowEnd = new GlideDateTime();
tomorrowEnd.setValue(gs.endOfTomorrow());
```

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss.</td>
</tr>
</tbody>
</table>

### GlideSystem - endOfYesterday()

Gets the date and time for the end of yesterday in GMT.

```javascript
var yesterdayEnd = new GlideDateTime();
yesterdayEnd.setValue(gs.endOfYesterday());
```

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format (yyyy-mm-dd hh:mm:ss).</td>
</tr>
</tbody>
</table>

### GlideSystem - eventQueue(String name, Object glideRecord, String parm1, String parm2, String queue)

Queues an event for the event manager.

```javascript
GlideSystem.eventQueue(String name, Object glideRecord, String parm1, String parm2, String queue);
```

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the event being queued.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>glideRecord</td>
<td>Object</td>
<td>GlideRecord object, such as &quot;current&quot;.</td>
</tr>
<tr>
<td>parm1</td>
<td>String</td>
<td>(Optional) Saved with the instance if specified.</td>
</tr>
<tr>
<td>parm2</td>
<td>String</td>
<td>(Optional) Saved with the instance if specified.</td>
</tr>
<tr>
<td>queue</td>
<td>String</td>
<td>Name of the queue.</td>
</tr>
</tbody>
</table>

 Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
if (current.operation() != 'insert' && current.comments.changes()) {
    gs.eventQueue("incident.commented", current, gs.getUserID(), gs.getUserName());
}
```

Scoped equivalent

To use the `eventQueue()` method in a scoped application, use the corresponding scoped method: Scoped GlideSystem - eventQueue(String name, Object instance, String parm1, String parm2, String queue).

GlideSystem - eventQueueScheduled(String name, Object glideRecord, String parm1, String parm2, Object expiration)

Queues an event for the event manager at a specified date and time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the event being queued.</td>
</tr>
<tr>
<td>glideRecord</td>
<td>Object</td>
<td>GlideRecord object, such as &quot;current&quot;.</td>
</tr>
<tr>
<td>parm1</td>
<td>String</td>
<td>(Optional) Saved with the instance if specified.</td>
</tr>
<tr>
<td>parm2</td>
<td>String</td>
<td>(Optional) Saved with the instance if specified.</td>
</tr>
<tr>
<td>expiration</td>
<td>Object</td>
<td>Date and time to process this event.</td>
</tr>
</tbody>
</table>
if (current.operation() != 'insert' && current.comments.changes()) {
    gs.eventQueueScheduled("incident.commented", current, gs.getUserID(), gs.getUserName(),
    new GlideDateTime('2018-06-02 20:00:00'));
}

Scoped equivalent

To use the `eventQueueScheduled()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - eventQueueScheduled(String name, Object instance, String parm1, String parm2, Object expiration)`.

GlideSystem - flushMessages()

Clears session messages saved using `addErrorMessage()` or `addInfoMessage()`.

Session messages are shown at the top of the form. In client side scripts, use `g_form.clearMessages()` to remove session messages.

GlideSystem - getAvatar()

Returns the file path to the user's avatar.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The file path to the user's avatar.</td>
</tr>
</tbody>
</table>

```javascript
var avatarFile = gs.getUser().getAvatar();
gs.addInfoMessage('User avatar ID: ' + avatarFile);
```

Output:

```
User avatar ID: c148ed13741310042106710ce41f149.iix?t=small
```

**GlideSystem - getCurrentScopeName()**

Returns the name of the current application scope.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Current scope name.</td>
</tr>
</tbody>
</table>

This example shows how to use `getCurrentScopeName()` to get the scope of the processor.

```javascript
var incident_GR = new GlideRecord('Incident');
if (incident_GR.get("2e3f6baddb9ad600added8f9618cb")) {
    gs.debug("processor scope = " + gs.getCurrentScopeName());
    var w = new global.Workflow();
```
Scoped equivalent

To use the `getCurrentScopeName()` method in a scoped application, use the corresponding scoped method: *Scoped GlideSystem - getCurrentScopeName()*.

**GlideSystem - getDisplayColumn(String tableName)**

Retrieves the display column for the table.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table from which to get the display column name.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Display column name</td>
</tr>
</tbody>
</table>

```javascript
// Return the sys_id value for a given table and its display value
function GetIDValue(table, displayValue) {
    var rec = new GlideRecord(table);
    var dn = gs.getDisplayColumn(table);
    if (rec.get(dn, displayValue))
        return rec.sys_id;
    else
        return null;
}
```

**GlideSystem - getDisplayValueFor(String tableName, String recordID, String fieldName)**

Returns the display value for a specified field on a specified record.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table.</td>
</tr>
<tr>
<td>recordID</td>
<td>String</td>
<td>Sys_id for the record.</td>
</tr>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field whose display value to return.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Display value for the field.</td>
</tr>
</tbody>
</table>

This example shows how to use `getDisplayValueFor()` to return the display value of the `number` field.

```javascript
var value =
  gs.getDisplayValueFor(current.getTableName(), current.getValue('sys_id'), 'number');
```

### GlideSystem - getErrorMessages()

Returns the list of error messages for the session that were added by `addErrorMessage()`.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>List of error messages</td>
</tr>
</tbody>
</table>

This example shows how to parse through and pass back error messages using `getErrorMessages()`.

```javascript
MySessionUtil.getSessionError = function() {
  var msg = null;
  var msgs = gs.getErrorMessages().toArray();
  if (msgs.length > 0) {
```
Scoped equivalent

To use the `getErrorMessages()` method in a scoped application, use the corresponding scoped method: Scoped GlideSystem - `getErrorMessages()`.

GlideSystem - `getEscapedProperty(String key, Object substituteObject)`

Retrieves the property and escapes it for XML parsing.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Key for the property whose value should be returned.</td>
</tr>
<tr>
<td>substituteObject</td>
<td>Object</td>
<td>Optional. Object to return if the property is not found.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Property, or the substituteObject if the property is not found.</td>
</tr>
</tbody>
</table>

```javascript
var propValue = gs.getEscapedProperty("com.example.my_test_property");
gs.addInfoMessage("This is my property value " + propValue);
```

GlideSystem - `getImpersonatingUserDisplayName()`

Returns the display name of the impersonating user.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Display name of the impersonating user.</td>
</tr>
</tbody>
</table>

This example shows how to obtain the impersonator's user display name.

```javascript
var user = gs.getUserDisplayName();
gs.print("The current user display name is: " + user);

var name = gs.getImpersonatingUserDisplayName();
gs.print("Impersonating user display name: " + name);
```

Output:

The current user display name is: abel.tuter
Impersonating user display name: admin

**GlideSystem - getImpersonatingUserName()**

Returns the name of the impersonating user or null if not impersonating.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of impersonating user</td>
</tr>
</tbody>
</table>

This example shows how to obtain the impersonator's user name.

```javascript
var name = gs.getImpersonatingUserName();
var actual_user = null;
if (name == null || name == '')
    actual_user = user;
else
    actual_user = Packages.com.glide.sys.User.getUser(name);

var recent_impersonations = actual_user.getPreference('recent.impersonations');
var admin_role=actual_user.hasRole("admin");
```
GlideSystem - getInfoMessages()

Retrieves the list of info messages for the session that were added by addInfoMessage().

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>List of info messages.</td>
</tr>
</tbody>
</table>

This example shows how to obtain the information messages.

```javascript
var messages = gs.getInfoMessages();
if (messages.toString().indexOf('Conflicts Detected') == -1)
    gs.addInfoMessage(msg);
```

GlideSystem - getInitials()

Returns the user's initials.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The user's initials.</td>
</tr>
</tbody>
</table>

```javascript
var userInitials = gs.getUser().getInitials();
gs.addInfoMessage('User initials: ' + userInitials);
```

Output:

User initials: SA
**GlideSystem - getMessage(String messageID, Object args)**

Retrieves translated messages from the Message [sys_ui_message] table to display in a UI.

If the specified message identifier (key) exists in the Message [sys_ui_message] table for the current language, the method returns the translated message. If the specified message identifier does not exist for the current language, the method returns the English version of the message. If the message identifier does not exist in the table, then it returns the message ID.

For additional information about the Message table, see [Message table](#).

硭 Note: If the UI message has a tick ('), there may be issues with the message in the script; to escape the ticks ('), use `getMessageS(String, Object)`.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>messageID</td>
<td>String</td>
<td>Message identifier. You can locate this value in the Key field of the Message [sys_ui_message] table. Note the Key field may look exactly like the actual message string.</td>
</tr>
<tr>
<td>args</td>
<td>Object</td>
<td>Optional. List of strings or other values defined by java.text.MessageFormat that replace the variables within the specified message. For example: <code>gs.getMessage(&quot;Abort adding action '{0}', same subflow can't be added twice in this subflow.&quot;, current.action.name);</code> In this example '{0}' is replaced with the content of current.action.name.</td>
</tr>
</tbody>
</table>

硭 Note: The passed in values are not translated. They are inserted verbatim in the message.

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Requested UI message.</td>
</tr>
</tbody>
</table>

This example shows the message returned for the current user when Spanish is the current language.
```javascript
var my_message = gs.getMessage("rows will not be updated");
alert(my_message);
```

Output

```
las filas no se actualizarán
```

This example shows how to replace a single variable within a message.

```javascript
// current.action.name is "update record"
var my_message = gs.getMessage("Abort adding action '{0}', same subflow can't be added twice in this subflow.", current.action.name);
alert(my_message);
```

Output

```
Abort adding action update record, same subflow can't be added twice in this subflow.
```

This example shows how to replace multiple variables within a message.

```javascript
// current.sub_flow.name is 'schedule users'
// current.action.name is "update record"
var my_message = gs.getMessage("Abort adding action '{0}', same subflow can't be added twice in {1} subflow.", [current.action.name, current.sub_flow.name]);
alert(my_message);
```

Output

```
Abort adding action update record, same subflow can't be added twice in schedule users subflow.
```

**Scoped equivalent**

To use the `getMessage()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - getMessage(String id, Array args)**.

**GlideSystem - getMessageS(String messageId, Object args)**

Retrieves translated messages to display in the UI and escapes all ticks (').

If the specified message identifier (key) exists in the Message [sys_ui_message] table for the current language, the method returns the translated message. If the specified message identifier does not exist for the current language, the method returns the English version of the message. If the message identifier does not exist in the table, then it returns the message ID.

For additional information about the Message table, see Message table.

Useful if you are inserting into a JavaScript expression from Jelly.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MessageID</td>
<td>String</td>
<td>Message identifier. You can locate this value in the Key field of the Message [sys_ui_message] table. Note the Key field may look exactly like the actual message string.</td>
</tr>
<tr>
<td>args</td>
<td>Object</td>
<td>Optional. List of strings or other values defined by java.text.MessageFormat that replace the variables within the specified message. For example: gs.getMessage(&quot;Abort adding action '{0}', same subflow can't be added twice in this subflow.&quot;, current.action.name); In this example '{0}' is replaced with the content of current.action.name.</td>
</tr>
</tbody>
</table>

**Note:** The passed in values are not translated. They are inserted verbatim in the message.

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Specified message with ticks escaped.</td>
</tr>
</tbody>
</table>

```java
/*
"I love France" translates to "J'aime la France" in French.
Rendering this without escaping the tick in "J'aime" would break Jelly, because the tick would prematurely end the variable assignment, and everything that follows (aime la France') would be a jelly syntax error.
*/
var my_message = '${gs.getMessageS("I love France")}';
alert(my_message);
```

Output:

J'aime la France

---

**GlideSystem** - **getNodeValue(Object obj, Number index)**

Returns the node value for the specified index.
function doInsert(nodeList) {
    gs.print('Doing insert');
    var task = new GlideRecord('task');
    task.initialize();
    for (var x=0; x < nodeList.size(); x++) {
        var name = gs.getNodeName(nodeList, x);
        var value = gs.getNodeValue(nodeList, x);
        task.setValue(name, value);
    }
    task.insert();
}

GlideSystem - getNodeName(Object obj, Number index)
Returns the node name for the specified index.

function doInsert(nodeList) {
    gs.print('Doing insert');
    var task = new GlideRecord('task');
    task.initialize();
    for (var x=0; x < nodeList.size(); x++) {
        var name = gs.getNodeName(nodeList, x);
        var value = gs.getNodeValue(nodeList, x);
        task.setValue(name, value);
    }
    task.insert();
}

GlideSystem - getNodeName(Object obj, Number index)
Returns the node name for the specified index.

function doInsert(nodeList) {
    gs.print('Doing insert');
    var task = new GlideRecord('task');
    task.initialize();
    for (var x=0; x < nodeList.size(); x++) {
        var name = gs.getNodeName(nodeList, x);
        var value = gs.getNodeValue(nodeList, x);
        task.setValue(name, value);
    }
    task.insert();
}

GlideSystem - getNodeName(Object obj, Number index)
Returns the node name for the specified index.
function doInsert(nodeList) {
  gs.print('Doing insert');
  var task = new GlideRecord('task');
  task.initialize();
  for (var x=0; x < nodeList.size(); x++) {
    var name = gs.getNodeName(nodeList, x);
    var value = gs.getNodeValue(nodeList, x);
    task.setValue(name, value);
  }
  task.insert();
}

GlideSystem - getPreference(String key, Object default)
Returns the specified user preference.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Key for the preference.</td>
</tr>
<tr>
<td>default</td>
<td>Object</td>
<td>Default value to use if the specified preference is not found.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Preference value. If no preference, returns the specified default value.</td>
</tr>
</tbody>
</table>

function getSelectedProject() {
  var array= new Array();
  var prj_id = gs.getPreference("prj_id"); //This will fetch value from user preference
  var gr = new GlideRecord('pm_project_task');
  gr.addQuery('parent', prj_id);
  gr.addActiveQuery();
  while(gr.next()) {
    array.push(gr.sys_id.toString());
  }
  return array;
}
GlideSystem - getProperty(String key, Object alt)

Returns the value of a Glide property. If the property is not found, returns the specified alt value.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Key for the property whose value should be returned.</td>
</tr>
<tr>
<td>alt</td>
<td>Object</td>
<td>Optional. Alternate object to return if the property is not found.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the Glide property, or the alternate object defined above.</td>
</tr>
</tbody>
</table>

//Check for attachments and add link if there are any
var attachment_link = '';
var rec = new GlideRecord('sc_req_item');
rec.addQuery('sys_id', current.request_item);
rec.query();
if(rec.next()){
  if(rec.hasAttachments()){
    attachment_link = gs.getProperty('glide.servlet.uri') + rec.getLink();
  }
}

GlideSystem - getScriptError(String script)

Returns the script error found in the specified script, if there is one.

Note: The script is not executed by this function, it is only checked for syntax errors.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>script</td>
<td>String</td>
<td>Script to check for errors.</td>
</tr>
</tbody>
</table>
**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Error message. Null if there is no error.</td>
</tr>
</tbody>
</table>

```java
if (gs.isValidScript(current.script) == false)
{
    current.setAbortAction(true);
    var error = gs.getScriptError(current.script);
    current.script.setError(error);
}
```

**GlideSystem - getSession()**

Returns a GlideSession object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideSession Object</td>
<td>GlideSession object for the current session</td>
</tr>
</tbody>
</table>

```java
if (!gs.hasRole("admin") && gs.getSession().isInteractive() && gs.getUserName() != "guest"){
    var qc = current.addQuery('u_visibility', "both");
    qc.addOrCondition('u_visibility', '');
    if (gs.getImpersonatingUserName() != null) {
        gs.getSession().clearClientData('navQuery');
    }
    var navQuery = gs.getSession().getClientData('navQuery');
    if (navQuery == null) {
        var isManager = gs.getUser().getRecord().getValue('u_is_manager');
        if (!isManager) {
            qc.addOrCondition('u_visibility', 'nonmanager');
            gs.getSession().putClientData('navQuery', 'nonmanager');
        } else {
            qc.addOrCondition('u_visibility', 'manager');
            gs.getSession().putClientData('navQuery', 'manager');
        }
    }
}
```
Scoped equivalent
To use the `getSession()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - getSession()**.

**GlideSystem - getSessionID()**

Returns the GlideSession session ID.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Session ID</td>
</tr>
</tbody>
</table>

```javascript
var gr_NOW = new GlideRecord('v_user_session');
var sessionId = gs.getSessionID();
gs.log("Session ID is: " + sessionId);
```

**Output**

```
Session ID is: FEE589B3DB7EE4103DD9C39D139619D7
```

Scoped equivalent
To use the `getSessionID()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - getSessionID()**.

**GlideSystem - getStyle(String tableName, String fieldName, String fieldValue)**

Returns the style defined for the table, field, and field value.
### GlideSystem - getUser()

Returns a reference to the user object for the current user.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideUser</td>
<td>Reference to a user object for the current user.</td>
</tr>
</tbody>
</table>

The following example shows how to get the current user object.

```javascript
var user = gs.getUser();
gs.print("The current user is: " + user);
```

**Output:**

```
The current user is: com.glide.sys.User@db5dd9
```

The following example shows how to check whether the current user has the workflow_admin role.
var role = gs.getUser().hasRole('workflow_admin');
gs.print("Does the current user have the workflow_admin role: "+role);

Output:
Does the current user have the workflow_admin role: true

Scoped equivalent
To use the `getUser()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - getUser().**

GlideSystem - getUserDisplayName()
Returns the name field of the current user. For example, the API returns John Smith instead of jsmith.

| Parameters | | |
|------------|---------------------------------|
| None       | | |

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

The following example shows how to get the current user's display name.

```javascript
var user = gs.getUserDisplayName();
gs.print("The current user display name is: "+user);
```

Output:
The current user display name is: abel.tuter

Scoped equivalent
To use the `getUserDisplayName()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - getUserDisplayName().**

GlideSystem - getUserID()
Returns the sys_id of the current user.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the current user.</td>
</tr>
</tbody>
</table>

This example gets the current user's sys_id.

```javascript
var currentUserId = gs.getUserID();
gs.print("Current user ID: "+currentUserId);
```

Output:

Current user ID: 6816f79cc0a8016401c5a33be04be441

Scoped equivalent

To use the `getUserID()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - getUserID()**.

**GlideSystem - getUserName()**

Returns the name of the current user, such as jsmith.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>User name of the current user.</td>
</tr>
</tbody>
</table>

The following example shows how to get the current user's display name object.

```javascript
var user = gs.getUserName();
gs.print("The current user name is: "+user);
```
```javascript
var impUser = new GlideImpersonate();
impUser.impersonate("62826bf03710200044e0bfc8bcbe5df1");
var user = gs.getUserName();
gs.print("The impersonated user name is: " + user);
```

**Output:**

The current user name is: admin
The impersonated user name is: abel.tuter

**Scoped equivalent**

To use the `gs.getUserName()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - getUserName().**

**GlideSystem - getUserNameByUserID(String user_id)**

Returns the user name based on a user ID.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user_id</td>
<td>String</td>
<td>User ID of the user whose user name to return.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the user.</td>
</tr>
</tbody>
</table>

The following example shows how to get the specified user's display name.

```javascript
var displayName = gs.getUserNameByUserID('abel.tuter');
gs.info('User display name is: ' + displayName);
```

**Output:**

User display name is: Abel Tuter

**GlideSystem - getXMLNodeList(String xml)**

Constructs an array of all the nodes and values in an XML document.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xml</td>
<td>String</td>
<td>XML document to parse.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array list</td>
<td>List of names and values of the outer most XML node.</td>
</tr>
</tbody>
</table>

The following example shows how to get a list of XML nodes and their associated values in an array list.

```javascript
var nodeList =
gs.getXMLNodeList("<note><to>Tove</to><from>Jani</from><heading>Reminder</heading><body>Don't forget me this weekend!</body></note>"; gs.print("The node contains: " + nodeList);
```

**Output:**

The node contains: [to : Tove, from : Jani, heading : Reminder, body : Don't forget me this weekend!]

---

**GlideSystem - getXMLText(String xml, String xpathQuery)**

Returns the XML text for the first element in the XML string that matches the XPath query.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xml</td>
<td>String</td>
<td>XML string to search.</td>
</tr>
<tr>
<td>xpathQuery</td>
<td>String</td>
<td>XPath query to match.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>XML node matching the search parameters.</td>
</tr>
</tbody>
</table>

The following example shows how to get the value of a specified XML element within a passed-in node.
var nodeList =
gs.getXMLText("<outer><note><to>Tove</to><from>Jani</from><heading>Reminder</heading><body>
Don't forget me this weekend!</body></note></outer>", "//from");
gs.print ("The from element contains: " + nodeList);

Output:
The from element contains: Jani

**GlideSystem - hasRole(String roleName)**

Determines if the current user has at least one of the passed-in roles.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>roleName</td>
<td>String</td>
<td>Comma-separated list of roles.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the current user has at least one of the specified roles. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: User has at least one of the passed-in roles. Also returns true if the user has the administrator role.</td>
</tr>
<tr>
<td></td>
<td>• false: User does not have any of the passed-in roles.</td>
</tr>
</tbody>
</table>

The following example shows how to check whether the current user has the admin or groups_admin role.

```javascript
if (!gs.hasRole("admin, groups_admin") && gs.getSession().isInteractive()) {
    var qc = current.addQuery("u_hidden", ",!-",",true"); //cannot see hidden groups...
    qc.addOrCondition("sys_id", "javascript:gteMyGroups()"); //...unless in the hidden group
}
```

**Scoped equivalent**

To use the `hasRole()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - hasRole(Object role)**.

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**GlideSystem - hasRoleInGroup(Object roleName, Object group)**

Determines if the current user has the specified role within a specified group.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>roleName</td>
<td>Object</td>
<td>Name of the role.</td>
</tr>
<tr>
<td>group</td>
<td>Object</td>
<td>Sys_id of the group to check for the specified role.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the current user has the specified role in the specified group. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: If all of the following conditions are met.</td>
</tr>
<tr>
<td></td>
<td>◦ The logged-in user is assigned to the specified role.</td>
</tr>
<tr>
<td></td>
<td>◦ The granted by field on the user role record is set to the specified group.</td>
</tr>
<tr>
<td></td>
<td>◦ The inherited field on the user role record is false.</td>
</tr>
<tr>
<td></td>
<td>• false: If any one of the conditions above are not met.</td>
</tr>
</tbody>
</table>

The following example shows how to check whether the sys_user_group group has the role_name role.

```javascript
var group = new GlideRecord('sys_user_group');
group.addQuery('name', 'GROUP_NAME');
group.setLimit(1);
group.query();
if (group.next()) {
    if (gs.hasRoleInGroup('role_name', group)) {
        gs.print('User has role in group');
    } else {
        gs.print('User does NOT have role in group');
    }
}
```
GlideSystem - hoursAgo(Number hours)

Returns a date and time for the specified number of hours ago. The returned value is adjusted for the time zone of the instance.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hours</td>
<td>Number</td>
<td>Number of hours ago.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>UTC date and time for the specified number of hours ago. Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

```javascript
var currentDateTime = gs.hoursAgo(0);
gs.info("Current date/time: " + currentDateTime);

var agoDateTime = gs.hoursAgo(1);
gs.info("Ago date/time: " + agoDateTime);
```

Output

Current date/time: 2021-03-17 20:53:25
Ago date/time: 2021-03-17 19:53:25

Scoped equivalent

To use the `hoursAgo()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - hoursAgo(Number hours)`.

GlideSystem - hoursAgoEnd(Number hours)

Returns a date and time for the end of the hour for the specified number of hours ago. The returned value is adjusted for the time zone of the instance.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hours</td>
<td>Number (Integer)</td>
<td>Number of hours ago.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>UTC date and time for the end of the specified number of hours ago. Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

```javascript
var currentDateTime = gs.hoursAgoEnd(0);
gs.print("Current date/time: " + currentDateTime);

var agoDateTime = gs.hoursAgoEnd(1);
gs.print("Ago date/time: " + agoDateTime);
```

Output

Current date/time: 2021-03-17 20:59:59
Ago date/time: 2021-03-17 19:59:59

Scoped equivalent

To use the `hoursAgoEnd()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - hoursAgoEnd(Number hours)**.

**GlideSystem - hoursAgoStart(Number hours)**

Returns a date and time for the start of the hour for the specified number of hours ago. The returned value is adjusted for the time zone of the instance.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hours</td>
<td>Number</td>
<td>Number of hours ago.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>UTC date and time for the end of the specified number of hours ago. Format: yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

```javascript
var currentDateTime = gs.hoursAgoStart(0);
gs.print("Current date/time: " + currentDateTime);
```
var agoDateTime = gs.hoursAgoStart(1);
gs.print("Ago date/time: " + agoDateTime);

Output

Current date/time: 2021-03-17 20:00:00
Ago date/time: 2021-03-17 19:00:00

Scoped equivalent

To use the `hoursAgoStart()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - hoursAgoStart(Number hours)`.

GlideSystem - isFirstOfMonth(Object date)

Checks whether the date in the specified date object is the first day of the month.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>Object</td>
<td>Date object on which to check the date, such as GlideDateTime or GlideDate.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean  | Flag that indicates whether the date in the specified date object is the first day of the month. Possible values:  
  • true: First day of the month.  
  • false: Not the first day of the month. |

This example show how to determine whether the date in the specified date object is the first day of the month.

```javascript
var gdt = new GlideDateTime();
gdt.setValue('2021-04-01 12:00:00');
var currentDay = gs.isFirstOfMonth(gdt);
gs.print("First day of month: " + currentDay);
```
gdt.setValue('2021-04-21 12:00:00');
var currentDay = gs.isFirstDayOfMonth(gdt);
gs.print("First day of month: " + currentDay);

Output
First day of month: true
First day of month: false

GlideSystem - isFirstDayOfWeek(Object date)
Checks whether the date in the specified date object is the first day of the week. This method uses the ISO standard of Monday being the first day of the week.

Parameters
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>Object</td>
<td>Date object on which to check the date, such as GlideDateTime or GlideDate.</td>
</tr>
</tbody>
</table>

Returns
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the date in the current date object is the first day of the week (Monday). Possible values: • true: First day of the week. • false: Not the first day of the week.</td>
</tr>
</tbody>
</table>

This example show how to determine whether the date in the specified date object is the first day of the week.

```javascript
var gdt = new GlideDateTime();
gdt.setValue('2021-03-02 12:00:00');
var currentDay = gs.isFirstDayOfWeek(gdt);
gs.info("First day of week: " + currentDay);

gdt.setValue('2021-03-22 12:00:00');
var currentDay = gs.isFirstDayOfWeek(gdt);
gs.info("First day of week: " + currentDay);
```

Output
First day of week: false
First day of week: true
GlideSystem - isFirstDayOfYear(Object date)
Checks whether the date in the specified date object is the first day of the year.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>Object</td>
<td>Date object on which to check the date, such as GlideDateTime or GlideDate.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the date in the specified date object is</td>
</tr>
<tr>
<td></td>
<td>the first day of the year.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: First day of the year.</td>
</tr>
<tr>
<td></td>
<td>• false: Not the first day of the year.</td>
</tr>
</tbody>
</table>

This example show how to determine whether the date in the specified date object is the first day of the year.

```javascript
var gdt = new GlideDateTime();
gdt.setValue('2020-12-31 12:00:00');
var currentDay = gs.isFirstDayOfYear(gdt);
gs.info("First day of year: "+ currentDay);

gdt.setValue('2021-01-01 12:00:00');
var currentDay = gs.isFirstDayOfYear(gdt);
gs.info("First day of year: "+ currentDay);
```

Output

First day of year: false
First day of year: true

GlideSystem - isInteractive()
Checks if the current session is interactive.
An example of an interactive session is when a user logs in using the log-in screen. An example of a non-interactive session is using a SOAP request to retrieve data.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates whether the session is interactive. Possible values:  
• true: Session is interactive.  
• false: Session is not interactive. |

```javascript
if (!gs.hasRole('admin') && gs.isInteractive()) {
    var qc1 = current.addQuery('u_group', '');
    var gra = new GlideRecord('sys_user_grmember');
    gra.addQuery('user', gs.getUserID());
    gra.query();
    while (gra.next()) {
        qc1.addOrCondition('u_group', gra.group);
    }
}
```

### Scoped equivalent

To use the `isInteractive()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - isInteractive()`.

### GlideSystem - isLastDayOfMonth(Object date)

Checks whether the date in the specified date object is the last day of the month.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>Object</td>
<td>Date object on which to check the date, such as GlideDateTime or GlideDate.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates whether the date in the specified date object is the last day of the month. Possible values:  
  • true: Last day of the month.  
  • false: Not the last day of the month. |

This example show how to determine whether the date in the specified date object is the last day of the month.

```javascript
var gdt = new GlideDate();
gdt.setValue('2020-12-31 12:00:00');
var currentDay = gs.isLastDayOfMonth(gdt);
gs.print("First day of year: " + currentDay);

gdt.setValue('2021-01-01 12:00:00');
var currentDay = gs.isLastDayOfMonth(gdt);
gs.print("First day of year: " + currentDay);
```

**Output**

```
Last day of month: true
Last day of month: false
```

### GlideSystem - isLastDayOfWeek(Object date)

Checks whether the date in the specified date object is the last day of the week. This method uses the ISO standard of Sunday being the last day of the week.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>Object</td>
<td>Date object on which to check the date, such as GlideDateTime or GlideDate.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean  | Flag that indicates whether the date in the current date object is the last day of the week (Sunday). Possible values:  
  • true: Last day of the week.  
  • false: Not the last day of the week. |

This example shows how to determine whether the date in the specified date object is the last day of the week.

```javascript
var gdt = new GlideDate();
gdt.setValue('2021-03-01 12:00:00');
var currentDay = gs.isLastDayOfWeek(gdt);
gs.info("Last day of week: " + currentDay);

gdt.setValue('2021-03-21 12:00:00');
var currentDay = gs.isLastDayOfWeek(gdt);
gs.info("Last day of week: " + currentDay);
```

**Output**

First day of week: false

First day of week: true

### GlideSystem - isLastDayOfYear(Object date)

Checks whether the date in the specified date object is the last day of the year.
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean  | Flag that indicates whether the date in the current date object is the last day of the year. Possible values:  
| • true: Last day of the year.  
| • false: Not the last day of the year. |

This example show how to determine whether the date in the specified date object is the last day of the year.

```javascript
var gdt = new GlideDateTime();
gdt.setValue('2020-12-31 12:00:00');
var currentDay = gs.isLastDayOfYear(gdt);
gs.info("Last day of year: " + currentDay);

gdt.setValue('2021-01-01 12:00:00');
var currentDay = gs.isLastDayOfYear(gdt);
gs.info("Last day of year: " + currentDay);
```

Output

```
Last day of year: true
Last day of year: false
```

**GlideSystem - isLoggedIn()**

Determines if the current user is currently logged in.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean  | Flag that indicates whether the current user is logged in. Possible values:  
|          |                                                                             |
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• true:</td>
<td>User is logged in.</td>
</tr>
<tr>
<td>• false:</td>
<td>User is not logged in.</td>
</tr>
</tbody>
</table>

This example shows how to determine whether the current user is logged in.

```java
if (gs.isLoggedIn())
    gs.info("Current user is logged in");
else
    gs.info("Current user is NOT logged in");
```

Output

```
Current user is logged in
```

**Scoped equivalent**

To use the `isLoggedIn()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - isLoggedIn()`.

**GlideSystem - isMobile()**

Determines whether the request came from a mobile device.

You can use this method in UI action conditions and business rules.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the request came from a mobile device. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Mobile device.</td>
</tr>
<tr>
<td></td>
<td>• false: Non-mobile device.</td>
</tr>
</tbody>
</table>
This example shows how to determine whether the current device is a mobile device.

```javascript
if(gs.isMobile())
    gs.info("Submitted from mobile UI");
else
    gs.info("NOT submitted from mobile UI");
```

**Output**

Submitted from mobile UI

**Scoped equivalent**

To use the `isMobile()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - isMobile()`.

**GlideSystem - lastWeek()**

Returns the date and time one week ago in GMT.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>String</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

This example shows how to obtain the date that is one week from the date in the current date object.

```javascript
var gdt = new GlideDateTime();
gdt.setValue('2021-03-22 01:00:00');

gs.info("Current date: " + gs.nowDateTime());
gs.info("One week earlier: " + gs.lastWeek());
```

**Output**
GlideSystem - log(String message, String source)
Logs a message to the system log and saves it to the syslog table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Message to log.</td>
</tr>
<tr>
<td>source</td>
<td>String</td>
<td>Optional. Source of the message.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to post a log entry using constants and variables.

```javascript
var count = new GlideAggregate('incident');
count.addQuery('active', 'true');
count.addAggregate('COUNT', 'category');
count.query();
while (count.next()) {
    var category = count.category;
    var categoryCount = count.getAggregate('COUNT', 'category');
    gs.log("The are currently " + categoryCount + " incidents with a category of " + category, "Incident Counter");
}
```

GlideSystem - logError(String message, String source)
Logs an error to the system log and saves it to the syslog table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Error message to log.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>String</td>
<td>Optional. Source of the message.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example logs an error if the Incident table cannot be found.

```javascript
var gr_NOW = new GlideRecord("incident");
if (!gr_NOW.isValid()) {
    gs.logError('Incident table could not be found');
    return;
}
```

**GlideSystem - logWarning(String message, String source)**

Logs a warning to the system log and saves it to the syslog table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Message to log.</td>
</tr>
<tr>
<td>source</td>
<td>String</td>
<td>Optional. Source of the message.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example logs a warning if the Incident table cannot be found.

```javascript
var gr_NOW = new GlideRecord("incident");
if (!gr_NOW.isValid()) {
    gs.logWarning('Incident table could not be found');
    return;
}
```
**GlideSystem - minutesAgo(Number minutes)**

Returns a date and time for the specified number of minutes ago.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>minutes</td>
<td>Number</td>
<td>Number of minutes in the past to return.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>UTC date and time for the specified number of minutes ago.</td>
</tr>
<tr>
<td></td>
<td>Format: <code>yyyy-mm-dd hh:mm:ss</code></td>
</tr>
</tbody>
</table>

This example shows how to use the minutesAgo() method in an addQuery() call.

```java
// Check to see if the user has failed to login too many times
// when the limit is reached, lock the user out of the system
//
// Check failed logins in the last 15 minutes
var now_GR = new GlideRecord('sysevent');
now_GR.addQuery('name', 'login.failed');
now_GR.addQuery('parm1', event.parm1.toString());
now_GR.addQuery('sys_created_on', '>=', gs.minutesAgo(15));
now_GR.query();
var rowCount = now_GR.getRowCount();
if(rowCount >= 5){
    var now_GR = new GlideRecord("sys_user");
    now_GR.addQuery("user_name", event.parm1.toString());
    now_GR.query();
    if (now_GR.next()) {
        now_GR.locked_out = true;
        now_GR.update();
        gs.log("User " + event.parm1 + " locked out due to too many invalid login attempts");
    }
}
```

**GlideSystem - minutesAgoEnd(Number minutes)**

Returns a date and time for the end of the minute a certain number of minutes ago.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>minutes</td>
<td>Number</td>
<td>An integer number of minutes</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped equivalent

To use the `minutesAgoEnd()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - minutesAgoEnd(Number minutes)`.

**GlideSystem - minutesAgoStart(Number minutes)**

Returns a date and time for the start of the minute a certain number of minutes ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>minutes</td>
<td>Number</td>
<td>An integer number of minutes</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped equivalent

To use the `minutesAgoStart()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - minutesAgoStart(Number minutes)`.

**GlideSystem - monthsAgo(Number months)**

Returns a date and time for a certain number of months ago.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>months</td>
<td>Number</td>
<td>An integer number of months</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT on today's date of the specified month, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped equivalent

To use the `monthsAgo()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - monthsAgo(Number months)`.

`GlideSystem - monthsAgoEnd(Number months)`

Returns a date and time for the last day of the month a certain number of months ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>months</td>
<td>Number</td>
<td>An integer number of months</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT end of the month the specified number of months ago, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

`GlideSystem - monthsAgoStart(Number months)`

Returns a date and time for the start of the month a certain number of months ago.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>months</td>
<td>Number</td>
<td>An integer number of months</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT start of the month the specified number of months ago, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

### Scoped equivalent

To use the `monthsAgoStart()` method in a scoped application, use the corresponding scoped method: **Scoped GlideSystem - monthsAgoStart(Number months)**.

### GlideSystem - nil(Object obj)

Queries an object and returns true if the object is null or contains an empty string.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>Object</td>
<td>The object to be checked.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if null or empty string; otherwise, false.</td>
</tr>
</tbody>
</table>

```javascript
if (!current.u_date1.nil() && !current.u_date2.nil()) {
    var start = current.u_date1.getGlideObject().getNumericValue();
    var end = current.u_date2.getGlideObject().getNumericValue();
    if (start > end) {
        gs.addInfoMessage('start must be before end');
        current.u_date1.setError('start must be before end');
        current.setAbortAction(true);
    }
}
```
Scoped equivalent

To use the `nil()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - nil(Object o)`.

**GlideSystem - now()**

Returns the current date in UTC.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The current date in the user-defined format, in UTC.</td>
</tr>
</tbody>
</table>

```javascript
// When the user password changes then set the u_password_last_reset field
// to now so we know when to force another update
var now_GR = new GlideRecord("sys_user");
if (now_GR.get(event.parm1.toString())) {
    // Do something based on the Password Changing
    gs.log("The user password changed so do something else...");
    now_GR.u_password_last_reset = gs.now();
    now_GR.update();
}
```

**GlideSystem - nowNoTZ()**

Returns the current date and time in UTC format.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The current UTC date time</td>
</tr>
</tbody>
</table>

// When the user password changes then set the u_password_last_reset field
// to now so we know when to force another update

var now GR = new GlideRecord("sys_user");
if (now GR.get(event.parm1.toString())) {
    // Do something based on the Password Changing
    gs.log("The user password changed so do something else...");
    now GR.u_password_last_reset = gs.nowNoTZ();
    now GR update();
}

GlideSystem - nowDateTime()

Returns the current date and time in the user-defined format.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Current date and time in the user-defined format. For more information on setting the system date and time format, see Date and Date/Time fields.</td>
</tr>
</tbody>
</table>

var currentDate Time = gs.nowDateTime();

gs.print("Current date/time: " + currentDate Time);

Output

Current date/time: 2021-03-17 14:04:02
When setting a variable in a workflow script to the current date and time, use the `setDisplayValue()` method. The following script sets the workflow variable `end_date` to the current date and time.

```
current.variables.end_date.setDisplayValue(gs.nowDateTime());
```

**GlideSystem - nowGlideDateTime()**

Gets a GlideDateTime object with the current date and time.

After you get a GlideDateTime object with the current date and time, you can use the GlideDateTime methods to perform date-time operations, such as performing date-time calculations, formatting a date-time, or converting between date-time formats.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>Object</td>
</tr>
</tbody>
</table>

Sets the field `u_target_date` to the current date and time in GMT format.

```
current.u_target_date = gs.nowGlideDateTime();
```

**GlideSystem - print(String message)**

Writes a message to the system log.

This method does not write the message to the syslog table unless debug has been activated.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>message</td>
</tr>
</tbody>
</table>
void

var rec = new GlideRecord('incident');
rec.addQuery('active',false);
rec.query();
while (rec.next()) {
  gs.print('Inactive incident ' + rec.number + ' deleted');
  rec.deleteRecord();
}

GlideSystem - quartersAgo(Number quarters)
Returns a date and time for a certain number of quarters ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>quarters</td>
<td>Number</td>
<td>An integer number of quarters</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of the quarter that was the specified number of quarters ago, in the format yyyy-mm-dd hh:mm:ss.</td>
</tr>
</tbody>
</table>

GlideSystem - quartersAgoEnd(Number quarters)
Returns a date and time for the last day of the quarter, for a specified number of quarters ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>quarters</td>
<td>Number</td>
<td>An integer number of quarters</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT end of the quarter that was the specified number of quarters ago, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped equivalent

To use the `quartersAgoEnd()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - quartersAgoEnd(Number quarters)`.

**GlideSystem - quartersAgoStart(Number quarters)**

Returns a date and time for the first day of the quarter, for a specified number of quarters ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>quarters</td>
<td>Number</td>
<td>An integer number of quarters</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT end of the month that was the specified number of quarters ago, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped equivalent

To use the `quartersAgoStart()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - quartersAgoStart(Number quarters)`.

**GlideSystem - setProperty(String key, String value, String description)**

Sets the specified key to the specified value.
**Note:** Care should be taken when setting system properties (sys_properties) using this method as it causes a system-wide cache flush. Each flush can cause system degradation while the caches rebuild. If a value must be updated often, it should not be stored as a system property. In general, you should only place values in the sys_properties table that do not frequently change.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>The key for the property to be set.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The value of the property to be set.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>A description of the property.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
gs.setProperty("glide.foo","bar","foo");
gs.info(gs.getProperty("glide.foo"));
```

Output: bar

**Scoped equivalent**

To use the `setProperty()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - setProperty(String key, String value, String description)`.

### GlideSystem - setRedirect(Object URI)

Sets the redirect URI for this transaction, which then determines the next page the user will see.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URI</td>
<td>Object</td>
<td>URI to set as the redirect</td>
</tr>
</tbody>
</table>
This example redirects the user to a particular catalog item, and passes along the current email as a parameter.

```java
gs.setRedirect("com.glideapp.servicecatalog_cat_item_view.do?sysparm_id=d41ce5bac611227a0167f4bf8109bf70&sysparm_user="+
current.sys_id + "+&sysparm_email=" + current.email)
```

**Scoped equivalent**

To use the `setRedirect()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - setRedirect(Object o)`.

**GlideSystem - setReturn(Object URI)**

Sets the return URI for this transaction. This determines what page the user will be directed to when they return from the next form.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URI</td>
<td>Object</td>
<td>URI to set as the return location.</td>
</tr>
</tbody>
</table>

This example ensures that the user will be returned to the current page when they are done with the next one.

```java
gs.setReturn (current.getLink(true));
```

**GlideSystem - tableExists(String tableName)**

Determines if a database table exists.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the table exists, false otherwise.</td>
</tr>
</tbody>
</table>

Scoped equivalent

To use the `tableExists()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - tableExists(String name)`.

**GlideSystem - userID()**

Returns the sys_id of the user associated with this session. Use `getUserID()` instead.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>sys_id of the current user</td>
</tr>
</tbody>
</table>

**GlideSystem - workflowFlush(Object glideRecord)**

Deletes any open scheduled job records in the Schedule (sys_trigger) table for the specified GlideRecord.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>glideRecord</td>
<td>Object</td>
<td>The GlideRecord</td>
</tr>
</tbody>
</table>
GlideSystem - yearsAgo(Number years)

Gets a date and time for a certain number of years ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>years</td>
<td>Number</td>
<td>An integer number of years</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of the year that is the specified number of years ago, in the format yyyy-mm-dd hh:mm:ss.</td>
</tr>
</tbody>
</table>

Scoped equivalent

To use the yearsAgo(Number years) method in a scoped application, use the corresponding scoped method: Scoped GlideSystem - yearsAgo(Number years).

GlideSystem - yesterday()

Returns yesterday's time (24 hours ago).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT for 24 hours ago, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>
Scoped equivalent

To use the `yesterday()` method in a scoped application, use the corresponding scoped method: `Scoped GlideSystem - yesterday()`.

GlideSystem - Scoped

The scoped GlideSystem (referred to by the variable name 'gs' in any server-side JavaScript) API provides a number of convenient methods to get information about the system, the current logged in user, etc.

Many of the GlideSystem methods facilitate the easy inclusion of dates in query ranges, and are most often used in filters and reporting.

Scoped GlideSystem - addErrorMessage(String message)

Adds an error message for the current session.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>message</td>
<td>String</td>
<td>Message to add.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
gs.include("PrototypeServer");
var ValidatePasswordStronger = Class.create();
ValidatePasswordStronger.prototype = {
    process : function() {
        var user_password = request.getParameter("user_password");
        var min_len = 8;
        var rules = "Password must be at least " + min_len + " characters long and contain a digit, an uppercase letter, and a lowercase letter.";
        if (user_password.length() < min_len) {
            gs.addErrorMessage("TOO SHORT: " + rules);
            return false;
        }
        var digit_pattern = new RegExp("[0-9]", "g");
        if (!digit_pattern.test(user_password)) {
            gs.addErrorMessage("DIGIT MISSING: " + rules);
        }
    }
};
```
return false;
}
var upper_pattern = new RegExp("[A-Z]", "g");
if (!upper_pattern.test(user_password)) {
    gs.addErrorMessage("UPPERCASE MISSING: " + rules);
    return false;
}
var lower_pattern = new RegExp("[a-z]", "g");
if (!lower_pattern.test(user_password)) {
    gs.addErrorMessage("LOWERCASE MISSING: " + rules);
    return false;
}
return true; // password is OK
}

Scoped GlideSystem - addInfoMessage(String message)
Adds an info message for the current session. This method is not supported for asynchronous business rules.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Info message to add.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
Scoped GlideSystem - base64Encode(String source)
Creates a base64 string from the specified string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>String</td>
<td>String to encode.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Encoded base64 string.</td>
</tr>
</tbody>
</table>

Example

```java
var mac = new CertificateEncryption;
var key = "sample_key";
key = gs.base64Encode(key);
mac.generateMac(key, "HmacSHA256", "sample_data");
```

Scoped GlideSystem - base64Decode(String source)
Returns an ASCII string from the specified base64 string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>String</td>
<td>A base64 encoded string.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The decoded string.</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - beginningOfLastMonth()
Returns the date and time for the beginning of last month in GMT.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of last month, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - beginningOfLastWeek()

Returns the date and time for the beginning of last week in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of last week, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - beginningOfNextMonth()

Returns the date and time for the beginning of next month in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of next month, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>
Scoped GlideSystem - beginningOfWeek()
Returns the date and time for the beginning of next week in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The GMT beginning of next week, in the format yyyy-mm-dd hh:mm:ss.</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - beginningOfYear()
Returns the date and time for the beginning of next year in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of next year, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - beginningOfMonth()
Returns the date and time for the beginning of this month in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of next year, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of this month, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - beginningOfThisQuarter()

Returns the date and time for the beginning of this quarter in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of this quarter, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - beginningOfThisWeek()

Returns the date and time for the beginning of this week in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of this week, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - beginningOfThisYear()

Returns the date and time for the beginning of this year in GMT.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>String</td>
<td>Format: yyyy-mm-dd</td>
</tr>
<tr>
<td>range</td>
<td>String</td>
<td>Start, end, or a time in the 24 hour format hh:mm:ss.</td>
</tr>
</tbody>
</table>

**Scoped GlideSystem - daysAgo(Number days)**

Returns the date and time for a specified number of days ago.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>days</td>
<td>Number</td>
<td>Integer number of days</td>
</tr>
</tbody>
</table>
function contractNoticeDue() {
    var now_GR = new GlideRecord("contract");
    now_GR.addQuery("u_contract_status", "Active");
    now_GR.query();
    while (now_GR.next()) {
        if ((now_GR.u_termination_date <= gs.daysAgo(-90)) && (now_GR.u_contract_duration == "Long")) {
            now_GR.u_contract_status = "In review";
        } else if ((now_GR.u_termination_date <= gs.daysAgo(-50)) &&
            (now_GR.u_contract_duration == "Medium")) {
            now_GR.u_contract_status = "In review";
        } else if ((now_GR.u_termination_date <= gs.daysAgo(-10)) &&
            (now_GR.u_contract_duration == "Short")) {
            now_GR.u_contract_status = "In review";
        }
    }
    now_GR.update();
}

Scoped GlideSystem - daysAgoEnd(Number days)
Returns the date and time for the end of the day a specified number of days ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>days</td>
<td>Number</td>
<td>Integer number of days</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT end of the day in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>
Scoped GlideSystem - daysAgoStart(Number days)

Returns the date and time for the beginning of the day a specified number of days ago.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>days</td>
<td>String</td>
<td>Integer number of days</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT start of the day in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('sysapproval_approver');
now_GR.addQuery('state', 'requested');
now_GR.addQuery('sys_updated_on', '<', gs.daysAgoStart(5));
now_GR.query();
```

Scoped GlideSystem - debug(String message, Object parm1, Object parm2, Object parm3, Object parm4, Object parm5)

Writes a debug message to the system log.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The log message with place holders for any variable arguments.</td>
</tr>
<tr>
<td>parm1</td>
<td>Object</td>
<td>(Optional) First variable argument.</td>
</tr>
<tr>
<td>parm2</td>
<td>Object</td>
<td>(Optional) Second variable argument.</td>
</tr>
<tr>
<td>parm3</td>
<td>Object</td>
<td>(Optional) Third variable argument.</td>
</tr>
<tr>
<td>parm4</td>
<td>Object</td>
<td>(Optional) Fourth variable argument.</td>
</tr>
<tr>
<td>parm5</td>
<td>Object</td>
<td>(Optional) Fifth variable argument.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
gs.debug("This is a debug message");
var myFirstName = "Abel";
var myLastName = "Tuter";
gs.debug("This is a debug message from {0}.{1}", myFirstName, myLastName);
```

Output:

This is a debug message (sys.scripts extended logging)
This is a debug message from Abel.Tuter (sys.scripts extended logging)

**Scoped GlideSystem - endOfLastMonth()**

Returns the date and time for the end of last month in GMT.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT end of last month, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

**Scoped GlideSystem - endOfLastWeek()**

Returns the date and time for the end of last week in GMT.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT end of last week, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

 Scoped GlideSystem - endOfLastYear()

Returns the date and time for the end of last year in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

 Scoped GlideSystem - endOfNextMonth()

Returns the date and time for the end of next month in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

 Scoped GlideSystem - endOfNextWeek()

Returns the date and time for the end of next week in GMT.
### Scoped GlideSystem - endOfNextYear()

Returns the date and time for the end of next year in GMT.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

### Scoped GlideSystem - endOfThisMonth()

Returns the date and time for the end of this month in GMT.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>
Scoped GlideSystem - endOfThisQuarter()

Returns the date and time for the end of this quarter in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - endOfThisWeek()

Returns the date and time for the end of this week in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - endOfThisYear()

Returns the date and time for the end of this year in GMT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - error(String message, Object parm1, Object parm2, Object parm3, Object parm4, Object parm5)

Writes an error message to the system log.

This method accepts up to five variable arguments (varargs) in the message using the Java MessageFormat placeholder replacement pattern.

Note: Variables must contain valid values for this method to provide correct output.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The log message with place holders for any variable arguments.</td>
</tr>
<tr>
<td>param1</td>
<td>Object</td>
<td>(Optional) First variable argument.</td>
</tr>
<tr>
<td>param2</td>
<td>Object</td>
<td>(Optional) Second variable argument.</td>
</tr>
<tr>
<td>param3</td>
<td>Object</td>
<td>(Optional) Third variable argument.</td>
</tr>
<tr>
<td>param4</td>
<td>Object</td>
<td>(Optional) Fourth variable argument.</td>
</tr>
<tr>
<td>param5</td>
<td>Object</td>
<td>(Optional) Fifth variable argument.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
gs.error("This is an error message");
var myFirstName = "Abel";
var myLastName = "Tuter";
gs.error("This is an error message from {0}.{1}", myFirstName, myLastName);
```

Output:
This is an error message
This is an error message from Abel.Tuter

Scoped GlideSystem - eventQueue(String name, Object instance, String parm1, String parm2, String queue)

Queues an event for the event manager.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the event being queued.</td>
</tr>
<tr>
<td>instance</td>
<td>Object</td>
<td>GlideRecord object, such as &quot;current&quot;.</td>
</tr>
<tr>
<td>parm1</td>
<td>String</td>
<td>Optional. Saved with the instance if specified.</td>
</tr>
<tr>
<td>parm2</td>
<td>String</td>
<td>Optional. Saved with the instance if specified.</td>
</tr>
<tr>
<td>queue</td>
<td>String</td>
<td>Optional. Name of the queue.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

if (current.operation() != 'insert' && current.comments.changes()) {
    gs.eventQueue('incident.commented', current, gs.getUserID(), gs.getUserName());
}

Scoped GlideSystem - eventQueueScheduled(String name, Object instance, String parm1, String parm2, Object expiration)

Queues an event in the event manager.

The passed in event schedule information is stored in the Events [sysevent] table. For additional information on events, see Events.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the event to queue.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance</td>
<td>Object/ String</td>
<td>GlideRecord object or the sys_id of the record to which this event applies.</td>
</tr>
<tr>
<td>parm1</td>
<td>String</td>
<td>Optional. String to pass into the event script. This parameter is free-form and depends on the implementation of the event script. Default: If the instance parameter is a GlideRecord object, then the default is the display value for that GlideRecord (now_GR.getDisplayValue) otherwise it is null.</td>
</tr>
<tr>
<td>parm2</td>
<td>String</td>
<td>Optional. String to pass into the event script. This parameter is free-form and depends on the implementation of the event script. Default: null</td>
</tr>
<tr>
<td>expiration</td>
<td>Object</td>
<td>Optional. GlideDateTime object or a date/time type element that specifies the date and time to process the event. Default: Current date/time</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to call the method using a GlideRecord, two parameters that are passed to the event script, and a specified time to schedule the event.

```javascript
var requestXml = requestBody.dataString;
var processTime = new GlideDateTime();
processTime.addDaysLocalTime(7);
gs.eventQueueScheduled('sn_app.user.deactivate', now_GR, requestXML, gs.getUserID(), processTime);
```

This example shows how to call the method using a sys_id and the default values for the optional parameters.

```javascript
gs.eventQueueScheduled('event.test', '0e29421383101000dada83ec37d9292d', '', '', '');
```
Scoped GlideSystem - executeNow(GlideRecord job)
Executes a job for a scoped application.
You can only use this method on a job in the same application as the script calling this method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>job</td>
<td>GlideRecord</td>
<td>Job to run.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Returns the sys_id of the scheduled job. Returns null if the job is global.</td>
</tr>
</tbody>
</table>

Example

```javascript
scheduleImportJob: function () {
    var jobId = '61847fe04c603300fa9bb64c2b491dac';
    var now_GR = new GlideRecord('sysauto_script');
    if (!now_GR.get(jobId)) {
        throw new Error('Unable to find import job');
    }
    gs.executeNow(now_GR);
}
```

Scoped GlideSystem - generateGUID()
Generates a GUID that can be used when a unique identifier is required.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A 32-character hexadecimal GUID.</td>
</tr>
</tbody>
</table>
personalId = gs.generateGUID();
gs.info(personalId);

Output: af770511ff013100e04bfffffffff6

**Scoped GlideSystem - getCallerScopeName()**

Gets the caller scope name; returns null if there is no caller.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The caller's scope name, or null if there is no caller.</td>
</tr>
</tbody>
</table>

This example shows two script includes that are set to Accessible from all application scopes. One script include is created within app_scope_a.

```javascript
var Scopea = Class.create();

Scopea.prototype = {
  initialize: function() {
  },

  callerScope: function() {
    var scopeb = new app_scope_b.Scopeb();
    return scopeb.callerscope();
  },

  type: 'Scopea'
}
```

Output:

This example shows another script include created within app_scope_b.

```javascript
var Scopeb = Class.create();

Scopeb.prototype = {
  initialize: function() {
```
```javascript
this._constructorCallerScope = gs.getCallerScopeName();

callerscope: function() {
    return gs.getCallerScopeName();
},

getConstructorCallerScope: function() {
    return this._constructorCallerScope;
},

type: 'Scopeb'
```

Output:

This script can be used within scope app_scope_a to get the scope name of the caller—in this case app_scope_b.

```javascript
gs.info(new Scopea().getCallerScopeName());
```

**Scoped GlideSystem - getCssCacheVersionString()**

Gets a string representing the cache version for a CSS file.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The CSS cache version.</td>
</tr>
</tbody>
</table>

```javascript
var verStr = gs.getCssCacheVersionString();
gs.info(verStr);
```

Output: _d82979516f0171005be8883e6b3ee4cf&theme=

**Scoped GlideSystem - getCurrentApplicationId()**

Gets the ID of the current application as set using the Application Picker.
**Scoped GlideSystem - getCurrentApplicationId()**

Gets the current application's sys_id, or global in none is set.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var currentId = gs.getCurrentApplicationId();
gs.info(currentId);
```

Output: `04936cb16f30b1005be8883e6b3ee4e0`

**Scoped GlideSystem - getCurrentScopeName()**

Gets the name of the current scope.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var currentScope = gs.getCurrentScopeName();
gs.info(currentScope);
```

**Scoped GlideSystem - getErrorMessages()**

Returns the error messages that were added by `addErrorMessage()` for the session.

```javascript
var errorMessage = gs.getErrorMessages();
gs.info(errorMessage);
```
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array of Strings</td>
<td>Error messages associated with the session.</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - getEscapedMessage(String id, Array args)

Retrieves the specified message from the Message [sys_ui_message] table. If the message has HTML special characters, replaces them with the corresponding HTML name codes. For example, & becomes &amp;.

If the specified message identifier (Key) exists in the Message [sys_ui_message] table for the current language, the method returns the translated message with all special characters returned as escape sequences. If the specified message identifier does not exist for the current language, then the method returns the English version of the message with all special characters returned as escape sequences. If the message identifier does not exist in the table, then it returns the message ID.

For additional information about the Message table, see Message table.

Note: If the UI message has a tick (‘), there may be issues with the message in the script; to escape the ticks (‘), use getMessageS(String, Object).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Message identifier. You can locate this value in the Key field of the Message [sys_ui_message] table. Note the Key field may look exactly like the actual message string.</td>
</tr>
<tr>
<td>args</td>
<td>Array</td>
<td>Optional. List of strings or other values defined by java.text.MessageFormat that replace the variables within the specified message. For example: gs.getMessage(&quot;Abort adding action '{0}', same subflow can't be added twice in this subflow.&quot;, current.action.name);</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>In this example '{0}' is replaced with the content of <code>current.action.name</code>.</td>
</tr>
</tbody>
</table>

⚠ **Note:** The passed in values are not translated. They are inserted verbatim in the message.

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>UI message with HTML special characters replaced with HTML name codes.</td>
</tr>
</tbody>
</table>

This example shows the ampersand character being replaced with the equivalent ampersand HTML name code.

```javascript
var my_message = gs.getEscapedMessage("Is the summary & details accurate?");
alert(my_message);
```

Output

Is the summary &amp; details accurate?

This example shows how to replace a single variable within a message and how the `<` and `>` symbols in the message are transposed to the corresponding HTML name codes of `&lt` and `&gt`.

```javascript
// current.action.name is "update record"
var my_message = gs.getEscapedMessage("Abort adding action '{0}', same <subflow> can't be added twice in this subflow.", current.action.name);
alert(my_message);
```

Output

Abort adding action update record, same &ltsubflow&gt can't be added twice in this subflow.

### Scoped GlideSystem - getMessage(String id, Array args)

Retrieves translated messages from the Message [sys_ui_message] table to display in a UI.

If the specified message identifier (key) exists in the Message [sys_ui_message] table for the current language, then the method returns the translated message.
If the specified message identifier does not exist for the current language, then the method returns the English version of the message. If the message identifier does not exist in the table, then it returns the message ID.

For additional information about the Message table, see Message table

Note: If the UI message has a tick ('), there may be issues with the message in the script; to escape the ticks ('), use getMessages(String, Object).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Message identifier. You can locate this value in the Key field of the Message [sys_ui_message] table. Note the Key field may look exactly like the actual message.</td>
</tr>
<tr>
<td>args</td>
<td>Array</td>
<td>Optional. List of strings or other values defined by java.text.MessageFormat that replace the variables within the specified message. For example: gs.getMessage(&quot;Abort adding action '{0}', same subflow can't be added twice in this subflow.&quot;, current.action.name); In this example '{0}' is replaced with the content of current.action.name. Note: The passed in values are not translated. They are inserted verbatim in the message.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>UI message.</td>
</tr>
</tbody>
</table>

This example shows the message returned for the current user when Spanish is the current language.

```javascript
var my_message = gs.getMessage("rows will not be updated");
alert(my_message);
```

Output

```
las filas no se actualizarán
```

This example shows how to replace a single variable within a message.
// current.action.name is "update record"
var my_message = gs.getMessage("Abort adding action '{0}', same subflow can't be added
twice in this subflow.", current.action.name);
alert(my_message);

Output
Abort adding action update record, same subflow can't be added twice in this subflow.

This example shows how to replace multiple variables within a message.

// current.sub_flow.name is 'schedule users'
// current.action.name is "update record"
var my_message = gs.getEscapedMessage("Abort adding action '{0}', same subflow can't be
added twice in {1} subflow.", [current.action.name, current.sub_flow.name]);
alert(my_message);

Output
Abort adding action update record, same subflow can't be added twice in schedule users
subflow.

Scoped GlideSystem - getProperty(String key, Object alt)

Gets the value of a Glide property. If the property is not found, returns an alternate value.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>The key for the property whose value should be returned.</td>
</tr>
<tr>
<td>alt</td>
<td>Object</td>
<td>(Optional) Alternate object to return if the property is not found.</td>
</tr>
</tbody>
</table>

var attachment_link = gs.getProperty('glide.servlet.uri');
gs.info(attachment_link);

Output: https://instance.service-now.com/
Scoped GlideSystem - getSession()

Gets a reference to the current Glide session.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A reference for the current session.</td>
</tr>
</tbody>
</table>

```javascript
if (!gs.hasRole("admin") && !gs.hasRole("user_admin") && gs.getSession().isInteractive()) {
    current.addQuery("active", "true");
}
```

Scoped GlideSystem - getSessionID()

Retrieves the GlideSession session ID.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The session ID.</td>
</tr>
</tbody>
</table>

```javascript
var myUserObject = gs.getSessionID();
gs.info(myUserObject);
```

Output:

A0D4E5416F3F21005BE8883E6B3EE4B8
Scoped GlideSystem - getSessionToken()

This method is no longer available. Instead, use gs.getSession().getSessionToken().

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The session token.</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - getTimeZoneName()

Returns the name of the time zone associated with the current user.

This method has been deprecated. Instead, use the Scoped GlideSession - getTimeZoneName() method in the GlideSession API.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The time zone name.</td>
</tr>
</tbody>
</table>

gs.info(gs.getTimeZoneName());

Scoped GlideSystem - getUrlOnStack()

Gets the current URI for the session.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The URI.</td>
</tr>
</tbody>
</table>

```javascript
gs.info(gs.getUrlOnStack());
```

Scoped GlideSystem - getUser()

Returns a reference to the scoped GlideUser object for the current user.

See GlideUser - Scoped for a list of available methods.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideUser</td>
<td>Reference to a scoped user object.</td>
</tr>
</tbody>
</table>

```javascript
var myUserObject = gs.getUser();
```

Scoped GlideSystem - getUserDisplayName()

Gets the display name of the current user.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>The name field of the current user. Returns Abel Tuter, as opposed to abel.tuter.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
gs.info(gs.getUserDisplayName());
```

**Scoped GlideSystem - getUserID()**

Gets the sys_id of the current user.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

```javascript
gs.info(gs.getUserID());
```

**Scoped GlideSystem - getUserName()**

Gets the user name, or user id, of the current user.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

```javascript
gs.info(gs.getUserName());
```
Scoped GlideSystem - hasRole(Object role)
Determines if the current user has the specified role.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>role</td>
<td>Object</td>
<td>The role to check.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the user had the role. Returns true for users with the administrator role.</td>
</tr>
</tbody>
</table>

```java
if (!gs.hasRole("admin") && !gs.hasRole("groups_admin") && gs.getSession().isInteractive()) {
    var qc = current.addQuery("u_hidden", "!=" , "true"); // cannot see hidden groups...
    qc.addOrCondition("sys_id", "javascript:getMyGroups()" ); // ... unless in the hidden group
    gs.info("User has admin and groups admin roles");
} else {
    gs.info("User does not have both admin and groups admin roles");
}
```

Scoped GlideSystem - hoursAgo(Number hours)
Returns the date and time for a specified number of hours ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hours</td>
<td>Number</td>
<td>Integer number of hours</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>
if (current.operation() == 'insert') {
    // If no due date was specified, calculate a default
    if (current.due_date == '') {
        if (current.urgency == '1') {
            // Set due date to 4 hours ahead of current time
            current.due_date = gs.hoursAgo(-4);
        }
        if (current.urgency == '2') {
            // Set due date to 2 days ahead of current time
            current.due_date = gs.daysAgo(-2);
        }
        if (current.urgency == '3') {
            // Set due date to 7 days ahead of current time
            current.due_date = gs.daysAgo(-7);
        }
    }
}

Scoped GlideSystem - hoursAgoEnd(Number hours)
Returns the date and time for the end of the hour a specified number of hours ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hours</td>
<td>Number</td>
<td>Integer number of hours</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - hoursAgoStart(Number hours)
Returns the date and time for the start of the hour a specified number of hours ago.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hours</td>
<td>Number</td>
<td>Integer number of hours</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - include(String name)

Provides a safe way to call a script include from the sandbox, allowing only the inclusion of trusted scripts.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the script to include.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean| Flag that indicates whether the script include worked. Possible values:
|        | • true: Script include worked. |
|        | • false: Script include failed. |

This example gets the LDAPUtils in the current script context.

```javascript
var ldapServer = new GlideRecord("ldap_server_config");
ldapServer.addActiveQuery();
ldapServer.query();
gs.include("LDAPUtils");
var ldapUtils = new LDAPUtils();
var errMsg = "";
while (ldapServer.next()) {
   var ldap = new GlideLDAP();
   var dn = ldapServer.dn;
   var env = ldap.setup();
```
if (env == null) {
    errMsg = "Failed environment setup, missing URL";
    gs.eventQueue("ldap.connection_failed", ldapServer, ldapServer.getDisplayValue(), errMsg);
    gs.logError("LDAP server " + ldapServer.getDisplayValue() + " failed scheduled connection test: " + errMsg, "LDAP");
}

Scoped GlideSystem - info(String message, Object parm1, Object parm2, Object parm3, Object parm4, Object parm5)

Writes an info message to the system log.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The log message with place holders for any variable arguments.</td>
</tr>
<tr>
<td>param1</td>
<td>Object</td>
<td>(Optional) First variable argument.</td>
</tr>
<tr>
<td>param2</td>
<td>Object</td>
<td>(Optional) Second variable argument.</td>
</tr>
<tr>
<td>param3</td>
<td>Object</td>
<td>(Optional) Third variable argument.</td>
</tr>
<tr>
<td>param4</td>
<td>Object</td>
<td>(Optional) Fourth variable argument.</td>
</tr>
<tr>
<td>param5</td>
<td>Object</td>
<td>(Optional) Fifth variable argument.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

gs.info("This is an info message");
var myFirstName = "Abel";
var myLastName = "Tuter";
gs.info("This is an info message from {0}.{1}", myFirstName, myLastName);

Output:

This is an info message
This is an info message from Abel.Tuter
Scoped GlideSystem - isDebugging()

Determines if debugging is active for a specific scope.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if either session debugging is active or the log level is set to debug for the specified scope.</td>
</tr>
</tbody>
</table>

```javascript
gs.debug("This is a log message");
var myFirstName = "Abel";
var myLastName = "Tuter";
gs.debug("This is a log message from {0}.{1}", myFirstName, myLastName);
gs.info(gs.isDebugging());
```

Scoped GlideSystem - isInteractive()

Checks if the current session is interactive. An example of an interactive session is when a user logs in normally. An example of a non-interactive session is using a SOAP request to retrieve data.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the session is interactive.</td>
</tr>
</tbody>
</table>

```javascript
if (!gs.hasRole("admin") && gs.getSession().isInteractive()) {
    var qc1 = current.addQuery('u_group','');
    var gra = new GlideRecord('sys_user_grmember');
    gra.addQuery('user', gs.getUserID());
```
gra.query();
while (gra.next()) {
    qc1.addOrCondition('u_group', gra.group);
}

Scoped GlideSystem - isLoggedIn()
Determines if the current user is currently logged in.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the current user is logged in.</td>
</tr>
</tbody>
</table>

gs.info(gs.isLoggedIn());

Scoped GlideSystem - isMobile()
You can determine if a request comes from a mobile device.

This method can be used in UI action conditions and business rules.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the request comes from a mobile device; otherwise, false.</td>
</tr>
</tbody>
</table>

if (gs.isMobile())
    gs.info("submitted from mobile UI");
else
  gs.info("NOT submitted from mobile UI");

Scoped GlideSystem - monthsAgo(Number months)
Returns the date and time for a specified number of months ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>months</td>
<td>Number</td>
<td>Integer number of months</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT on today's date of the specified month, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - minutesAgoEnd(Number minutes)
Returns the date and time for the end of the minute a specified number of minutes ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>minutes</td>
<td>Number</td>
<td>Integer number of minutes</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - minutesAgoStart(Number minutes)
Returns the date and time for the start of the minute a specified number of minutes ago.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>minutes</td>
<td>Number</td>
<td>Integer number of minutes</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - monthsAgoStart(Number months)

Returns the date and time for the start of the month a specified number of months ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>months</td>
<td>Number</td>
<td>Integer number of months</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT start of the month the specified number of months ago, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - nil(Object o)

Queries an object and returns true if the object is null, undefined, or contains an empty string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>o</td>
<td>Object</td>
<td>The object to be checked.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the object is null, undefined, or contains an empty string; otherwise, returns false.</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord();
gs.info(gs.nil(now_GR));
```

Scoped GlideSystem - quartersAgoEnd(Number quarters)

Returns the date and time for the last day of the quarter for a specified number of quarters ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>quarters</td>
<td>Number</td>
<td>Integer number of quarters</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT end of the quarter that was the specified number of quarters ago, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - quartersAgoStart(Number quarters)

Returns the date and time for the first day of the quarter for a specified number of quarters ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>quarters</td>
<td>Number</td>
<td>Integer number of quarters</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT end of the month that was the specified number of quarters ago, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

**Scoped GlideSystem - setProperty(String key, String value, String description)**

Sets the specified key to the specified value if the property is within the script's scope.

**Note:** Care should be taken when setting system properties (sys_properties) using this method as it causes a system-wide cache flush. Each flush can cause system degradation while the caches rebuild. If a value must be updated often, it should not be stored as a system property. In general, you should only place values in the sys_properties table that do not frequently change.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>The key for the property to be set.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The value of the property to be set.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>A description of the property.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
gs.setProperty("glide.foo","bar","foo");
gs.info(gs.getProperty("glide.foo"));
```

Output: bar

**Scoped GlideSystem - setRedirect(Object o)**

Sets the redirect URI for this transaction, which then determines the next page the user will see.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>o</td>
<td>Object</td>
<td>URI object or URI string to set as the redirect</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
gs.setRedirect("com.glideapp.servicecatalog_cat_item_view.do?sysparm_id=d41ce5bac61227a0167f4bf8109bf70=sysparm_user=" + current.sys_id + "&sysparm_email=" + current.email)
```

### Scoped GlideSystem - tableExists(String name)

Determines if a database table exists.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the table to check for existence.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the table exists. False if the table was not found.</td>
</tr>
</tbody>
</table>

```java
gs.info(gs.tableExists('incident'));
```

### Scoped GlideSystem - urlEncode(String url)

Encodes non-ASCII characters, unsafe ASCII characters, and spaces so the returned string can be used on the Internet. Uses UTF-8 encoding. Uses percent (%) encoding.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>The string to be encoded.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A string with non-ASCII characters, unsafe ASCII characters, and spaces encoded.</td>
</tr>
</tbody>
</table>

**Scoped GlideSystem - urlDecode(String url)**

Replaces UTF-8 encoded characters with ASCII characters.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>A string with UTF-8 percent (%) encoded characters.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A string with encoded characters replaced with ASCII characters.</td>
</tr>
</tbody>
</table>

**Scoped GlideSystem - warn(String message, Object parm1, Object parm2, Object parm3, Object parm4, Object parm5)**

Writes a warning message to the system log.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The log message with place holders for any variable arguments.</td>
</tr>
<tr>
<td>param1</td>
<td>Object</td>
<td>(Optional) First variable argument.</td>
</tr>
<tr>
<td>param2</td>
<td>Object</td>
<td>(Optional) Second variable argument.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>param3</td>
<td>Object</td>
<td>(Optional) Third variable argument.</td>
</tr>
<tr>
<td>param4</td>
<td>Object</td>
<td>(Optional) Fourth variable argument.</td>
</tr>
<tr>
<td>param5</td>
<td>Object</td>
<td>(Optional) Fifth variable argument.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
gs.warn("This is a warning");
var myFirstName = "Abel";
var myLastName = "Tuter";
gs.warn("This is a warning from {0}.{1}", myFirstName, myLastName);
```

**Output:**

```
This is a warning
This is a warning from Abel.Tuter
```

### Scoped GlideSystem - xmlToJSON(String xmlString)

Takes an XML string and returns a JSON object.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xmlString</td>
<td>String</td>
<td>The XML string to be converted.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>A JSON object representing the XML string. Null if unable to process the XML string.</td>
</tr>
</tbody>
</table>

```javascript
var jsonObject = gs.xmlToJSON(xmlString);
```
Scoped GlideSystem - yearsAgo(Number years)

Returns a date and time for a certain number of years ago.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>years</td>
<td>Number</td>
<td>An integer number of years</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT beginning of the year that is the specified number of years ago, in the format yyyy-mm-dd hh:mm:ss.</td>
</tr>
</tbody>
</table>

Scoped GlideSystem - yesterday()

Returns yesterday's time (24 hours ago).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>GMT for 24 hours ago, in the format yyyy-mm-dd hh:mm:ss</td>
</tr>
</tbody>
</table>

GlideSystemRunLevel API - Global

Provides methods to get and set the current system run level.

Setting the system run level allows applications to change or limit their features based on operational toggles. For more information about using operational toggles, see Operational toggles.

The system run level can be set to one of the defined system run levels.

- Fully Operational
- Slightly Degraded
• Moderately Degraded
• Severely Degraded

The system run level set on one node is synchronized to all nodes in the instance. The admin role is required to use this API.

**GlideSystemRunLevel – getCurrentLevel()**

Retrieves the current system run level.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Gets the current system run level.

```javascript
var currentRunLevel = sn_run_level.GlideSystemRunLevel.getCurrentLevel();
gs.info(currentRunLevel);
```

Output:

```
fully_operational
```

**GlideSystemRunLevel – switchLevel(String newLevel)**

Switches the system run level.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>newLevel</td>
<td>String</td>
<td>The ID value of the new system run level. This value is from the ID field in the System Run Level [sys_run_level] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>ID of the previous system run level. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• fully_operational</td>
</tr>
<tr>
<td></td>
<td>• slightly_degraded</td>
</tr>
<tr>
<td></td>
<td>• moderately_degraded</td>
</tr>
<tr>
<td></td>
<td>• severely_degraded</td>
</tr>
</tbody>
</table>

Sets the system run level to Slightly Degraded.

```
sn_run_level.GlideSystemRunLevel.switchLevel('slightly_degraded');
```

Output:

```
SystemRunLevel requested to change from: fully_operational to: slightly_degraded
```

**GlideTableHierarchy - Scoped**

The Scoped GlideTableHierarchy API provides methods for handling information about table relationships.

**Scoped GlideTableHierarchy - getAllExtensions()**

Returns an array of strings containing all tables that extend the current table and includes the current table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

| Type   | Description                                                                
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of strings containing the tables in the hierarchy that includes</td>
</tr>
<tr>
<td></td>
<td>the current table.</td>
</tr>
</tbody>
</table>

var table = new GlideTableHierarchy("task");
gs.info(table.getAllExtensions());

Output: Line breaks added for clarity.

task, sc_task, problem_task, change_phase, sc_req_item, kb_submission, release_phase, problem, ticket, sm_task, hr_task, change_task, change_request, change_request_imac, incident, release_task, vtb_task, sm_order, hr_case, sysapproval_group, sc_request

**Scoped GlideTableHierarchy - getBase()**

Returns the parent class.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The parent class.</td>
</tr>
</tbody>
</table>

var table = new GlideTableHierarchy("cmdb_ci_server");
gs.info(table.getBase());

Output:

cmdb_ci_computer

**Scoped GlideTableHierarchy - getHierarchy()**

Returns an array of strings containing all classes in the hierarchy of the current table.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of strings of the classes in the hierarchy.</td>
</tr>
</tbody>
</table>

```javascript
var table = new GlideTableHierarchy("incident");
gs.info(table.getHierarchy());
```

Output:

```
incident,task
```

**Scoped GlideTableHierarchy - getName()**

Returns the table's name.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The table's name.</td>
</tr>
</tbody>
</table>

```javascript
var table = new GlideTableHierarchy("incident");
gs.info(table.getName());
```

Output:

```
incident
```

**Scoped GlideTableHierarchy - getRoot()**

Returns the top level class in the hierarchy.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the root class.</td>
</tr>
</tbody>
</table>

```javascript
var table = new GlideTableHierarchy("cmdb_ci_server");
gs.info(table.getRoot());
```

Output:

```javascript
cmdb
```

**Scoped GlideTableHierarchy - getTables()**

Returns an array of strings of the table names in the hierarchy.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of strings containing the names of tables in the hierarchy.</td>
</tr>
</tbody>
</table>

```javascript
var table = new GlideTableHierarchy("incident");
gs.info(table.getTables());
```

Output:

```javascript
incident, task
```

**Scoped GlideTableHierarchy - getTableExtensions()**

Returns an array of strings containing all tables that extend the current table.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of strings containing the tables that extend the current table.</td>
</tr>
</tbody>
</table>

```javascript
var table = new GlideTableHierarchy("task");
gs.info(table.getTableExtensions());
```

**Output:** Line breaks added for clarity

```
sc_task,problem_task,change_phase,sc_req_item,kb_submission,release_phase,problem, 
ticket,sm_task,hr_task,change_task,change_request,change_request_imac,incident, 
release_task,vtb_task,sm_order/hr_case/sysapproval_group/sc_request
```

### Scoped GlideTableHierarchy - GlideTableHierarchy(String tableName)

Instantiates a GlideTableHierarchy object.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>The name of the table.</td>
</tr>
</tbody>
</table>

```javascript
var table = new GlideTableHierarchy("incident");
gs.info(table.getTables());
```

**Output:**

```
incident,task
```

### Scoped GlideTableHierarchy - isBaseClass()

Returns true if this is a base class.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the current table has no parent and has extensions.</td>
</tr>
</tbody>
</table>

```javascript
var table = new GlideTableHierarchy("incident");
gs.info(table.isBaseClass());
```

Output:

false

Scoped GlideTableHierarchy - isSoloClass()

Returns true if this table is not in a hierarchy.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the current table has no parent and no extensions.</td>
</tr>
</tbody>
</table>

```javascript
var table = new GlideTableHierarchy("sys_user");
gs.info(table.isSoloClass());
```

Output:

true

Scoped GlideTableHierarchy - hasExtensions()

Returns true if this class has been extended.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the current table has extensions.</td>
</tr>
</tbody>
</table>

```javascript
var table = new GlideTableHierarchy("incident");
gs.info(table.hasExtensions());
```

Output:

```
false
```

**GlideTextReader - Scoped**

Provides the ability to read single lines from an input stream. Because an input stream is used, it is not subject to the 5MB attachment size limit.

**Scoped GlideTextReader - getEncoding()**

Returns the character encoding of the input stream.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The character encoding of the input stream.</td>
</tr>
</tbody>
</table>

**Scoped GlideTextReader - GlideTextReader(GlideScriptableInputStream inputStream)**

Creates a scoped GlideTextReader object for the specified input stream.
## GlideScriptableInputStream

The input stream to be read.

### GlideTextReader - readLine()

Returns a single line from the input stream and returns a string. Since this is working off of a stream, it is not subject to the 5MB size limit.

### GlideTime - Scoped

The scoped GlideTime class provides methods for performing operations on GlideTime objects, such as instantiating GlideTime objects or working with GlideTime fields.

### Scoped GlideTime - getByFormat(String format)

Returns the time in the specified format.

For information on the time formats that you can pass into this method, see Global date and time field format. You can pass these values in any order and add separators such as a dash, space, backslash, or colon. In addition, you can also pass the character "a" to return am/pm.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| format | String | String to use to format the time. For example, if the current time is 10:30:20 then:  
- hh:mm:ss = 10:30:20  
- HH = 10  
- HH — ss = 10 — 20  
- hh a = 10 AM  
- s:m:H ss:mm:hh = 20:30:10 20:30:10  
- HH:mm:ss.SSSZ = 10:30:20.000+0000  
- HH/mm/ss = 10/30/20 |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Time in the specified format.</td>
</tr>
</tbody>
</table>

```javascript
var gt = new GlideTime();
gt.setValue('12:00:00');
gs.info(gt.getByFormat("HH:mm"));
```

Output:

```
12:00
```

Scoped GlideTime - getDisplayValue()

Gets the time in the current user's display format and time zone.

When designing business rules or script includes remember that this method may return values in different formats for different users.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The time in the user's format and time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gt = new GlideTime();
gt.setDisplayValue("12:00:00"); // User Time Zone
gs.info(gt.getDisplayValue()); // User Time Zone
```

Scoped GlideTime - getDisplayValueInternal()

Gets the display value in the current user's time zone and the internal format (HH:mm:ss).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The time value for the GlideTime object in the current user's time zone and the internal time format of HH:mm:ss.</td>
</tr>
</tbody>
</table>

```javascript
var gt = new GlideTime();
gt.setValue("01:00:00"); //Internal Time Zone , UTC
gs.info(gt.getDisplayValueInternal()); //User Time Zone
```

Scoped GlideTime - getHourLocalTime()

Returns the hours part of the time using the local time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>Number</td>
<td>The hours using the local time zone.</td>
<td></td>
</tr>
</tbody>
</table>

**Scoped GlideTime - getHourOfDayLocalTime()**

Returns the hours part of the time using the local time zone. The number of hours is based on a 24 hour clock.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The hours using the local time zone. The number of hours is based on a 24 hour clock.</td>
</tr>
</tbody>
</table>

**Scoped GlideTime - getHourOfDayUTC()**

Returns the hours part of the time using the UTC time zone. The number of hours is based on a 24 hour clock.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The hours using the UTC time zone. The number of hours is based on a 24 hour clock.</td>
</tr>
</tbody>
</table>
**Scoped GlideTime - getHourUTC()**

Returns the hours part of the time using the UTC time zone. The number of hours is based on a 12 hour clock. Noon and midnight are represented by 0, not 12.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Number</td>
</tr>
</tbody>
</table>

**Scoped GlideTime - getMinutesLocalTime()**

Returns the number of minutes using the local time zone.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Number</td>
</tr>
</tbody>
</table>

**Scoped GlideTime - getMinutesUTC()**

Returns the number of minutes in the hour based on the UTC time zone.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Number</td>
</tr>
</tbody>
</table>
## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of minutes in the hour using the UTC time zone.</td>
</tr>
</tbody>
</table>

### Scoped GlideTime - getSeconds()

Returns the number of seconds in the current minute.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of seconds in the minute.</td>
</tr>
</tbody>
</table>

### Scoped GlideTime - getValue()

Gets the time value stored in the database by the GlideTime object in the internal format, HH:mm:ss, and the system time zone.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The time value in the internal format and system time zone.</td>
</tr>
</tbody>
</table>

```javascript
var gt = new GlideTime();
gs.info(gt.getValue()); // Internal Time Zone, UTC
```
**Scoped GlideTime - GlideTime()**

Instantiates a GlideTime object with the current time.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gt = new GlideTime();
gs.info(gt.getDisplayValue());
```

**Scoped GlideTime - GlideTime(Number milliseconds)**

Instantiates a GlideTime object with its time starting at midnight UTC plus the passed in value.

The value is adjusted for the timezone in which the instance resides. For example, if the user's instance is in a time zone that is -8 hours from the UTC, and 1000 is passed in, the time in the GlideTime object is 16:00:10 (00:00:00 - 8 hours +10 seconds.)

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>milliseconds</td>
<td>Number</td>
<td>Time to add to midnight UTC. Unit: Milliseconds</td>
</tr>
</tbody>
</table>

This example shows adding 10 seconds to midnight (UTC), which is then adjusted for the time zone in which the instance resides. In this case, -8 hours from UTC.

```javascript
var gt = new GlideTime(10000);
gs.info(gt.getDisplayValue());
```

**Output**

```
16:00:10
```

**Scoped GlideTime - setDisplayValue(String asDisplayed)**

Sets a time value using the current user's display format and time zone.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>asDisplayed</td>
<td>String</td>
<td>The time in the current user’s display format and time zone. The parameter must be formatted using the current user’s preferred display format, such as HH:mm:ss.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gt = new GlideTime();
gt.setDisplayValue('01:00:00');  // User Time Zone
gs.info(gt.getDisplayValueInternal());  // User Time Zone

**Scoped GlideTime - setValue(String o)**

Sets the time of the GlideTime object in the internal time zone.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>o</td>
<td>String</td>
<td>The time in hh:mm:ss format.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var gt = new GlideTime();
gt.setValue('01:00:00');  //Internal Time Zone, UTC
gs.info("time is "+ gt.getByFormat('hh:mm:ss'));

**Scoped GlideTime - subtract(GlideTime startTime, GlideTime endTime)**

Gets the duration difference between two GlideTime object values.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startTime</td>
<td>GlideTime</td>
<td>The start value.</td>
</tr>
<tr>
<td>endTime</td>
<td>GlideTime</td>
<td>The end value.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDuration</td>
<td>The duration between the two values.</td>
</tr>
</tbody>
</table>

```javascript
var gd1 = new GlideTime();
gd1.setDisplayValue("09:00:00");
var gd2 = new GlideTime();
gd2.setDisplayValue("09:10:00");

var dur = GlideDate.subtract(gd1, gd2); //the difference between gdt1 and gdt2
gs.info(dur.getDisplayValue());
```

### GlideTimeline - Global

The GlideTimeline class provides the core implementation for configuring and displaying a Glide Windowing Toolkit Timeline.

For security, the GlideTimeline has already been instantiated as a single instance variable named glideTimeline. All configurations should be made in the client script section of the corresponding schedule page that references this instance variable.

### GlideTimeline - groupByParent(Boolean b)

Specifies whether or not to group timeline items by their parent. If true, this will nest all child items inside their parent. This affects the ordering of display and children will always be listed immediately after their parent. The default value for the groupByParent property is false.

If true, this will nest all child items inside their parent. This affects the ordering of display and children will always be listed immediately after their parent. The default value for the groupByParent property is false.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>If true, displays Timeline Items grouped by parent.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
glideTimeline.groupByParent(true);
```

**GlideTimeline - registerEvent(String strServerEvent, String strScriptIncludeName)**

Registers the specified Timeline server event. The `strServerEvent` must be one of the allowed events for registration to work correctly. When the event occurs, the GlideTimeline sends a request to the server and processes the event as handled inside the `strScriptIncludeName` class.

The `strServerEvent` must be one of the allowed events for registration to work correctly. When the event occurs, the GlideTimeline sends a request to the server and processes the event as handled inside the `strScriptIncludeName` class.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strServerEvent</td>
<td>String</td>
<td>Specifies one of the following case-sensitive events:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• getItems</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• elementMoveX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• elementMoveY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• elementMoveXY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• elementSuccessor</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• elementTimeAdjustStart</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• elementTimeAdjustEnd</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• inputBox</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• itemMove</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strScriptIncludeName</td>
<td>String</td>
<td>Specifies the name of the class to receive the <code>strServerEvent</code>. This class must be defined in a script include that extends AbstractTimelineSchedulePage.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

`glideTimeline.registerEvent("getItems", "TimelineGanttSchedulePage");`

**GlideTimeline - setAutoRefresh(Number intSeconds)**

Specifies the number of seconds to wait before performing an auto refresh of the data on the timeline. Setting the number of seconds to 0 will turn auto refresh off. By default, auto refresh is disabled. If `intSeconds` is greater than 0 and less than the minimum allowed time in seconds (10), it will be set to 10 seconds.

Setting the number of seconds to 0 will turn auto refresh off. By default, auto refresh is disabled. If `intSeconds` is greater than 0 and less than the minimum allowed time in seconds (10), it will be set to 10 seconds.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>intSeconds</td>
<td>Number</td>
<td>An integer specifying the time in seconds between auto-refreshing.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

`glideTimeline.setAutoRefresh(15); // Sets the interval for auto-refreshing to 15 seconds.`
GlideTimeline - setDefaultPointIconClass(String icon_class)

Specifies the default icon class to use for Timeline Spans with zero duration if no icon class was explicitly specified in the properties of the Timeline Span returned from the server. The default value for the setDefaultPointIconClass property is milestone.

The default value for the setDefaultPointIconClass property is milestone.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>icon_class</td>
<td>String</td>
<td>String that specifies one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• milestone</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• blue_square</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• sepia_square</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• green_square</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• red_square</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• black_square</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• blue_circle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• sepia_circle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• green_circle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• red_circle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• black_circle</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
glideTimeline.setDefaultPointIconClass('blue_circle');
```

GlideTimeline - setExtraAjaxParam(String strName, String strValue)

Allows setting of additional parameters in the client script to be made available to the corresponding Script Include events by using the getParameter() method. URI parameters that are prefixed with sysparm_timeline_ will automatically be included in all server side AJAX calls.
URI parameters that are prefixed with `sysparm_timeline_` will automatically be included in all server side AJAX calls.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strName</td>
<td>String</td>
<td>The URI parameter name.</td>
</tr>
<tr>
<td>strValue</td>
<td>String</td>
<td>The value of strName.</td>
</tr>
</tbody>
</table>

**GlideTimeline - setInitialViewRange(String objStartDate, String objEndDate)**

Specifies the initial viewable range for the timeline. The format of the start and end dates must be in the default timestamp format: `yyyy-MM-dd HH:mm:ss`. The default range is the range that specifies the earliest Timeline Span point to the end of the latest Timeline Span. If the `initialViewRange` property is specified, it will override the default range.

The format of the start and end dates must be in the default timestamp format: `yyyy-MM-dd HH:mm:ss`. The default range is the range that specifies the earliest Timeline Span point to the end of the latest Timeline Span. If the `initialViewRange` property is specified, it will override the default range.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>objStartDate</td>
<td>String</td>
<td>The start time of the view range in format: <code>yyyy-MM-dd HH:mm:ss</code> .</td>
</tr>
<tr>
<td>objEndDate</td>
<td>String</td>
<td>The end time of the view range in format: <code>yyyy-MM-dd HH:mm:ss</code> .</td>
</tr>
</tbody>
</table>
### GlideTimeline - setInitialViewRange(Number objStartDate, Number objEndDate)

Specifies the initial viewable range for the timeline. The default range is the range that specifies the earliest Timeline Span point to the end of the latest Timeline Span. If the `initialViewRange` property is specified, it will override the default range.

The default range is the range that specifies the earliest Timeline Span point to the end of the latest Timeline Span. If the `initialViewRange` property is specified, it will override the default range.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>objStartDate</td>
<td>Number</td>
<td>The start time of the view range in milliseconds.</td>
</tr>
<tr>
<td>objEndDate</td>
<td>Number</td>
<td>The end time in milliseconds.</td>
</tr>
</tbody>
</table>

```javascript
// Sets the initial range to begin on June 20th, 2010 at 8:00 AM and end on June 28th, 2010 at 2:00 PM UTC time.
glideTimeline.setInitialViewRange("2010-06-20 08:00:00", "2010-06-28 14:00:00");
```

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
// Sets the initial range to begin on June 20th, 2010 at 8:00 AM and end on June 28th, 2010 at 2:00 PM UTC time.
glideTimeline.setInitialViewRange(1277046000000, 1277647200000);
```
GlideTimeline - setInitialViewRange(String objStartDate, Number objEndDate)

Specifies the initial viewable range for the timeline. The default range is the range that specifies the earliest Timeline Span point to the end of the latest Timeline Span. If the `initialViewRange` property is specified, it will override the default range.

The default range is the range that specifies the earliest Timeline Span point to the end of the latest Timeline Span. If the `initialViewRange` property is specified, it will override the default range.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>objStartDate</td>
<td>String</td>
<td>The start time of the view range in format: yyyy-MM-dd HH:mm:ss.</td>
</tr>
<tr>
<td>objEndDate</td>
<td>Number</td>
<td>The end time in milliseconds.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example Explanation

```javascript
// Sets the initial range to begin on June 20th, 2010 at 8:00 AM and end on June 28th, 2010 at 2:00 PM UTC time.
glideTimeline.setInitialViewRange("2010-06-20 08:00:00", 1277647200000);
```

GlideTimeline - setInitialViewRange(Number objStartDate, String objEndDate)

Specifies the initial viewable range for the timeline. The default range is the range that specifies the earliest Timeline Span point to the end of the latest Timeline Span. If the `initialViewRange` property is specified, it will override the default range.

The default range is the range that specifies the earliest Timeline Span point to the end of the latest Timeline Span. If the `initialViewRange` property is specified, it will override the default range.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>objStartDate</td>
<td>Number</td>
<td>The start time of the view range in milliseconds.</td>
</tr>
<tr>
<td>objEndDate</td>
<td>String</td>
<td>The end time of the view range in format: yyyy-MM-dd HH:mm:ss.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### Example Explanation

```java
// Sets the initial range to begin on June 20th, 2010 at 8:00 AM and end on June 28th, 2010 at 2:00 PM UTC time.
glideTimeline.setInitialViewRange(1277046000000, "2010-06-28 14:00:00");
```

### GlideTimeline - setReadOnly(Boolean b)

Enables or disables all timeline event interaction. If enabled, event interaction is determined from the corresponding attributes specified by each Timeline Item. The default value for the `readOnly` property is `false`.

If enabled, event interaction is determined from the corresponding attributes specified by each Timeline Item. The default value for the `readOnly` property is `false`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>If <code>true</code>, marks the entire timeline as <code>read-only</code> (non-interactive).</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
glideTimeline.setReadOnly(true);
```
GlideTimeline - showDependencyLines(Boolean b)

Specifies whether or not to show dependency lines between Timeline Spans. This method applies only if the set of Timeline Items returned from the server includes dependency relationships. The default value for the showDependencyLines property is **false**.

This method applies only if the set of Timeline Items returned from the server includes dependency relationships. The default value for the showDependencyLines property is **false**.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>If <strong>true</strong>, displays dependency lines on the timeline; otherwise, does not.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
glideTimeline.showDependencyLines(true);
```

GlideTimeline - showGridLines(Boolean bShowGridlines, Number amount)

Specifies whether or not to show grid lines for each row of data on the timeline. By default, grid lines are enabled.

By default, grid lines are enabled.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bShowGridlines</td>
<td>Boolean</td>
<td>If <strong>true</strong>, timeline shows grid lines; otherwise, does not display grid lines.</td>
</tr>
<tr>
<td>amount</td>
<td>Number</td>
<td>amount</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td>ReturnValue</td>
</tr>
</tbody>
</table>

```
glideTimeline.showGridLines(false); // Disables grid lines.
```

**GlideTimeline - showLeftPane(Boolean b)**

Specifies whether or not to show the left hand pane in the timeline. The default value for the `leftPane` property is `true`.

The default value for the `leftPane` property is `true`.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>If <code>true</code>, the timeline shows the left pane; otherwise, the left pane is not be displayed.</td>
</tr>
</tbody>
</table>

```
glideTimeline.showLeftPane(false);
```

**GlideTimeline - showLeftPaneAsTree(Boolean b)**

Specifies how to show child items in the left pane of the timeline. Child items are displayed either as nested, indented nodes with expand and collapse capability or on a single indent level. The default value for the `showLeftPaneAsTree` property is `false`.

The default value for the `showLeftPaneAsTree` property is `false`. 
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>If true, child item nodes are shown as indented with expand/collapse capability; otherwise, all left pane items are displayed at a single indent level.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
glideTimeline.showLeftPaneAsTree(true);```

**GlideTimeline - showLeftPaneInputBox(Boolean b, String strDefaultValue)**

Specifies whether or not to show the text input box at the bottom of the left pane with a default value as specified by `strDefaultValue`. If the left pane is disabled via `showLeftPane()`, the input box will not be visible. The default value for the `showLeftPaneInputBox` property is `false`.

If the left pane is disabled via `showLeftPane()`, the input box will not be visible. The default value for the `showLeftPaneInputBox` property is `false`.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>If true, show the left pane input box.</td>
</tr>
<tr>
<td>strDefaultValue</td>
<td>String</td>
<td>The default value to display in the input box.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
glideTimeline.showLeftPaneInputBox(true, 'Add a new task ...');```
GlideTimeline - showSummaryPane(Boolean b)

Specifies whether or not to show the summary pane at the bottom of the timeline. The default value for the showSummaryPane property is true.

The default value for the showSummaryPane property is true.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>If true, the timeline includes the summary pane; otherwise, the summary pane is not displayed.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
glideTimeline.showSummaryPane(false);
```

GlideTimeline - showTimelineText(Boolean b)

Specifies whether or not to show the timeline text underneath each Timeline Span in the primary timeline pane. The default value for the showTimelineText property is false.

The default value for the showTimelineText property is false.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>If true, displays descriptive text underneath each Timeline Span; otherwise, no text is displayed underneath each Timeline Span.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
glideTimeline.showTimelineText(true);
```
GlideTimeline - snapVertScrollingIntoRows(Boolean b)

Specifies whether or not the vertical movement of timeline span objects (if appropriately registered to perform this event) should snap adjust into the closest row. By default this value is enabled.

By default this value is enabled.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td>b</td>
<td>Boolean</td>
</tr>
</tbody>
</table>

GlideTimeline - sortByLeftLabelText(Boolean b)

Specifies whether or not to group items by their left label text. The default value for the sortByLeftLabelText property is false.

The default value for the sortByLeftLabelText property is false.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td>b</td>
<td>Boolean</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

glideTimeline.snapVertScrollingIntoRows(false);

GlideTimeline - sortByLeftLabelText(Boolean b)

Specifies whether or not to group items by their left label text. The default value for the sortByLeftLabelText property is false.

The default value for the sortByLeftLabelText property is false.
glideTimeline.sortByLeftLabelText(true);

**GlideTimeline - sortByStartDate(Boolean b)**

Specifies whether or not to sort the list of Timeline Items returned by the earliest start date of an item's Timeline Span objects. If `groupByParent()` is set `true`, items are sorted after grouping has occurred. The default value for the `sortByStartDate` property is `false`.

If `groupByParent()` is set `true`, items are sorted after grouping has occurred. The default value for the `sortByStartDate` property is `false`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>If <code>true</code>, sort Timeline Items chronologically starting with their earliest start date.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

glideTimeline.sortByStartDate(true);

**GlideTimeline - sortByTimelineLabelText(Boolean b)**

Specifies whether or not to sort the list of Timeline Items returned in alphabetical order according to the text that was specified to show in the Timeline Pane.

⚠️ **Note:** This sort order still applies even if the timeline text has been set `false` via the `showTimelineText()` method. Additionally, if `groupByParent()` is set `true`, items will be sorted appropriately after grouping has occurred.

The default value for the `sortByTimelineLabelText` property is `false`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>If <code>true</code>, sorts Timeline Items alphabetically by the text specified in each item's timeline span text.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
glideTimeline.sortByTimelineLabelText(true);
```

**GlideTimelineItem - Global**

`GlideTimelineItem` extends the abstract `ScheduleItem` class to define additional properties that are specific to the time line.

A time line item is essentially any item that is displayed in a singular row across the time line. A `GlideTimelineItem` has zero or more associated spans (`TimelineSpan` objects).

**GlideTimelineItem - createTimelineSpan(String tableName)**

Creates a new `TimelineSpan` object associated with the current instance object.

If no other `TimelineSpan` objects exist, the newly created object will share the same `sys_id` as current instance object. Otherwise, a randomly generated GUID will be used.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>The name of the table associated with current object.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The newly-created span object instance.</td>
</tr>
</tbody>
</table>

**GlideTimelineItem - createTimelineSpan(String tableName, String sys_id)**

Creates a new `TimelineSpan` object associated with the current instance object using the specified table and `sys_id`.
### GlideTimelineItem - getImage()

Returns a string specifying the name of the image file associated with the current GlideTimelineItem.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the image file associated with the current GlideTimelineItem. If no image is associated with the current item, an empty string (&quot;&quot;) is returned.</td>
</tr>
</tbody>
</table>

### GlideTimelineItem - getIsDroppable()

Indicates whether or not the current instance object should be allowed as a "drop zone" when moving timeline elements vertically.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### GlideTimelineItem - getImage()

Returns a string specifying the name of the image file associated with the current GlideTimelineItem.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the image file associated with the current GlideTimelineItem. If no image is associated with the current item, an empty string (&quot;&quot;) is returned.</td>
</tr>
<tr>
<td>Returns</td>
<td></td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>Boolean</td>
<td>True if droppable; false otherwise.</td>
</tr>
</tbody>
</table>

**GlideTimelineItem - getLeftLabelText( )**

Returns the text to be displayed in the left pane (if enabled).

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The value of the text to be displayed in the left pane.</td>
</tr>
</tbody>
</table>

**GlideTimelineItem - getParent( )**

Returns the unique `sysId` of the current GlideTimelineItem's parent object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The unique sysId of the current GlideTimelineItem's parent object. If the parent does not exist, this will return an empty string (&quot;&quot;).</td>
</tr>
</tbody>
</table>

**GlideTimelineItem - getTimelineSpans( )**

Returns all the `TimelineSpan` objects associated with the current instance in an ArrayList.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The list of TimelineSpan objects associated with the current instance.</td>
</tr>
<tr>
<td>Array</td>
<td></td>
</tr>
</tbody>
</table>

GlideTimelineItem - GlideTimelineItem(String tableName)

Create a "dummy" GlideTimelineItem object.

This is useful for creating rows that do not allow any YMoving into; however, contain nested children (e.g. The top-level "Users" row in the Group Resource Timeline). The sys_id needs to be unique for DOM level functions to parse correctly. By default this object will not be "droppable" because a table name was not specified.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>The name of the table associated with current object.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideTimelineItem - GlideTimelineItem(String tableName, String sys_id)

Constructor that sets the required table and sys_id properties.

The rest of this object’s properties should be set by the caller. By default, this object instance is "droppable" since a table name is specified.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>The name of the table associated with current object.</td>
</tr>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The sys ID for the object.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

GlideTimelineItem - isTextBold()

Indicates if the left pane text is set to be displayed using a bold style.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the text should be bolded; otherwise false.</td>
</tr>
</tbody>
</table>

GlideTimelineItem - setImage(String strImageName)

Sets the name of the image file (including it's path) to use as the icon for the item in the left pane.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strImageName</td>
<td>String</td>
<td>The name of the image, including its path.</td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**GlideTimelineItem - setIsDraggable(Boolean b)**

Sets whether or not the current instance object can be clicked and dragged into another GlideTimelineItem.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>b</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

**GlideTimelineItem - setLeftLabelText(String strText)**

Specifies the text to display in the left pane for this item.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>strText</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

**GlideTimelineItem - setParent(String sysId)**

Sets the parent of the current GlideTimelineItem.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysId</td>
<td>String</td>
<td>The sysID of the GlideTimelineItem that should become the parent of the current GlideTimelineItem.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideTimelineItem - setTextBold(Boolean b)**

Specifies whether or not to bold the text style of the item in the left pane.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Boolean</td>
<td>True if text in left pane should be bolded; otherwise false.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**GlideTransformLog - Scoped, Global**

Creates a GlideTransformLog object to log messages to localhost logs.

**Related information**

- GlideImportLog
- GlideImportSetRun
- GlideImportSetTable
- GlideImportSetTransformer
- GlideImportSetTransformMap

**GlideTransformLog - error(String message)**

Logs a message of type Error to localhost logs.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Transform log message.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importLog = new GlideTransformLog();
importLog.error('Error executing transform');
```

#### GlideTransformLog - GlideTransformLog()

Instantiates an GlideTransformLog object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importLog = new GlideTransformLog();
```

#### GlideTransformLog - info(String message)

Logs a message of type Info to localhost logs.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Transform log message.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
```javascript
var importLog = new GlideTransformLog();
importLog.info('Successfully executed the transform.');
```

**GlideTransformLog - warn(String message)**

Logs a message of type Warn to localhost logs.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Transform log message.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var importLog = new GlideTransformLog();
importLog.warn('Transform taking longer than expected');
```

**GlideUICompatibility - Scoped**

The scoped GlideUICompatibility class provides the ability for scoped applications to define their own minimum browser versions. This is done by creating system properties for the scoped application.

You create the properties using the sys_properties list and assign a version number. When you do this from the scoped application, the `<scope-name>` prefix is automatically added to the property name. The scoped application UI compatibility properties are:

- `<scope-name>.ui.ie_minimum`
- `<scope-name>.ui.chrome_minimum`
- `<scope-name>.ui.firefox_minimum`
- `<scope-name>.ui.safari_major_version_minimum`

You can then use the scoped GlideUICompatibility class to determine if the current browser is supported.

**Scoped GlideUICompatibility - getCompatibility()**

Returns the terms "block" or "allow" based upon the browser version.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Either block or allow</td>
</tr>
</tbody>
</table>

```
UICompatibility = new GlideUICompatibility(gs.getCurrentScopeName());
var blockOrAllow = UICompatibility.getCompatibility();
gs.info(blockOrAllow);
```

Output: allow

**Scoped GlideUICompatibility - GlideUICompatibility(String scopeName)**

Creates a GlideUICompatibility object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopeName</td>
<td>String</td>
<td>The application's scope name</td>
</tr>
</tbody>
</table>

**Scoped GlideUICompatibility - isBlocked()**

Determines if the browser is not supported.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the browser is not supported.</td>
</tr>
</tbody>
</table>
UICompatibility = new GlideUICompatibility(gs.getCurrentScopeName());
var blocked = UICompatibility.isBlocked();
gs.info(blocked);

Output: false

**GlideUIScripts - Client**
Access UI scripts from within client-side code.

There is no constructor for this class. Access methods using the g_ui_scripts global object in any client-side code, such as client or validation scripts.

If calling a UI script with UI Type set to Mobile / Service Portal, use the g_ui_scripts['nameOfScript'] syntax. If calling a UI script with the UI Type set to All or Desktop, use the getUIScript() method to load the script. However, this method is not supported in Internet Explorer 11 when called outside of the Angular application environment. If calling a UI script outside of an Angular context using IE11, you must call the script directly.

⚠ **Note:** This class does not support UI scripts with the Global field set to true.

**GlideUIScripts - getUIScript(String scriptName)**
Calls a UI script with the UI Type set to All or Desktop from a client script or other client-side code. Returns a promise.

Use the then() function to perform an asynchronous action after the call resolves.

⚠ **Note:** This method is not supported in Internet Explorer 11 when called outside of the Angular application environment. If calling a UI script outside of an Angular context using IE11, call the script directly using the g_ui_scripts['nameOfScript'] syntax.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scriptName</td>
<td>String</td>
<td>API name of the UI script to run.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Promise</td>
<td>The result of the asynchronous call.</td>
</tr>
</tbody>
</table>
function onLoad() {
  // Call the UI script directly if the UI Type is Mobile / Service Portal, for example:
  // g_ui_scripts['myUIScript'];

  // Use the method if the UI Type is All or Desktop
  g_ui_scripts.getUIScript('myUIScript').then(function(script) {
    script.myUIScriptMethod();
  }, function() {
    console.log('The script did not load');
  });
}

**GlideURI - Global**

Enables handling the URI parameter.

For more information on using URIs:

- Navigate by URL
- Navigation stack

See also `action - getGlideURI()`.

**GlideURI - deleteMatchingParameter(String match)**

Deletes one or more parameters from the URI that match the beginning of the provided string.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>match</td>
<td>String</td>
<td>Partial name of one or more parameters to remove from the query portion of a URI.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to delete URI parameters that partially match the name provided.

```javascript
// create the GlideURI object
var uri = action.getGlideURI();
```
uri.deleteMatchingParameter('sysparm_list_');
uri.deleteMatchingParameter('sysparm_record_');

**Scoped equivalent**
This method is not available in scoped applications.

**GlideURI - deleteParameter(String name)**
Removes a specified parameter from the query portion of the URI.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>name</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to delete a URI parameter.

```java
// create the GlideURI object
var uri = action.getGlideURI();

uri.deleteParameter('sysparm_referring_url');
```

**Scoped equivalent**
This method is not available in scoped applications.

**GlideURI - get(String name)**
Returns the value of the specified parameter.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>name</td>
</tr>
</tbody>
</table>
## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The value for the specified parameter.</td>
</tr>
</tbody>
</table>

```javascript
// create the GlideURI object
var uri = action.getGlideURI();
uri.set('sysparm_query', 'priority=2^active=true');
var fileString = uri.get('sysparm_query');
gs.info(fileString);
```

### Output:

```
priority=2^active=true
```

### Scoped equivalent

To use the `get()` method in a scoped application, use the corresponding scoped method: **GlideURI - `get(String name)`**.

### GlideURI - `getFileFromPath()`

Returns the file name portion of the URI.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gURI = action.getGlideURI();
var fileString = gURI.getFileFromPath();
gs.info(fileString);
```

### Scoped equivalent

To use the `getFileFromPath()` method in a scoped application, use the corresponding scoped method: **GlideURI - `getFileFromPath()`**.
**GlideURI - getMap()**

Returns a map (key value pairs) containing each parameter in the query and its associated value.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to get `sysparm_query` parameters. See also `action - getGlideURI()`.

```java
gs.action.getGlideURI().getMap().get('sysparm_query');
```

**Scoped equivalent**

This method is not available in scoped applications.

**GlideURI - set(String name, String value)**

Sets the specified parameter to the specified value.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The parameter name.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The value.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set value of a `sysparm_query` field. See also `action - getGlideURI()`.
```javascript
var gURI = action.getGlideURI();
gURI.set('sysparm_query', 'priority=2^active=true');
var fileString = gURI.get('sysparm_query');
gs.info(fileString);
```

Output:

```
priority=2^active=true
```

**Scoped equivalent**

To use the `set()` method in a scoped application, use the corresponding scoped method: GlideURI - `set(String name, String value)`.

**GlideURI - setView(String view)**

Adds the `sysparm_view` parameter to the query with the named view.

A view defines the elements that appear when a user opens a form or a list. The `sysparm_view` parameter specifies the view to be used for a list or a form. For more information on views, see View management.

See also:

- action - `getGlideURI()`
- GlideModalV3 - `switchView(String newView)`

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>view</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

The following example shows how to add the major incidents view to the URI query in the Incidents [incident] table. For example, `sysparm_view=Major%20Incidents`. See also action - `getGlideURI()`.

```javascript
// create the GlideURI object
var uri = action.getGlideURI();
```
// Adds the Major incidents to the query
uri.setView('Major Incidents');

// https://instance.service-now.com/incident.do?sys_id=0&sysparm_view=Major%20Incidents ...

Scoped equivalent
This method is not available in scoped applications.

GlideURI - toString(String path)
Reconstructs the URI string and performs the proper URL encoding by converting non-valid characters to their URL code. For example, converting & to '%26'.

Parameters set with the `set()` method are encoded with the URI as well.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>String</td>
<td>The base portion of the system URL to which the URI is appended.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The URL.</td>
</tr>
</tbody>
</table>

The following examples shows how to convert invalid characters to URL code in an instance URL. See also `action - getGlideURI()`.

```javascript
var gURI = action.getGlideURI();
fileString = gURI.toString('https://<your instance>.service-now.com/navpage.do');
```

Scoped equivalent
To use the `toString()` method in a scoped application, use the corresponding scoped method: GlideURI - toString(String path).

GlideURI - Scoped
Enables handling the URI parameter in scoped applications.

For more information on using URIs:

- Navigate by URL
- Navigation stack
See also: GlideURI - Global.

**GlideURI - GlideURI()**

Instantiates a GlideURI object.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

**GlideURI - get(String name)**

Returns the value of the specified parameter.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

```javascript
var gURI = new GlideURI();
gURI.set('sysparm_query', 'priority=2^active=true' );
var fileString = gURI.get('sysparm_query');
gs.info(fileString);
```

Output:

```
priority=2^active=true
```

**GlideURI - getFileFromPath()**

Returns the file name portion of the URI.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The file name portion of the URI.</td>
</tr>
</tbody>
</table>

```javascript
var gURI = new GlideURI();
var fileString = gURI.getFileFromPath();
gs.info(fileString);
```

---

**GlideURI - set(String name, String value)**

Sets the specified parameter to the specified value.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The parameter name.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The value.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var gURI = new GlideURI();
gURI.set('sysparm_query', 'priority=2^active=true' );
var fileString = gURI.get('sysparm_query');
gs.info(fileString);
```

### Output:

```javascript
priority=2^active=true
```
**GlideURI - toString(String path)**

Reconstructs the URI string and performs the proper URL encoding by converting non-valid characters to their URL code. For example, converting `&` to `%26`.

Parameters set with the `set()` method are encoded with the URI as well.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>String</td>
<td>The base portion of the system URL to which the URI is appended.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The URL.</td>
</tr>
</tbody>
</table>

```javascript
var gURI = new GlideURI();
fileString = gURI.toString('https://<your instance>.service-now.com/navpage.do');
```

**GlideURLV3 - Client**

Provides methods for manipulating a URI.

The GlideURLV3 API can be used in client-side scripts using ListV2 and ListV3 APIs.

ℹ️ **Note:** This API is not supported by Service Portal, Now Mobile, or Agent Workspace.

**GlideURLV3 - addParam(String name, String value)**

Adds a query string name-value pair to the URL.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the query string parameter.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Query string value.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The GlideURL</td>
</tr>
</tbody>
</table>

```javascript
var gu = new GlideURL('incident.do');
var url = gu.addParam('sys_id', '-1');
```

**GlideURLV3 - getURL(Object additionalParams)**

Get the entire context path and query string parameters as a single URI.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalParams</td>
<td>Object</td>
<td>A name-value pair object that contains parameters that are added to this URL request only. These additional parameters are not saved to the GlideURL object.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The GlideURL with the specified additional parameters added to the end.</td>
</tr>
</tbody>
</table>

**GlideURLV3 - GlideURL(String contextPath)**

Creates an instance of the GlideURL class.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextPath</td>
<td>String</td>
<td>A relative path for the URL.</td>
</tr>
</tbody>
</table>

**GlideUser - Client**

The GlideUser API provides access to information about the current user and current user roles. Using the GlideUser API avoids the need to use the slower GlideRecord queries to get user information.
The GlideUser methods and properties are accessed through a global object (g_user) that is only available in client scripts. GlideUser

- contains name and role information about the current user.
- is typically used in client scripts and UI policies but is also found in UI actions that run on the client.
- cannot be used in business rules or UI actions that run on the server.
- avoids the need for GlideRecord queries to get user information.

Session information about the current user and current user roles is contained in the client (web browser). All GlideUser methods except `getClientData()` access the session information that is available by default. The `getClientData()` method requires setup on the server and use of `putClientData()` to make session information available.

**GlideUser - firstName**

Returns the current user's first name.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>firstName</td>
<td>String</td>
<td>Current user's first name.</td>
</tr>
</tbody>
</table>

```javascript
alert('first name = ' + g_user.firstName);
```

**GlideUser - getClientData(String key)**

Returns a client value set using `setClientData()` or GlideSession -- `putClientData()`.

Session client data is a set of named strings that may be setup on the server using GlideSession -- `putClientData()`. You can use `getClientData()` during form load time to get information that the client script needs to make decisions about the form. For example, to identify which fields should be visible.

See also **GlideForm**.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the client data to retrieve.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the client data.</td>
</tr>
</tbody>
</table>

```javascript
var loginLanguage = g_user.getClientData("loginlanguage");
```

### GlideUser - getFullName()

Returns the first and last name of the current user.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The current user's full name.</td>
</tr>
</tbody>
</table>

```javascript
var formalName = g_user.getFullName();
```

### GlideUser - hasRole(String role, Boolean includeDefaults)

Returns true if the current user has the specified role or the admin role.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>role</td>
<td>String</td>
<td>Role to check.</td>
</tr>
<tr>
<td>includeDefaults</td>
<td>Boolean</td>
<td>Optional. Flag that indicates whether to include default roles, such as snc_internal and snc_external,</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>includeDefaults</td>
<td>Boolean</td>
<td>Optional. Flag that indicates whether to include default roles, such as snc_internal and snc_external, in the request. For additional information on roles, see Explicit roles. Default: false</td>
</tr>
<tr>
<td>role</td>
<td>String</td>
<td>Role to check.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the current user has the specified role.</td>
</tr>
</tbody>
</table>

```
var isInternal = g_user.hasRole('snc_internal', true);
```

```
var isItil = g_user.hasRole('itil');
```

GlideUser - `hasRoleExactly(String role, Boolean includeDefaults)`

Determines whether the current user has the specified role.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>includeDefaults</td>
<td>Boolean</td>
<td>Optional. Flag that indicates whether to include default roles, such as snc_internal and snc_external, in the request. For additional information on roles, see Explicit roles. Default: false</td>
</tr>
<tr>
<td>role</td>
<td>String</td>
<td>Role to check.</td>
</tr>
</tbody>
</table>
var isInternal = g_user.hasRoleExactly('snc_internal', true);

var isItil = g_user.hasRoleExactly('itil');

GlideUser - hasRoleFromList(String roles, Boolean includeDefaults)

Returns true if the current user has at least one of the specified roles or has the admin role.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>roles</td>
<td>String</td>
<td>Comma-separated list of roles to check</td>
</tr>
<tr>
<td>includeDefaults</td>
<td>Boolean</td>
<td>Optional. Flag that indicates whether to include default roles, such as snc_internal and snc_external, in the request. For additional information on roles, see Roles. Default: false</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the current user has a role in the list or the admin role.</td>
</tr>
</tbody>
</table>

var isOK = g_user.hasRoleFromList("itil, maint");

var isOK = g_user.hasRoleFromList("itil, maint, snc_internal", true);

GlideUser - hasRoles(Boolean includeDefaults)

Returns true if the current user has any role.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>includeDefaults</td>
<td>Boolean</td>
<td>Optional. Flag that indicates whether to include default roles, such as snc_internal and snc_external,</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>in the request. For additional information on roles, see Roles. Default: false</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the current user has at least one role.</td>
</tr>
</tbody>
</table>

```javascript
var yesRole = g_user.hasRoles();

var yesRole = g_user.hasRoles(true);
```

**GlideUser - lastName**

The current user's last name.

```javascript
alert('last name = ' + g_user.lastName);
```

**GlideUser - setClientData(String key, String value)**

Sets a client value that you can retrieve using `getClientData()`.

See also GlideForm.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the client data to store as a key.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Number</td>
<td>Value to assign to the key.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

```javascript
function onSubmit() {
  if (!g_user.getClientData('keyName')) {
    var now_GR = new GlideRecord('incident');
    now_GR.addActiveQuery();
    now_GR.setLimit(1);
    now_GR.query(cb);
    return false;
  }
  return true;
}

function cb(now_GR) {
  // <insert glide operation >
  g_user.setClientData('keyName', now_GR.getValue('<number>'));
  g_form.submit();
}
```

**GlideUser - userID**

Returns the sys_id of the current user.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userID</td>
<td>String</td>
<td>sys_id of the current user.</td>
</tr>
</tbody>
</table>

```javascript
var userID = g_user.userID;
alert('Current user ID = ' + userID);
```
GlideUser - userName

This property is the current user's username, for example gsmith02. It is not the user's name, for example George Smith.

<table>
<thead>
<tr>
<th>Field</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userName</td>
<td>String</td>
<td></td>
<td>Current user's username.</td>
</tr>
</tbody>
</table>

```javascript
var userName = g_user.userName;
alert('Current user = ' + userName);
```

GlideUser - Scoped

The scoped GlideUser API provides access to information about the current user and current user roles. Using the scoped GlideUser API avoids the need to use the slower GlideRecord queries to get user information.

Scoped GlideUser - getCompanyID()

Returns the current user's company sys_id.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Company sys_id</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getCompanyID());
```

Scoped GlideUser - getDisplayName()

Returns the current user's display name.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>User's display name</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getDisplayName());
```

### Scoped GlideUser - getDomainID()

Returns the identifier of the user's current session domain.

The identifier that is returned depends on the domain type and the instantiation of that domain.

- If the user is configured in the global domain, and does not use the domain picker to switch domains, the method returns null.
- If the user uses the domain picker to switch to the global domain, the method returns the string "global".
- For all other domains, the method returns the sys_id of that domain.

```javascript
var domain = new GlideRecord('domain');
domain.get(gs.getUser().getDomainID());
gs.info(domain.name);
```
**Scoped GlideUser - getEmail()**

Returns the user's email address.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>User's email address</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getEmail());
```

**Scoped GlideUser - getFirstName()**

Returns the user's first name.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>User's first name</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getFirstName());
```

**Scoped GlideUser - getID()**

Gets the sys_id of the current user.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>User's sys_id</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getID());
```

**Scoped GlideUser - getLastName()**

Returns the user's last name.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>User's last name</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getLastName());
```

**Scoped GlideUser - getName()**

Returns the user ID, or login name, of the current user.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>User ID</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getName());
```

**Scoped GlideUser - getPreference(String name)**

Get the specified user preference value for the current user.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the preference.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The preference value.</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
currentUser.savePreference('myPref','red');
gs.info(currentUser.getPreference('myPref'));
```

**Scoped GlideUser - getRoles()**

Returns a list of roles that includes explicitly granted roles, inherited roles, and roles acquired by group membership.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of all roles available to the user</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getRoles());
```

Scoped GlideUser - `getUserRoles()`

Returns the list of roles explicitly granted to the user.

Unlike the `getRoles()` method, this method does not return roles the user inherits or roles acquired from group membership.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of roles explicitly assigned to the user</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getUserRoles());
```

Scoped GlideUser - `hasRole(String role)`

Determines if the current user has the specified role.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>role</td>
<td>String</td>
<td>Role to check</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the user has the role.</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.hasRole('admin'));
```

Scoped GlideUser - isMemberOf(String group)

Determines if the current user is a member of the specified group.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group</td>
<td>String</td>
<td>Group to check</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the user is a member of the group.</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.isMemberOf('Capacity Mgmt'));
```

Scoped GlideUser - savePreference(String name, String value)

Saves a user preference value to the database.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The preference to save.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The preference value.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
currentUser.savePreference('myPref','red');
gs.info(currentUser.getPreference('myPref'));
```

**GlideUser - Global**

The GlideUser API provides access to information about the current user and current user roles.

Using the GlideUser API avoids the need to use the slower GlideRecord queries to obtain user information.

**GlideUser - getCompanyID()**

Returns the current user's company sys_id.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Company sys_id</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getCompanyID());
```

**Scoped equivalent**

To use the `getCompanyID()` method in a scoped application, use the corresponding scoped method: **Scoped GlideUser - getCompanyID()**.

**GlideUser - getDisplayName()**

Returns the current user's display name.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>User's display name</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getDisplayName());
```

**Scoped equivalent**

To use the `getDisplayName()` method in a scoped application, use the corresponding scoped method: **Scoped GlideUser - getDisplayName()**.

**GlideUser - getDomainDisplayValue()**

Returns the display value of the user's session domain.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The display value of the user's session domain.</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getDomainDisplayValue());
```

**Scoped equivalent**

There is no workaround for scoped applications.
**GlideUser - getEmail()**

Returns the user's email address.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>User's email address</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getEmail());
```

**Scoped equivalent**

To use the `getEmail()` method in a scoped application, use the corresponding scoped method: **Scoped GlideUser - getEmail().**

**GlideUser - getFirstName()**

Returns the user's first name.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object/String</td>
<td>User's first name.</td>
</tr>
</tbody>
</table>

**Note:** The data type for the returned value is object, however, the information is returned as a string.

```javascript
var currentUser = gs.getUser();
var userName = currentUser.getFirstName();
```
Scoped equivalent

To use the `getFirstName()` method in a scoped application, use the corresponding scoped method: `Scoped GlideUser - getFirstName()`.

**GlideUser - getID**

Returns the sys_id of the current user.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getID());
```

Scoped equivalent

To use the `getID()` method in a scoped application, use the corresponding scoped method: `Scoped GlideUser - getID()`.

**GlideUser - getLastName()**

Returns the user's last name.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
<tr>
<td>Type</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>Object/</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

**Note:** The data type for the returned value is object, however, the information is returned as a string.

```javascript
var currentUser = gs.getUser();
var userName = currentUser.getLastName();
gs.info('User Last Name: ' + userName);
gs.info('Data type: ' + typeof userName);
```

*** Script: User Last Name: Smith
*** Script: Data type: object

### Scoped equivalent

To use the `getLastName()` method in a scoped application, use the corresponding scoped method: `Scoped GlideUser - getLastName()`.

### GlideUser - getMyGroups()

Returns an iterator containing the list of all groups to which the user belongs. Only active groups are returned.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iterator</td>
<td>A list of sys_ids for the active groups to which the user belongs.</td>
</tr>
</tbody>
</table>

**An example**

```javascript
var groupsArray = gs.getUser().getMyGroups().toArray();
gs.info(groupsArray[0]);
```

Output: cfcbad03d711110050f5edcb9e61038f

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Scoped equivalent
There is no scoped equivalent for this method.

GlideUser - getName()
Returns the user ID, or login name, of the current user.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>User ID</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getName());
```

Scoped equivalent
To use the `getName()` method in a scoped application, use the corresponding scoped method: `Scoped GlideUser - getName()`.

GlideUser - getRoles()
Returns a list of roles associated with the user. Includes explicitly granted roles, inherited roles, and roles acquired by group membership.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Comma-separated list of user roles.</td>
</tr>
</tbody>
</table>
var currentUser = gs.getUser();
gs.info(currentUser.getRoles());

Output:

admin, hr_fulfiller, itsa_fulfiller, security_admin

Scoped equivalent

To use the `getRoles()` method in a scoped application, use the corresponding scoped method: `Scoped GlideUser - getRoles()`.

GlideUser - getUserByID (String id)

Returns the user object associated with the passed-in user ID (sys_id in sys_user) or user_name.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Unique ID (sys_id) or user_name of the desired user record.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>User object associated with the specified sys_id or user_name.</td>
</tr>
</tbody>
</table>

Example using user name (user_name).

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getFirstName()); // print the first name of the logged in user
var newUser = currentUser.getUserByID('abel.tuter'); // fetch a different user using the user_name field
gs.info(newUser.getFirstName()); // print the first name of the Abel Tuter user
```

Example using user ID (sys_id).

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.getFirstName()); // print the first name of the logged in user
var newUser = currentUser.getUserByID('62826bf03710200044e0bfc8bcbe5df1'); // fetch Abel Tuter user using sys_id from sys_user record
gs.info(newUser.getFirstName()); // print the first name of the Abel Tuter user
```
GlideUser - getUserRoles()

Returns the list of roles explicitly granted to the user.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object/String</td>
<td>List of comma-separated roles explicitly assigned to the user.</td>
</tr>
</tbody>
</table>

**Note:** The data type for the returned value is object, however, the information is returned as a string.

```javascript
var currentUser = gs.getUser();
var userRoles = currentUser.getUserRoles();
gs.info('User Roles: ' + userRoles);
gs.info('Data type: ' + typeof userRoles);
```

**Scoped equivalent**

To use the `getUserRoles()` method in a scoped application, use the corresponding scoped method: **Scoped GlideUser - getUserRoles()**.

GlideUser - hasRole(String role)

Determines if the current user has the specified role.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>role</td>
<td>String</td>
<td>Role to check</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the user has the role.</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.hasRole('admin'));
```

Scoped equivalent

To use the `hasRole()` method in a scoped application, use the corresponding scoped method: `Scoped GlideUser - hasRole(String role)`.

GlideUser - isMemberOf(String group)

Determines if the current user is a member of the specified group. Only active groups are evaluated by this method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group</td>
<td>String</td>
<td>Group to check</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the user is a member of the group.</td>
</tr>
</tbody>
</table>

```javascript
var currentUser = gs.getUser();
gs.info(currentUser.isMemberOf('Capacity Mgmt'));
```

Scoped equivalent

To use the `isMemberOf()` method in a scoped application, use the corresponding scoped method: `Scoped GlideUser - isMemberOf(String group)`.

GlideXMLUtil - Scoped, Global

Provides methods to remove invalid characters from an XML string, and to validate an XML string.

Access these methods using the static object `GlideXMLUtil`. This class is available in scoped and global scripts.
GlideXMLUtil - removeInvalidChars(String xmlString)

Removes invalid characters from an XML string.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xmlString</td>
<td>String</td>
<td>The string to be processed.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A string with invalid characters removed.</td>
</tr>
</tbody>
</table>

```javascript
var test = "test\btab";
var removedTest = GlideXMLUtil.removeInvalidChars(test);
gs.info(removedTest);
```

Output: testtab

GlideXMLUtil - validateXML(String xmlString, Boolean nsAware, Boolean forgiveUnclosed)

Determines if the specified string is valid XML.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xmlString</td>
<td>String</td>
<td>The string to be validated.</td>
</tr>
<tr>
<td>nsAware</td>
<td>Boolean</td>
<td>When true, the validation is aware of name spaces. When false, the validation ignores name spaces.</td>
</tr>
<tr>
<td>forgiveUnclosed</td>
<td>Boolean</td>
<td>When true, the validation does not check for &lt;xml&gt; tags enclosing the string.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Returns null if the string is valid. Returns an error string describing the error if the specified string is not valid.</td>
</tr>
</tbody>
</table>

```javascript
var s = "<?xml version="1.0" encoding="UTF-8"?><unload unload_date="2017-11-27 21:56:14"><incident action="INSERT_OR_UPDATE"><active>true</active></incident></unload>";
var xml = GlideXMLUtil.validateXML(s, false, false);
gs.info(xml);
```

Output: null

### Guided Tours - Client

Provides methods for launching and stopping guided tours.

This API includes methods used in Guided Tour Designer.

**Guided Tours - startTour(String tour_id, Number step_number, Function cb_function)**

Starts a tour. Because this method is asynchronous, you must pass a callback function to determine operation success.

Complete signature includes `top.NOW.guided_tours.api` preceding the method name.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tour_id</td>
<td>String</td>
<td>Sys ID of the tour from the Guided Tours [sys_embedded_tour_guide] table.</td>
</tr>
<tr>
<td>step_number</td>
<td>Number</td>
<td>Optional. Step at which to start the tour. If not provided (or step number is 0), tour starts from the beginning.</td>
</tr>
<tr>
<td>cb_function</td>
<td>Function</td>
<td>Optional. Callback function called by <code>startTour()</code> method after attempt to launch the tour.</td>
</tr>
<tr>
<td>cb_function.err</td>
<td>Object</td>
<td>Points to the error object if any occurred during the operation:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>err = { success: false, message: 'string containing the error object' }</code></td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Null otherwise.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

//create a callback function to handle the result of the API call
var cbFunction = function(err) {
  if (err) {
    console.log('Error Occurred');
  }
  else {
    console.log('The tour with tourid=%s was successfully launched', tourId);
  }
}

//calling the startTour method
top.NOW.guided_tours.api.startTour('a297e04b732313007077edcc5ef6a780', 2, cbFunction);

Guided Tours - endTour()

Stops a currently playing tour. This method silently exits if no tours are playing.

Complete signature includes top.NOW.guided_tours.api preceding the method name.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Null</td>
<td>Null</td>
</tr>
</tbody>
</table>
// create a callback function to end the tour if it starts correctly
var cbFunction = function(err) {
  if (err) {
    console.log('Error Occurred');
  }
  else {
    // tour has started so we can call endTour
    top.NOW.guided_tours.api.endTour();
  }
}

// calling the startTour method so that we can end the tour as soon as it starts
top.NOW.guided_tours.api.startTour('a297e04b732313007077edcc5ef6a780', 2, cbFunction);

Guided Tours - applyListFilter(Function filter_func)
Sets a function to retrieve filtered tour results when the getAllTours() method is called.

Complete signature includes top.NOW.guided_tours.api preceding the method name.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter_func</td>
<td>Function</td>
<td>Filter function that takes a single tour object from the tours[] array returned from getAllTours() method.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows basic API usage.

// create a filter function
var filtFunction = function(tour) {
  // only return those tours whose name starts with 'my'
  return tour.name.indexOf('my') === 0;
}

// apply the filter
top.NOW.guided_tours.api.applyListFilter (filtFunction);

//call the getAllTours method to observe the filtered tours
top.NOW.guided_tours.api.getAllTours (function(er, tours) {
  if(!er) {
    console.log('The filtered tours are: ');
    console.log(tours);
  }
});

The following example shows how to use the options field on the tour object to add JSON with custom tour identifiers for reading and filtering tours inside the filter_func() function.

top.NOW.guided_tours.api.applyListFilter(function(tour) {
  var options = (tour.options)? JSON.parse(tour.options): null;
  return (options && options.my_param) ? (options.my_param == my_value) : false;
});

Guided Tours - getAllTours(Function cb_function)

Gets a list of tours on the current page from which this method is called. Because this method is asynchronous, a callback function must be passed to determine operation success and get a list of tours.

Complete signature includes top.NOW.guided_tours.api preceding the method name.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cb_function</td>
<td>Function</td>
<td>Callback function called by getAllTours() after attempt to fetch all tours for the current page from which getAllTours() method is called.</td>
</tr>
<tr>
<td>cb_function.err</td>
<td>Object</td>
<td>Points to the error object if any occurred during the operation:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>err = { success: false, message: 'string containing the error object' }</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Null otherwise.</td>
</tr>
<tr>
<td>cb_function.tours</td>
<td>Array</td>
<td>List of available tours for the page.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If no tours are present on the page, cb_function.tours returns undefined.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>if(!tours) console.log('No tour present')</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

```javascript
//create a callback function to handle the result of the API call
var cbFunction = function(err, tours) {
  if (err) {
    console.log('Error Occurred');
  } else {
    if(!tours) console.log('No tour present')
    else {
      tours.forEach(function(t) {
        console.log(t);
      });
    }
  }
}
//calling the getTours method
top.NOW.guided_tours.api.getAllTours(cbFunction);
```

Guided Tours - loadPlayer()

Loads the guided tours player on a page in which guided tours player is not present by default.

Complete signature:

```javascript
NOW.guided_tours.api.loadPlayer()```

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Guided Tours - events.on(String event_name, Function listener_function)**

Attaches an event listener to a guided tour event.

Complete signature includes `top.NOW.guided_tours.api` preceding the method name.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_name</td>
<td>String</td>
<td>Event name to be attached to the listener. Valid event names:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• stepStarted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tourStarted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tourEnded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tourCompleted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tourFailed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tourAbandoned</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tourDismissed</td>
</tr>
<tr>
<td>listener_function</td>
<td>Function</td>
<td>Listener to be added.</td>
</tr>
<tr>
<td>listener_function</td>
<td>Object</td>
<td>Passed to <code>listener_function()</code> by each event in the following format:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For stepStarted events:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>{tour: '&lt;tour_sys_id&gt;', step: step_num}</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For all other events:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>{tour: '&lt;tour_sys_id&gt;'}</code></td>
</tr>
</tbody>
</table>

**Note:** Clear any event listener after it solves its purpose.

JSON parameters:
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tour_sys_id</td>
<td>String</td>
<td>Guided tour ID from the Guided Tours [sys_embedded_tour_guide] table</td>
</tr>
<tr>
<td>step_num</td>
<td>Number</td>
<td>Value between 0 (first step) and n (final step)</td>
</tr>
</tbody>
</table>

The following example shows basic API usage.

```javascript
//create a callback function to handle the result of the api call
var eventListenerTourStarted = function() {
  console.log('The tour has started');
};
var eventListenerTourEnded = function() {
  console.log('The tour has ended');
};

//attaching event listeners for tourStarted and tourEnded Events
top.NOW.guided_tours.events.on('tourStarted',eventListenerTourStarted);
top.NOW.guided_tours.events.on('tourEnded', eventListenerTourEnded);

... //start a tour
top.NOW.guided_tours.api.startTour ('a297e04b732313007077edcc5ef6a780', 2, cbFunction);
//As soon as the tour starts the eventListenerTourStarted gets fired
...
top.NOW.guided_tours.api.endTour();
// eventListenerTourEnded gets fired
...

//removing the event listeners
top.NOW.guided_tours.events.off('tourStarted',eventListenerTourStarted);
top.NOW.guided_tours.events.off('tourEnded', eventListenerTourEnded);
```

The following example shows how to use the `listener_function` parameter with `obj` as an argument.

```javascript
top.NOW.guided_tours.events.on("tourStarted", function (obj){console.log(obj)});
```

**Guided Tours - events.off(String event_name, Function listener_function)**

Removes an existing event listener.
Complete signature includes `top.NOW.guided_tours.api` preceding the method name.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_name</td>
<td>String</td>
<td>Event name to be removed from the listener. Valid event names:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tourStarted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tourEnded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tourCompleted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tourFailed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tourAbandoned</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tourDismissed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• stepStarted</td>
</tr>
<tr>
<td>listener_function</td>
<td>Function</td>
<td>Optional. If provided, specified listener function is removed from remaining event listeners attached with that event. If not provided, all listener functions attached to that event are removed.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
//create a callback function to handle the result of the api call
var eventListenerTourStarted = function() {
    console.log('The tour has started');
}
var eventListenerTourEnded = function() {
    console.log('The tour has ended');
}

//attaching event listeners for tourStarted and tourEnded Events
top.NOW.guided_tours.events.on('tourStarted', eventListenerTourStarted);
top.NOW.guided_tours.events.on('tourEnded', eventListenerTourEnded);
...
```

//start a tour
```
top.NOW.guided_tours.api.startTour ('a297e04b732313007077edcc5ef6a780', 2, cbFunction);
```
//As soon as the tour starts the eventListenerTourStarted gets fired
...
top.NOW.guided_tours.api.endTour();
// eventListenerTourEnded gets fired
...

//removing the event listeners

  top.NOW.guided_tours.events.off('tourStarted', eventListenerTourStarted);
  top.NOW.guided_tours.events.off('tourEnded', eventListenerTourEnded);

helpers - UI Builder

The helpers API provides general functionality that is common across page scripts, eliminating the need to write scripts for simple functionality such as opening and closing a modal.

This API is only available to page scripts, it is not available in any other UI Builder scripts including:

- component property value scripts
- component visibility scripts
- event payload scripts
- UX client script includes

helpers - helpers.modal.close(String modalId)

Closes the specified modal on the current page.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>modalId</td>
<td>String</td>
<td>Modal component ID of the modal to close. Component IDs are auto generated when a component is dragged and dropped on the UI Builder stage. You can locate the ID on the property page.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
This example shows closing a modal with a component ID that ends with `alert-modal`.

```javascript
function handler({api, event, imports, helpers}) {
    helpers.modal.close("[component-id$='alert_modal']")
}
```

**helpers - helpers.modal.open(String modalId, Object options)**

Opens the specified modal on the current page.

You can only display one modal at a time within a page. If a modal is currently open, and you call this method, the existing modal is hidden and the new modal appears.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>modalId</td>
<td>String</td>
<td>Component ID of the modal to open. Component IDs are auto generated when a component is dragged and dropped on the UI Builder stage. You can locate the ID on the property page.</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional.</td>
</tr>
</tbody>
</table>

```javascript
"options": {
    "content": "String",
    "contentFullWidth": Boolean,
    "headerLabel": "String",
    "size": "String"
}
```

- **options.content**
  - **Type**: String
  - **Description**: Text content for the modal.

- **options.contentFullWidth**
  - **Type**: Boolean
  - **Description**: Flag that indicates whether to remove the horizontal padding around the body of the modal in order to fit wider content. Valid values:
    - true: Remove the padding.
    - false: Do not remove the padding.
  - **Default**: false

- **options.headerLabel**
  - **Type**: String
  - **Description**: Text content for the modal header.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| options.size | String| Size of the modal container. Most sizes automatically expand to fill the viewport when the screen size is small. Valid values:  
• sm  
• md  
• lg  
• fullscreen  
Default: sm |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows opening a modal with a component ID that ends with `alert-modal`.

```javascript
function handler({api, event, imports, helpers}) {
    helpers.modal.open("[component-id$='alert_modal']")
}
```

`helpers - helpers.navigate.setRouteParams(Object params)`

Passes the specified parameters down to other components on the same page.

Use this method in any page component that wants to add a parameter in a URL. You may want to add a parameter to a URL when another component needs to know the current value of that parameter when it changes, so it can react to it. For example, use this method to pass the `selectedIndex` of a tab component so it reflected in the URL to give focus to that tab.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>params</td>
<td>Object</td>
<td>Key-value pairs of optional parameters to pass to other components.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>This must be a plain, flat object with only primitive values. Array or object references are ignored and not added to the URL. All specified keys # must be part of# the optional parameters# in the route configuration or they are also ignored. For additional information on optional parameters, see Create a page in UI Builder.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This code example shows how to append the URL `params/selected-tab-index/2`. Note that the parameter in the actual URL is changed from camel case to snake case, so `selectedTabIndex` becomes `selected-tab-index`.

```javascript
function handler({api, event, helpers, imports}) {
    helpers.navigate.setRouteParams({'params': {'selectedTabIndex': 2}});
}
```

**helpers** - `helpers.navigate.to(String route, Object fields, Object params, Boolean redirect, Boolean passiveNavigation, String targetRoute)`

Navigates from one screen to another based on the specified route and field information. URL changes and the respective screen loads are observed.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| route    | String  | Name of the route. Must be a valid entry from UX App Routes (sys_ux_app_route.list). This value is reflected in the URL, and the URL is created based on the `route`, `fields`, and `optionalParams` column values: /<route>/<field1Value>/<field2Value>/params/<optionalParamKey1>/<optionalParamValue1>/<optionalParamKey2>/<optionalParamValue2>  

For example: `/record/incident/12345/params/selectedTabIndex/4`
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>Object</td>
<td>Optional. Key-value pairs of required parameters. For example: 'fields': {'table': 'incident', 'sysId': '12345'}.</td>
</tr>
<tr>
<td>params</td>
<td>Object</td>
<td>Optional. Key-value pairs of optional parameters. For example: 'params': {'selectedTabIndex': 4}.</td>
</tr>
</tbody>
</table>
| redirect | Boolean    | Flag that indicates whether to remove the latest history entry from the browser history. For example, if you navigate to sites A, B and then C. If redirect is set to true while navigating to C, the history entry for B is removed. The browser history only shows only A and C. Valid values:  
  • true: Removes the latest history entry, and redirects to the latest URL.  
  • false: Does not remove any history entries. Default: false |
| passiveNavigation | Boolean | Flag that indicates whether to perform background navigation. Background navigation is when a page is opened but it is not active or shown. For example, opening an inactive tab for the page, but it is not visible but loaded in the background. Valid values:  
  • true: Perform background navigation.  
  • false: Do not perform background navigation. Default: false |
| targetRoute | String or Object | Sub navigation to a drill-down, deep-link, or sub-tab. If set to current, the current route does a sub-navigation under the current URL.  
For example, if /record/incident/123 is the current URL, and the following call is made:  
  helper.navigate.to('record', {'table': 'problem', 'sysId': '567'}, {}, false, false, 'current'); |
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>The following URL is generated: /record/incident/123/sub/record/problem/567</td>
</tr>
</tbody>
</table>

**Note:** `targetRoute` can be either a string such as 'current' or an object, such as navigation `NAV_ITEM_SELECTED` payload.

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to navigate to a page passing just the `route` parameter.

```javascript
function handler({api, event, imports, helpers}) {
    helpers.navigate.to('test');
}
```

This example shows how to navigate to a page passing the `route` and `fields` parameters.

```javascript
function handler({api, event, imports, helpers}) {
    helpers.navigate.to('test', {'key': 'value'});
}
```

This example shows how to navigate to a page passing the `route`, `fields`, and `params` parameters.

```javascript
function handler({api, event, helpers, imports}) {
    helpers.navigate.to('test', {'key': 'value'}, {'first': 'FirstName', 'last': 'LastName'});
}
```

**helpers - helpers.screen.updateStatus(Object statusObj)**

Enables pages to report their status updates, such as title, icon, dirty state, message, and error changes.

Status updates are reported to WorkspaceChrome or AppShell, whichever the outer layer is, and acting as the host.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| statusObj | Object   | Payload to send to the current page to report that the content has been updated. Valid values:  
  - `dirtyModalId`: (String) ID of the modal that has changed.  
  - `hasError`: (Boolean) Flag that indicates that there are errors on the page.  
  - `hasUpdate`: (Boolean) Flag that indicates that there were updates to the page.  
  - `icon`: (String) Name of the updated or new icon.  
  - `isDirty`: (Boolean) Flag that indicates whether the page is dirty (values have changed).  
  - `message`: (String) Updated/new message.  
  - `screenKey`: (String) ID of the screen on which the change occurred. Every screen has a `screenKey` as a property on the screen macroponent inside `sn-canvas-screen`.  
  - `status`: (String) Status operation for this action. This value must be one of the following: inserted, deleted, saved, or closed.  
  - `title`: (String) Updated/new display title.  
  - `tooltipPreview`: (JSON) Updated or new tool tip. For example, `tooltipPreview : { primaryTitle, secondaryContent: {} }` |

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
screen.updateStatus({'dirtyModalId': 'customModalId', 'isDirty': true});
```
helpers - helpers.snHttp(String url, Object options)

Makes an HTTP request to the ServiceNow instance and returns a promise with the results.

⚠️ **Note:** There is a known issue where objects named `options` are omitted from the HTTP response.

```json
{
  options: {},
  otherFields: {}
}
```

becomes

```json
{
  otherFields: {}
}
```

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>HTTP endpoint relative to the instance URL. For example, <code>/api/now/table/incident</code> or <code>/api/now/table/incident/a83820b58f723300e7e16c7827bdeed</code>.</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Describes the contents of the HTTP request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;options&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;batch&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;body&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;headers&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;method&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td>options.batch</td>
<td>Boolean</td>
<td>Optional. Flag that indicates whether this HTTP request should be batched with other HTTP requests made to the instance. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Make request as part of a batch request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Make dedicated request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: true</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options.body</td>
<td>Object</td>
<td>Optional. Data to send as the request body. Only applicable for request methods <code>PUT</code>, <code>POST</code>, and <code>PATCH</code>. Elements in the object depend on the type of HTTP method. For details, refer to the code examples below. Default: <code>{}</code></td>
</tr>
<tr>
<td>options.headers</td>
<td>Object</td>
<td>Optional. Additional HTTP request headers. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td></td>
<td>headers: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Content-Type&quot;: &quot;application/json&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Accept&quot;: &quot;application/xml&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td>options.method</td>
<td>String</td>
<td>Optional. HTTP method. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DELETE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• GET</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PATCH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• POST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• PUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: GET</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Object that describes any error returned by the REST API. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;data&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;options&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error.data</td>
<td>Response returned from the HTTP API. Data type: Defined by REST API</td>
</tr>
<tr>
<td>error.message</td>
<td>Message describing the error encountered when trying to process the HTTP request.</td>
</tr>
<tr>
<td></td>
<td><img src="https://via.placeholder.com/15" alt="Note: This parameter is not always returned." /> Data type: String</td>
</tr>
<tr>
<td>error.options</td>
<td>Describes the original HTTP request. Data type: Object</td>
</tr>
<tr>
<td>error.options.headers</td>
<td>Object containing a list of all of the request headers sent in the request. Data type: Object</td>
</tr>
<tr>
<td>error.options.responseHeaders</td>
<td>Object containing a list of all of the response headers sent in the request. Data type: Object</td>
</tr>
<tr>
<td>error.status</td>
<td>Returned HTTP error status code, such as 400, 405, or 500. Data type: Number</td>
</tr>
<tr>
<td>error.statusText</td>
<td>Returned HTTP status message, such as Bad Request. Data type: String</td>
</tr>
<tr>
<td>response</td>
<td>Returned when HTTP request is successful. The response to the HTTP request. Data type: Any</td>
</tr>
</tbody>
</table>

The following example show how to call `snHttp()` which returns a promise.
function handler({api, event, helpers, imports}) {
  helpers.snHttp('/api/now/table/u_movie', {method: 'GET'})
    .then(({response}) => {
      // do something with the table data
    })
    .catch(({error}) => {
      const message = `Error: ${error.data.error.message}`;
      console.error(message);
      api.emit('NOW_UXF_PAGE#ADD_NOTIFICATIONS', {
        id: 'alert5',
        status: 'high',
        icon: 'info-circle-outline',
        content: message,
        action: { type: 'dismiss' }
      });
    });
}

The following example show how to call `snHttp()` using `async` and `await`.

async function handler({helpers}) {
  try {
    const result = await helpers.snHttp('/api/now/table/u_movie', {method: 'GET'});
  } catch ({error}) {
    const message = `Error: ${error.data.error.message}`;
    console.error(message);
    api.emit('NOW_UXF_PAGE#ADD_NOTIFICATIONS', {
      id: 'alert5',
      status: 'high',
      icon: 'info-circle-outline',
      content: message,
      action: { type: 'dismiss' }
    });
  }
}

The following example show how to set up a POST request.

function handler({api, helpers, event, imports}) {
  helpers
    .snHttp('/api/now/table/incident', {
      method: "POST",
      body: {
        description: "Sample description",
      }
    });
}
The following example show how to set up a PUT request.

```javascript
function handler({api, helpers, event, imports}) {
  helpers
    .snHttp(`/api/now/table/incident/a83820b58f723300e7e16c7827bdeed2`, { 
        method: "PUT",
        body: {
            activity_due: "1970-04-02 18:26:17"
        },
        headers: {
            "Content-Type": "application/json",
            "Accept": "application/xml"
        }
    })
    .then(({ response }) => { 
        // handle PUT request response
    })
    .catch(({ error }) => { 
        // handle PUT request errors
    });
}

helpers - helpers.timing.clearInterval(Number timeoutId)
Cancels the execution of the function that was scheduled through a prior setInterval() call.
```
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeoutId</td>
<td>Number</td>
<td>Unique identifier of the scheduled function to clear. This value is returned by the corresponding setInterval() call.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows using `clearInterval()` to cancel a timing operation that was previously set using the `setInterval()` method. This function could be invoked by a user clicking a **Disable Auto-refresh** button on a page.

```javascript
function handler({api, helpers}) {
  api.setState('intervalId', ({currentValue}) => {
    if (currentValue > -1) {
      helpers.timing.clearInterval(currentValue);
    }
    return -1;
  });
}
```

**helpers** - `helpers.timing.clearTimeout(Number timeoutId)`

Cancels the execution of the function that was scheduled through a prior `setTimeout()` call.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeoutId</td>
<td>Number</td>
<td>Unique identifier of the scheduled function to clear. This value is returned by the corresponding setTimeout() call.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
This code example shows how to terminate a timer with the specified `timeoutId`.

```javascript
function handler({api, helpers}) {
    api.setState('timeoutId', ({currentValue}) => {
        if (currentValue > -1) {
            helpers.timing.clearTimeout(currentValue);
        }
        return -1;
    });
}
```

`helpers - helpers.timing.setInterval(Function fn, Number delay)`

Repeatedly executes the specified function, using the specified delay value as the interval between function calls.

Unlike the native JavaScript `setInterval()` method, this method does not support passing a code string to evaluate as the first argument. Any additional arguments are passed to the function itself.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fn</td>
<td>Function</td>
<td>Function to repeatedly execute.</td>
</tr>
<tr>
<td>delay</td>
<td>Number</td>
<td>Length of the time-interval between each function execution. Unit: Milliseconds</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Unique identifier of the function execution operation. Use this value in the <code>helpers - helpers.timing.clearInterval(Number timeoutId)</code> method if you need to cancel this operation.</td>
</tr>
</tbody>
</table>

This code example shows how to refresh the timestamp on a page every second. This function could be invoked by a user clicking an **Enable Auto-refresh** button on a page.

```javascript
function handler({api, helpers}) {
    // Every one second, refresh the value of current timestamp client state parameter
    const intervalId = helpers.timing.setInterval(() => {

    });
}
```
api.setState('currentTimestamp', new Date().toString(), 1000);

// The interval ID is kept in state to use when calling the
helpers.timing.clearInterval() method at a later point
api.setState('intervalId', intervalId);
}

helpers - helpers.timing.setTimeout(Function fn, Number delay)

Executes the specified function, after the specified delay.

Unlike the native JavaScript setTimeout() method, this method does not support passing a code string to evaluate as the first argument. Any additional arguments are passed to the function itself.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fn</td>
<td>Function</td>
<td>Function to execute.</td>
</tr>
<tr>
<td>delay</td>
<td>Number</td>
<td>Length of the time to wait before calling the specified function.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unit: Milliseconds</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Unique identifier of the function execution operation. Use this value in the helpers - helpers.timing.clearTimeout(Number timeoutId) method if you need to cancel this operation.</td>
</tr>
</tbody>
</table>

This code example shows how to set a 20 minute timer. You could associate this function with a button **Remind me in 20 minutes**.

```javascript
function handler({api, helpers}) {
  const timeoutId = helpers.timing.setTimeout(() => {
    api.emit('NOW_UXF_PAGE#ADD_NOTIFICATIONS', {
      id: 'alert5',
      status: 'high',
      icon: 'info-circle-outline',
```
content: 'Try to look away at something that is 20 feet away from you for a total of 20 minutes."

action: { type: 'dismiss' }
});
}, 20 * 60 * 1000);

// The timeout ID is kept in state to use when calling the helpers.timing.clearTimeout() method at a later point
api.setState('timeoutId', timeoutId);
}

helpers - helpers.translate(String message, String tokens)
Asynchronously retrieves and translates the specified message based on the current user's session language.

You can use this method with the api - setState(String stateParam, Any value) to bind the translated value to other fields on the page.

**Note:** You can call this method using a promise or `async` and `await`. The code examples below show both implementations.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Message to translate.</td>
</tr>
<tr>
<td>tokens</td>
<td>String</td>
<td>Optional. Comma-separated list of parameters to use for replacing string variables. For example:</td>
</tr>
</tbody>
</table>

```javascript
helpers.translate('Text {0} {1}', 'to', 'translate');
```

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Translated text string.</td>
</tr>
</tbody>
</table>

The following example shows how to pass in table field references to embed in the corresponding variables in a string, using a promise.

```javascript
function handler ({api, helpers}) {
  helpers.translate('Welcome {0} {1}!', user.firstName, user.lastName)
  .then((translatedText) => {
    ...
  });
}
The following example shows how to use `async` and `await` in your function instead of a promise.

```javascript
async function handler ({api, helpers}) {
  const translatedText = await helpers.translate('Welcome to {0}', 'ServiceNow');
  api.setState('greeting', translatedText);
}
```

**HistoryWalker - Scoped, Global**

The `HistoryWalker` API uses the audit/history tables to generate a historical version of an existing record.

It supports the ability to return a GlideRecord to a previous update count (walked GlideRecord) with the appropriate GlideElements populated. After the walked GlideRecord is retrieved, the API provides the ability to move forward and backward the update numbers navigating through its historical updates.

To use this class in scoped and global applications, use the `sn_hw` namespace identifier. The History Walker plugin (com.glide.history_walker) that is enabled by default is required to access the `HistoryWalker` API.

⚠ **Note:** For offline updates, the `HistoryWalker` API is initiated automatically and the only two methods that you can use are: `walkTo()` (the input argument for this method can only be zero) and `walkForward()`. Other available methods cannot be invoked for offline updates.

The `HistoryWalker` API provides two ways to retrieve the audit data:

- **Using History Set:** A History Set entry is created (if not available or not up to date) from the data in the Sys Audit `[sys_audit]` table for the record that you are going to walk through. The History Set table contains records (History Lines) with the actual changes to field values that occurred. Methods of the `HistoryWalker` API retrieve the history data from the generated History Lines, instead of querying the `sys_audit` table.

- **Using Sys Audit table:** In this case, the `HistoryWalker` API extracts data directly querying the `sys_audit` table.

By default, it populates the data to support the `changes()`, `changesFrom()`, and `changesTo()` methods in the walked record, as well as provides record and field level security. Additionally, it can enable journal fields and variables to be also populated in the walked GlideRecord when walking through the updates.
This API enables you to:

- Apply the appropriate history/audit data to get an existing GlideRecord to the state it was in a specific update count.
- Instruct the HistoryWalker API to use sys_audit table instead of sys_history_set/sys_history_line tables to retrieve its data.
- Turn off row-level access control.
- Turn off field-level access control.
- Turn off retrieval and processing of “changes” data.
- Enable journal fields.
- Enable variables.

**HistoryWalker - HistoryWalker(String tableName, String sysId)**

Fetches the database record based on the parameters, using the History Sets to retrieve the historic data.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of table containing the record to retrieve.</td>
</tr>
<tr>
<td>sysId</td>
<td>String</td>
<td>sys_id of the record to retrieve.</td>
</tr>
</tbody>
</table>

### Example:

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
if (hw.walkTo(3)) {
    var oldPriority = hw.getWalkedRecord().priority;
    gs.info('Incident priority in update number ' + hw.getUpdateNumber() + ' was ' + oldPriority);
} else

    gs.info('Incident does not have update number 3');
```

### Output:

```
Incident priority in update number 3 was 4
```
HistoryWalker - HistoryWalker(String tableName, String sysId, Boolean useAudit)

Fetches the database record based on the parameters, using the History Sets or Audit data to retrieve the historic data, depending on the third parameter.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of table containing the record to retrieve.</td>
</tr>
<tr>
<td>sysId</td>
<td>String</td>
<td>sys_id of the record to retrieve.</td>
</tr>
<tr>
<td>useAudit</td>
<td>Boolean</td>
<td>• If set to true, uses audit data to retrieve historic date.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If set to false, uses history set to retrieve historic date.</td>
</tr>
</tbody>
</table>

**Example**

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue(), true);
if (hw.walkTo(3)) {
    var oldPriority = hw.getWalkedRecord().priority;
    gs.info('Incident priority in update number ' + hw.getUpdateNumber() + ' was ' + oldPriority);
} else
    gs.info('Incident does not have update number 3');
```

**Output:**

```
Incident priority in update number 3 was 4
```

HistoryWalker - getUpdateNumber()

Gets the update number of the current walked glide record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>Current update number or, -1 if record is not found</td>
</tr>
</tbody>
</table>

Example:

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
hw.walkTo(3);
gs.info('Update number: ' + hw.getUpdateNumber());
```

Output:

```
Update number: 3
```

**HistoryWalker - getWalkedRecord()**

Gets the record filled with the history/audit data after walking to an update number.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The walked GlideRecord.</td>
</tr>
</tbody>
</table>

Example:

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
hw.walkTo(0);
var walkedRecord = hw.getWalkedRecord();
gs.info('Priority in update number 0: ' + walkedRecord.priority);
hw.walkTo(1);
```
walkedRecord = hw.getWalkedRecord();
gs.info('Short description in update number 1: ' + walkedRecord.short_description);

Output:

Priority in update number 0: 4
Short description in update number 1: My monitor has stopped working

HistoryWalker - getWalkedRecordCopy()

Returns a copy of the record filled with the history/audit data after walking to an update number.

**Note:** The `getWalkedRecord()` API might modify the obtained `walkedRecord` after walking to another update number. The `getWalkedRecordCopy()` API gets a clone to the walked record to prevent that.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>A copy of the walked GlideRecord.</td>
</tr>
</tbody>
</table>

**Example:**

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC00000015');

var walkedRecord = [];
var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
hw.walkTo(0);
walkedRecord[0] = hw.getWalkedRecordCopy();
hw.walkTo(1);
walkedRecord[1] = hw.getWalkedRecordCopy();

gs.info('Priority in update number 0: ' + walkedRecord[0].priority);
gs.info('Short description in update number 1: ' + walkedRecord[1].short_description);
```

Output:
HistoryWalker - isFieldLevelSecurity()

Specifies if the record-level read access is applied on the record when retrieving from the database.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if field level security is enabled, else returns false.</td>
</tr>
</tbody>
</table>

Example:

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
gs.info('Field level security is active: ' + hw.isFieldLevelSecurity());
```

Output:

Field level security is active: true

HistoryWalker - isRecordLevelSecurity()

Specifies if the record-level read access is applied on the record when retrieving from the database.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the record-level security is enabled, else returns false.</td>
</tr>
</tbody>
</table>

**Example:**

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
gs.info('Record level security is active: ' + hw.isRecordLevelSecurity());
```

**Output:**

Record level security is active: true

---

### HistoryWalker - isWithChanges()

Specifies if any of the methods that walk the record from one update to another, support the “changes” data for each element.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the changes support is enabled, else returns false.</td>
</tr>
</tbody>
</table>

**Example:**

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
gs.info('Changes is active: ' + hw.isWithChanges());
```

**Output:**

Changes is active: true
HistoryWalker - isWithJournalFields()

Specifies if journal type fields are populated from the historical values.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if journal fields are populated, else returns false.</td>
</tr>
</tbody>
</table>

**Example:**

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
gs.info('Populating journal fields is active: ' + hw.isWithJournalFields());
```

Output:

```
Populating journal fields is active: false
```

HistoryWalker - isWithVariables()

Specifies if values are set for variables that are recorded in the history.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if including values for variables, else returns false.</td>
</tr>
</tbody>
</table>

**Example:**

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
gs.info('Setting variables is active: ' + hw.isWithVariables());
```
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
gs.info('Populating variables is active: ' + hw.isWithVariables());

Output:

Populating variables is active: false

**HistoryWalker - setFieldLevelSecurity(Boolean fieldLevelSecurity)**

Sets the field-level read access on each element before setting the historical value of that element in the GlideRecord. If the field-level security is enabled, it prevents the API to populate the fields of the walked record that the user of the API does not have access to.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldLevelSecurity</td>
<td>Boolean</td>
<td>If set to true, field-level security is enabled. The default value is true.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Example:**

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
hw.setFieldLevelSecurity(false);
hw.walkTo(0);
```

**HistoryWalker - setRecordLevelSecurity(Boolean recordLevelSecurity)**

Sets the record-level read access on the record when retrieving from the database. The record-level security prevents the API to retrieve the walked record if the user of the API does not have access to the GlideRecord.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>recordLevelSecurity</td>
<td>Boolean</td>
<td>If set to true, record-level read access security is enabled. The default value is true.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example:

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
hw.setRecordLevelSecurity(false);
hw.walkTo(0);
```

**HistoryWalker - setWithChanges(Boolean withChanges)**

Sets the “changes” data support for each element for a method that walks the record from one update to another.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>withChanges</td>
<td>Boolean</td>
<td>If set to true, the “changes” data is supported for each element. The default value is true.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example: With the Changes data support

```javascript
var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
hw.walkTo(0);
do {
```
function printChangedFields(hw) {
    var walkedGr = hw.getWalkedRecord();
    var fields = GlideScriptRecordUtil.get(walkedGr).getChangedFieldNames();
    gs.info("Fields changed at update " + hw.getUpdateNumber() + " were:");
    for (var j = 0; j < fields.size(); j++)
        gs.info(" " + fields.get(j));
    gs.info("\n");
}

Example: Without the Changes data support

var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
hw.setWithChanges(false);
hw.walkTo(0);
do {
    var oldPriority = hw.getWalkedRecord().priority;
    gs.info('Incident priority in update number ' + hw.getUpdateNumber() + ' was ' +
    oldPriority);
} while (hw.walkForward());

HistoryWalker - setWithJournalFields(Boolean withJournalFields)
Specifies if journal type fields are populated from the historical values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>withJournalFields</td>
<td>Boolean</td>
<td>If set to true, include journal-type fields. Th default value is false.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
hw.setWithJournalFields(true);
if (hw.walkTo(0)) {
    var workNotes = hw.getWalkedRecord().work_notes;
    gs.info('Work Notes in update number ' + hw.getUpdateNumber() + ' was ' + workNotes);
}

HistoryWalker - setWithVariables(Boolean withVariables)
Specifies if variables are populated from the historical values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>withVariables</td>
<td>Boolean</td>
<td>If set to true, values are populated for variables. The default value is false.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example:

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
hw.setWithVariables(true);
hw.walkTo(0);
if (hw.walkTo(0)) {
    var varUrgency = hw.getWalkedRecord().variables.urgency;
    gs.info('Variable Urgency in update number ' + hw.getUpdateNumber() + ' was ' + varUrgency);
}
```
**HistoryWalker - walkBackward()**

Applies the appropriate history/audit data to get a walked GlideRecord to the state when it was one update number backward. If the previous update count is missing from the history/audit data, it will walk to the previous available update count.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if walking to the specified update number was possible. Else, returns false, for example if already walked to the update number 0.</td>
</tr>
</tbody>
</table>

**Example:**

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
hw.walkTo(incGr.sys_mod_count);
do {
    var oldPriority = hw.getWalkedRecord().priority;
    gs.info('Incident priority in update number ' + hw.getUpdateNumber() + ' was ' + oldPriority);
} while (hw.walkBackward())
```

**Output:**

```
Incident priority in update number 5 was 2
Incident priority in update number 4 was 4
Incident priority in update number 3 was 4
Incident priority in update number 2 was 4
Incident priority in update number 1 was 4
Incident priority in update number 0 was 4
```
**HistoryWalker - walkForward()**

Applies the appropriate history/audit data to get a walked GlideRecord to the state when it was one update number forward. If next update count is missing from the history/audit data, it will walk to the next available update count.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if walking to the specified update number was possible. Else, returns false, for example if already walked to the GlideRecord update count.</td>
</tr>
</tbody>
</table>

**Example:**

```javascript
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC00000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
hw.walkTo(0);
do {
    var oldPriority = hw.getWalkedRecord().priority;
    gs.info('Incident priority in update number ' + hw.getUpdateNumber() + ' was ' + oldPriority);
} while (hw.walkForward())
```

**Output:**

```
Incident priority in update number 0 was 4
Incident priority in update number 1 was 4
Incident priority in update number 2 was 4
Incident priority in update number 3 was 4
Incident priority in update number 4 was 4
Incident priority in update number 5 was 2
```
**HistoryWalker - walkTo(int updateCount)**

Applies the appropriate history/audit data to get a GlideRecord to the state it was in a specific update count. Use getWalkedRecord() or getWalkedRecordCopy() after walking to an update number to retrieve the “walked” GlideRecord.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>updateCount</td>
<td>Integer</td>
<td>The update number to walk to.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>true if walking to the specified update number was possible, false otherwise, for example if the requested update is greater than the update count of the GlideRecord, or if there is no history/audit data of the requested update number</td>
</tr>
</tbody>
</table>

**Example:**

```java
var incGr = new GlideRecord('incident');
incGr.get('number', 'INC0000015');

var hw = new sn_hw.HistoryWalker(incGr.getTableName(), incGr.getUniqueValue());
if (hw.walkTo(3)) {
    var oldPriority = hw.getWalkedRecord().priority;
    gs.info('Incident priority in update number ' + hw.getUpdateNumber() + ' was ' + oldPriority);
} else
    gs.info('Incident does not have update number 3');
```

**Output:**

```
Incident priority in update number 3 was 4
```

**HostnameJS - Global**

Formats host names according to property settings.

Use with any server-side script when you need to format host names.
HostnameJS - format(String hostname, String source)
Formats the specified host name according to the property settings.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hostname</td>
<td>String</td>
<td>The host name to format</td>
</tr>
<tr>
<td>source</td>
<td>String</td>
<td>The property settings source</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The system name</td>
</tr>
</tbody>
</table>

HostnameJS - getDomainName()
Returns the DNS domain name.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The domain name</td>
</tr>
</tbody>
</table>

```javascript
var hjs = new HostnameJS();
hjs.getDomainName();
```

HostnameJS - getSysName()
Returns the current system name.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The system name</td>
</tr>
</tbody>
</table>

```javascript
var hjs = new HostnameJS();
hjs.getSysName();
```

**hr_ActivitySet - Scoped**

Handles lifecycle event activity set cases.

Lifecycle event activity sets represent different stages in the lifecycle event process. You must define when the activity set is triggered, such as immediately upon creation of the lifecycle event case or after the completion of another activity set. Each activity set is associated with a single lifecycle event.

The hr_ActivitySet API requires the HR Lifecycle Events plugin (com.sn_hr_lifecycle_events).

See also Understanding Enterprise Onboarding and Transitions.

**hr_ActivitySet - hr_ActivitySet()**

Instantiates an instance of the hr_ActivitySet class.

**hr_ActivitySet - createLECaseByService(String hrService, Object caseValues)**

Creates a lifecycle event case based on a specified HR service.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hrService</td>
<td>String</td>
<td>Sys ID of HR service in the HR Services [sn_hr_core_service] table to be assigned to the lifecycle event case.</td>
</tr>
<tr>
<td>caseValues</td>
<td>Object</td>
<td>Key-value pairs containing a value for each case field.</td>
</tr>
<tr>
<td>caseValues.subject_person</td>
<td>String</td>
<td>Sys ID of subject person's name in the User [sys_user] table.</td>
</tr>
<tr>
<td>caseValues.opened_for</td>
<td>String</td>
<td>Sys ID of case opened for subject person in the Users [sys_user] table.</td>
</tr>
<tr>
<td>caseValues.subject_person_job</td>
<td>String</td>
<td>Sys ID of subject person's Business title in the Jobs [sn_hr_core_job] table.</td>
</tr>
<tr>
<td>caseValues.location</td>
<td>String</td>
<td>Sys ID of subject person's location in HR profile in the Locations [cmn_location] table.</td>
</tr>
<tr>
<td>caseValues.department</td>
<td>String</td>
<td>Sys ID of subject person's department in HR profile in the Departments [cmn_department] table.</td>
</tr>
<tr>
<td>caseValues.subject_person_hr_profile</td>
<td>String</td>
<td>Sys ID of subject person's HR profile in the HR Profiles [sn_hr_core_profile] table.</td>
</tr>
<tr>
<td>caseValues.short_description</td>
<td>String</td>
<td>Case description.</td>
</tr>
<tr>
<td>caseValues.state</td>
<td>Number</td>
<td>Number representing uniquely-defined custom case state.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys ID of the created case in the HR Lifecycle Events Cases [sn_hr_le_case] table.</td>
</tr>
</tbody>
</table>

Output includes summary with Operation Table and Row Count columns. Click the link in the row count column adjacent sn_hr_le_case table to view the new case.

```javascript
var caseValues = {
    'subject_person': '62826bf03710200044e0bfc8bcbe5df1',
    'opened_for': '0e826bf03710200044e0bfc8bcbe5d7c',
    'subject_person_job': 'f8d56900587d3700964f4efae452b3db',
    'location': '25ab9d1d0a0a0bb300537fd25687439d',
    'department': '93b25282c0a8000b0b55c8ab34e2f1e6',
    'subject_person_hr_profile': '330370019f22120047a2d126c42e7012',
    'short_description': 'Short description',
    'state': 10
};

var activity = new sn_hr_le.hr_ActivitySet();
var createCase = new activity.createLECaseByService('4719647c67a60300132a6c3b5685ef16', caseValues);
```

Output:

```javascript
...
insert sn_hr_le_case 1
...
```

hr_ActivitySet - hasActiveCaseByService(String spUser, String hrService, String jobId)

Checks for a case matching the specified combination of subject person user, HR service, and job.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spUser</td>
<td>String</td>
<td>Sys ID of subject person user in the Users [sys_user] table.</td>
</tr>
<tr>
<td>hrService</td>
<td>String</td>
<td>Sys ID of HR service in the HR Services [sn_hr_core_service] table.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jobId</td>
<td>String</td>
<td>Sys ID of subject user's job title in the HR Services [sn_hr_core_service] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if matching case exists, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var userId = '62826bf03710200044e0bfc8bcbe5df1'; // Abel Tuter
var hrServiceId = '64fa4c53534222003066a5f4a11c0875'; // Onboarding HR Service
var jobId = '3D62826bf03710200044e0bfc8bcbe5df1'; // Sample Job sys_id for the example

var activity = new sn_hr_le.hr_ActivitySet();

var hasActiveLECase = activity.hasActiveCaseByService(userId, hrServiceId, jobId);

gs.info('User has an active HR case: ' + hasActiveLECase);
```

Output:

```
sn_hr_le: User has an active HR case: false
```

**hr_ActivityUtils - Scoped**

Enables creating HR cases from a record producer.

The hr_ActivityUtils script include requires the HR Lifecycle Events plugin (com.sn_hr_lifecycle_events).

**hr_ActivityUtils - createCaseFromProducer(GlideRecord current, Object producer, String cat_item.sys_id)**

Creates a case from a record producer and service. This method is intended to be called as-is to create a producer and does not require supplying data.

The following variables can be configured in a record producer to provide additional capabilities as follows:
• **job** – References `sn_hr_core_job` and maps to `subject_person_job` in an HR case

• **opened_for** – References `sys_user` and maps `opened_for` in an HR case

• **concurrent_job** – If the record producer includes this variable (by activating checkbox) and parameters have job information, creates a new job in `sn_hr_core_job` table

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>current</td>
<td>GlideRecord</td>
<td>GlideRecord for case to be created.</td>
</tr>
<tr>
<td>producer</td>
<td>Object</td>
<td>Comma-separated key value pair containing data as questions and answers that are automatically provided when a user submits an answer to a record producer question.</td>
</tr>
<tr>
<td>cat_item.sys_id</td>
<td>String</td>
<td>Looks up the HR service of case being created and variable names to populate the description.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>New case in related table, varies by product.</td>
</tr>
</tbody>
</table>

This method is intended to be called as-is to create a producer and does not require supplying data.

```
new sn_hr_core.hr_ActivityUtils(current).createCaseFromProducer(producer, cat_item.sys_id);
```

### hr_Utils - Scoped

Provides basic functionality for the HR application.

The `hr_Utils` API requires the HR core plugin (com.sn_hr_core).

**hr_Utils - hr_Utils()**

Instantiates an instance of the `hr_Utils` class.
hr_Utils - getPrimaryJob(String userId)

Gets the Sys ID of the active primary job for a provided user.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userId</td>
<td>String</td>
<td>Sys ID of the HR user from the User [sys_user] table assigned an active or future job in which Primary is true.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>If present and active, Sys ID of the primary job from the Jobs [sn_hr_core_job] table, null otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var result = new sn_hr_core.hr_Utils().getPrimaryJob('5137153cc611227c000bbd1bd8cd2007');
gs.info('Result: ' + result);
```

Output:

Result: eb3c69463cd63740964fb8b1ce04f9ae

hr_Utils - switchPrimaryJob(String userId String jobId)

Switches the primary job of a user.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userId</td>
<td>String</td>
<td>Sys ID of HR user from the Users [sys_user] table.</td>
</tr>
<tr>
<td>jobId</td>
<td>String</td>
<td>Job Sys ID from the Jobs [sn_hr_core_job] table.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>Encoded JSON with message and status, error otherwise.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var result = new sn_hr_core.hr_Utils().switchPrimaryJob('5137153cc611227c000bbd1bd8cd2007', 'bc884e72c0ebf00964fb8b1ce04f9d7');
gs.info('Result: ' + result);
```

Output:
```
Result: {"message":"Switched primary job for the user successfully","status":"success"}
```

**HttpRequestAuthedData - Scoped**

The `HttpRequestAuthedData()` API provides methods to access and set values in a signed REST or SOAP request.

Generate outbound signing requests using these APIs in the following order:

1. `HttpRequestData`: Build the API request.
2. `AuthCredential`: Create a credential object or update an existing one. Use the credential to sign the request through the `RequestAuthAPI` class.
3. `RequestAuthAPI`: Sign the request and return an `HttpRequestAuthedData` object.
4. `HttpRequestAuthedData`: Get information about the signed request.
5. `GlideHTTPRequest`: Send the signed request.

Before using these APIs, you must [configure an authentication algorithm](#) to sign the request and associate it with the credential used to authenticate the request.

Use this API in scoped scripts with the `sn_auth` namespace identifier. You can instantiate this class using the constructor, or you can return an `HttpRequestAuthedData` object from the `generateAuth()` method in the `RequestAuthAPI` class.

**HttpRequestAuthedData - HttpRequestAuthedData()**

Instantiates an `HttpRequestAuthedData` object.

You can instantiate this class using the constructor, or you can return an `HttpRequestAuthedData` object from the `generateAuth()` method in the `RequestAuthAPI` class.
### HttpRequestAuthedData - addHeader(String key, String value)

Adds a header to the HttpRequestAuthedData object.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the HTTP header.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value of the HTTP Header.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
// Define HttpRequestData
var endpoint= "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);

//Get AuthCredential
var credential = new
  sn_cc.StandardCredentialsProvider().getAuthCredentialByID("5b61c16f73533300f662cff8faf6a74b");

// Create RequestAuthAPI and sign the request
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var output = signingAPI.generateAuth();

// Update the signed response
output.addHeader('access-rights', 'public-read');
```
HttpRequestAuthedData - addQueryParam(String key, String value)

Adds a query parameter to the HttpRequestAuthedData object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the query parameter.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value of the query parameter.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

// Define HttpRequestData
var endpoint= "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);

//Get AuthCredential
var credential = new
sn_cc.StandardCredentialsProvider().getAuthCredentialByID("5b61c16f73533300f662c6f8f6a74b");

// Create RequestAuthAPI and sign the request
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var output = signingAPI.generateAuth();

// Update the signed response
output.addQueryParameter('api_version', 'v2');

HttpRequestAuthedData - getCredentialValue()

Returns the credential value that was included when the request was signed.
Use a credential value to store a cookie, signature, or other value needed for an authentication algorithm. For example, create a Get Connection Info (GCI) step with a script that retrieves and stores a one-time token for a REST or SOAP call.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Text of the credential value.</td>
</tr>
</tbody>
</table>

This example shows the retrieval of a credential value.

```javascript
var RequestAuthGCISigner = Class.create();
RequestAuthGCISigner.prototype = Object.extend(new RequestAuthInternal(), {
    initialize: function() {
        RequestAuthInternal.prototype.initialize.call(this);
    },
    generateAuth: function(authAPI) {
        var requestData = authAPI.getHttpRequestData();
        // get credentials
        var credential = authAPI.getAuthCredential();
        var username = credential.getAttribute("user_name");
        var password = credential.getAttribute("password");
        var httpRequestSignedData = new sn_auth.HttpRequestAuthedData();
        var directive = requestData.getDirective();
        // set CREDENTIAL_VALUE
        httpRequestSignedData.setCredentialValue(username+":"+password);
        httpRequestSignedData.setDirective("CREDENTIAL_VALUE");
        httpRequestSignedData.setStatus("SUCCESS");
        gs.info("The Credential Value is " + httpRequestSignedData.getCredentialValue());
        return httpRequestSignedData;
    },
    type: 'RequestAuthGCISigner'
});
```
Output:

The Credential Value is someUsername:somePassword

HttpRequestAuthedData - getDirective()

Returns whether the signature is applied to the request in the header, as a query parameter, or as a credential value.

By default, the system applies the signature to the header. You can apply the signature as a query parameter or credential value by using the setDirective() API.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String| Whether the signature is applied to the header, as a query parameter, or as a credential value. Values include:  
  • HEADER: The signature is applied to the request header.  
  • QUERY: The signature is applied as a query parameter.  
  • CREDENTIAL_VALUE: The signature is applied as a credential value. |

```javascript
// Define HttpRequestData
var endpoint = "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);

//Get AuthCredential
```
var credential = new
sn_cc.StandardCredentialsProvider().getAuthCredentialByID("5b61c16f73533300f662cfff8fa6a74b");

// Create RequestAuthAPI and sign the request
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var output = signingAPI.generateAuth();
output.setDirective("header");

gs.info("Signature applied to: " + output.getDirective());

Output:

Signature applied to: HEADER

HttpRequestAuthedData - getHeaderMap()

Returns an object containing the headers that were included when the request was signed.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Object | Key-value pairs that define all of the headers included when the request was signed. Each key-value pair includes these parts:
  • key: String. Name of the header.
  • value: String. Value of the header. |

// Define the HttpRequestData object
var endpoint= "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addHeader('x-amz-acl' , 'public-read' );
// Get an AuthCredential object
var credential = new sn_cc.StandardCredentialsProvider().getAuthCredentialByID("5b61c16f73533300f662cffe8f6a74b");

// Create the RequestAuthAPI object and sign the request
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var output = signingAPI.generateAuth();

// get the signed response detail
var authH = headerM["Authorization"];
var xamzdateH = headerM["X-Amz-Date"];
var content256 = headerM["x-amz-content-sha256"];

gs.debug('authH:' + authH );
gs.debug('xamzdateH:' + xamzdateH );
gs.debug('content256:' + content256 );

Output:

*** Script: [DEBUG] authH:AWS4-HMAC-SHA256
Credential=lsjfljljflillfwek/20191127/us-east-1/s3/aws4_request,
SignedHeaders=host;x-amz-acl;x-amz-content-sha256;x-amz-date,
Signature=5c349011324910c34596ba7abbd10e07c4127774049f8953418bd7bed7d02b90

*** Script: [DEBUG] xamzdateH:20191127T175210Z
*** Script: [DEBUG] content256:UNSIGNED-PAYLOAD

HttpRequestAuthedData - getQueryMap()

Returns an object containing the query parameters included when the request was signed.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Object | Key-value pairs that define the query parameters included when the request was signed. Each key-value pair includes these parts:
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• key:</td>
<td>String. Name of the query parameter.</td>
</tr>
<tr>
<td>• value:</td>
<td>String. Value of the query parameter.</td>
</tr>
</tbody>
</table>

```javascript
// Define the HttpRequestData object
var endpoint= "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addQueryParam('api_version', 'v2');
httpRequestData.addQueryParam('limit', '1000');
httpRequestData.addQueryParam('offset', '1');

// Get an AuthCredential object
var credential = new sn_cc.StandardCredentialsProvider().getAuthCredentialByID("5b61c16f73533300f662cff8faf6a74b");

// Create the RequestAuthAPI object and sign the request
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var output = signingAPI.generateAuth();

// Get the signed response
queryMap = output.getQueryMap();

gs.info("API Version is: " + queryMap["api_version"]);```

Output:

API Version is: v2

**HttpRequestAuthedData - getStatus()**

Returns the status of the request signing.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The status of the request signing.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• SUCCESS: The system successfully signed the request.</td>
</tr>
<tr>
<td></td>
<td>• FAIL: The system failed to sign the request.</td>
</tr>
<tr>
<td></td>
<td>• SKIPPED: The system skipped signing because the information was incomplete.</td>
</tr>
</tbody>
</table>

```javascript
// Define the HttpRequestData object
var endpoint = "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);

// Get an AuthCredential object
var credential = new sn_cc.StandardCredentialsProvider().getAuthCredentialByID("5b61c16f73533300f662cfffaf6a74b");

// Create the RequestAuthAPI object and sign the request
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var output = signingAPI.generateAuth();

// Get the signed status
gs.info("Status is: " + output.getStatus());
```

Output:

```
Status is: SUCCESS
```
HttpRequestAuthedData - getStatusMessage()

Returns a detailed message about the request signing. If the request signing fails, this method returns the error message.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Message about the request signing. If the request signing fails, returns one of these error messages:</td>
</tr>
</tbody>
</table>

- **credential is not valid**: Check the AuthCredential object you used to sign the request. Make sure an authentication algorithm is associated with the credential. For more information, see [Configure an authentication algorithm](#).

- **Request data is empty. Request auth is skipped**: Check the HttpRequestData object because required information may be missing.

- **Can't find script includes scope**: Verify that the authentication algorithm record and associated script includes used to sign the request are in the correct scope.

```javascript
// Define the HttpRequestData object
var endpoint= "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addQueryParam('api_version', 'v2');
httpRequestData.addQueryParam('limit', '1000');
httpRequestData.addQueryParam('offset', '1');

// Get AuthCredential object and set an attribute
var credential = new sn_auth.AuthCredential();
```
credential.setAttribute("user_name", "admin");
credential.setAttribute("password", "admin");

// Create the RequestAuthAPI object and sign the request
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var output = signingAPI.generateAuth();

// Get the signed status
gs.info("Status message is: " + output.getStatusMessage());

Output:

Status message is: credential is not valid.

HttpRequestAuthedData - setCredentialValue(String credential_value)

Sets a credential value for the HttpRequestAuthedData object.

Use a credential value to store a cookie, signature, or other value needed for an authentication algorithm. For example, create a Get Connection Info (GCI) step with a script that retrieves and stores a one-time token for a REST or SOAP call.

Use the setCredentialValue() method to set the Credential Value data pill for REST, SOAP, and GCI steps in Flow Designer.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>credential_value</td>
<td>String</td>
<td>Text of the credential value.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to set a credential value for a GCI step.

```javascript
var RequestAuthGCISigner = Class.create();
RequestAuthGCISigner.prototype = Object.extend(new RequestAuthInternal(), {
    initialize: function() {
        RequestAuthInternal.prototype.initialize.call(this);
    },
    generateAuth: function(authAPI) {
        var requestData = authAPI.getHttpRequestData();
```
// get credentials
var credential = authAPI.getAuthCredential();
var username = credential.getAttribute("user_name");
var password = credential.getAttribute("password");
var httpRequestSignedData = new sn_auth.HttpRequestAuthedData();
var directive = requestData.getDirective();
var b64data = GlideStringUtil.base64encode(username+":"+password);

// set CREDENTIAL_VALUE
httpRequestSignedData.setCredentialValue("Basic "+b64data);
httpRequestSignedData.setDirective("CREDENTIAL_VALUE");
httpRequestSignedData.setStatus("SUCCESS");
return httpRequestSignedData;
}

type: 'RequestAuthGCISigner'
});

HttpRequestAuthedData - setDirective(String directive)
Defines whether to apply the signature to the signed request in the header, as a query parameter, or as a credential value.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>directive</td>
<td>String</td>
<td>Whether to apply the signature to the header, query parameter, or credential value of the signed request. Values include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• query: Applies signature as a query parameter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• header: Applies signature to the request header.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• credential_value: Applies signature to the credential value.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

// Define HttpRequestData
var endpoint= "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);

// Get AuthCredential
var credential = new sn_cc.StandardCredentialsProvider().getAuthCredentialByID("5b61c16f73533300f662c8f8f6a74b");

// Create RequestAuthAPI and sign the request
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var output = signingAPI.generateAuth();
output.setDirective("header");

HttpRequestAuthedData - setStatus(String status)
Sets the status of the request signing.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>String</td>
<td>The status of the request signing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• success: The system successfully signed the request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• fail: The system failed to sign the request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• skipped: The system skipped signing because the information was incomplete.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

// Define the HttpRequestData object
var endpoint = "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);

// Get an AuthCredential object
var credential = new sn_cc.StandardCredentialsProvider().getAuthCredentialByID("5b61c16f73533300f662c0f8bfa6a74b");

// Create the RequestAuthAPI object and sign the request
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var output = signingAPI.generateAuth();
output.setStatus("fail");

// Get the signed status
gs.info("Status is: " + output.getStatus());

Output:

Status is: FAIL

HttpRequestAuthedData - setStatusMessage(String statusMessage)
Sets a status message for the request signing.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>statusMessage</td>
<td>String</td>
<td>Message to set for the request signing status.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

// Define the HttpRequestData object
var endpoint= "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);

// Get an AuthCredential object
var credential = new sn_cc.StandardCredentialsProvider().getAuthCredentialByID("5b61c16f73533300f662cff8faf6a74b");

// Create the RequestAuthAPI object and sign the request
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var output = signingAPI.generateAuth();
output.setStatusMessage("The request was successfully signed.");

// Get the signed status message
gs.info("Status message is: " + output.getStatusMessage());

Output:

Status message is: The request was successfully signed.

HttpRequestData - Scoped

The HttpRequestData API provides methods to build and manipulate a REST request before applying a signature and sending it to an endpoint.

Generate outbound signing requests using these APIs in the following order:

1. HttpRequestData: Build the API request.
2. AuthCredential: Create a credential object or update an existing one. Use the credential to sign the request through the RequestAuthAPI class.
3. RequestAuthAPI: Sign the request and return an HttpRequestAuthedData object.
4. HttpRequestAuthedData: Get information about the signed request.
5. GlideHTTPRequest: Send the signed request.

Before using these APIs, you must configure an authentication algorithm to sign the request and associate it with the credential used to authenticate the request.

Use this API in scoped scripts with the sn_auth namespace identifier. You can instantiate this class using the constructor, or you can return an HttpRequestData object from the getHttpRequestData() method in the RequestAuthAPI class.
HttpRequestData - HttpRequestData()
Instantiates an HttpRequestData object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

var httpRequestData = new sn_auth.HttpRequestData();

HttpRequestData - addHeader(String key String value)
Adds a header to the HttpRequestData object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the HTTP header.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value of the HTTP Header.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example creates a REST request payload.

```javascript
var endpoint= "https://s3.amazonaws.com";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addHeader('x-amz-acl', 'public-read' );
```

HttpRequestData - addQueryParam(String key, String value)
Adds a query parameter to the HttpRequestData object.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the query parameter.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value of the query parameter.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var endpoint= "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod('put');
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addQueryParam('api_version', 'v2');
```

**HttpRequestData - deleteHeader(String key)**

Removes a header from the HttpRequestData object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the HTTP header to remove.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var endpoint= "https://s3.amazonaws.com";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
```
HttpRequestData.setHttpMethod('put');
var content = "Action=SendMessage&MessageBody=This is a test message";
HttpRequestData.setContent(content);
HttpRequestData.deleteHeader('x-amz-acl');

**HttpRequestData - getConnectionAliasID()**

Returns the sys_id of the Connection & Credential alias associated with the IntegrationHub REST step.

You can pass the sys_id to the ConnectionInfoProvider API to retrieve information about the connection.

ℹ️ **Note:** You can only use this method in an IntegrationHub REST step. For more information, see REST step.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the alias record from the Connection &amp; Credential Aliases [sys_alias] table associated with the request.</td>
</tr>
</tbody>
</table>

```javascript
var httpRequestData = new sn_auth.HttpRequestData();
var id = httpRequestData.getConnectionAliasID();

var credential = new sn_cc.ConnectionInfoProvider().
var connectionInfo = credential.getConnectionInfo(id);
```

**HttpRequestData - getConnectionExtendedAttribute(String name)**

Returns the value of a connection attribute associated with the Connection & Credential alias associated with IntegrationHub REST step.

ℹ️ **Note:** You can only use this method in an IntegrationHub REST step. For more information, see REST step.

For more information about connection attributes, see Create connection attributes for IntegrationHub.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the connection attribute.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the connection attribute.</td>
</tr>
</tbody>
</table>

This example returns the value of a connection attribute set on the Connection & Credential alias associated with the REST step.

```javascript
var httpRequestData = new sn_auth.HttpRequestData();
gs.info("API Version is: " +
    httpRequestData.getConnectionExtendedAttribute('api_version'));
```

HttpRequestData - getContent()

Returns the content of the request in the HttpRequestData object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Content of the request.</td>
</tr>
</tbody>
</table>

```javascript
var endpoint= "https://s3.amazonaws.com";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod('put');
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addHeader('x-amz-acl', 'public-read');
var ct = httpRequestData.getContent();
```
Output:

```
Action=SendMessage&MessageBody=This is a test message
```

**HttpRequestData - getDate()**

Returns the date when the request was signed.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The difference between January 1, 1970 UTC and when the request was signed. Unit: Milliseconds</td>
</tr>
</tbody>
</table>

```javascript
var endpoint = "https://s3.amazonaws.com";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.addQueryParam("list-type", "2");
httpRequestData.setService('s3');
httpRequestData.setHttpMethod('get');

gs.info("Date: "+httpRequestData.getDate());
```

Output:

```
Date: 1349333576093
```

**HttpRequestData - getDirective()**

Returns whether the signature is applied to the request in the header or as a query parameter.

By default, the system applies the signature to the header. You can apply the signature as a query parameter using the `setDirective()` API.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Whether the signature is applied to the header or as a query parameter. Values include: • HEADER: The signature is applied to the request header. • QUERY: The signature is applied as a query parameter.</td>
</tr>
</tbody>
</table>

```javascript
var endpoint= "https://s3.amazonaws.com";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.addQueryParam("list-type", "2");
httpRequestData.setService('s3');
httpRequestData.setHttpMethod('get');
gs.info( "Signature applied to: " + httpRequestData.getDirective() );
```

Output:

Signature applied to: HEADER

HttpRequestData - getEndpoint()

Returns the endpoint set for the request.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Endpoint set for the request.</td>
</tr>
</tbody>
</table>
var endpoint = "https://s3.amazonaws.com";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.addQueryParam("list-type", "2");
httpRequestData.setService('s3');
httpRequestData.setHttpMethod('get');
httpRequestData.setDirective("query");
gs.info( "Endpoint is: " + httpRequestData.getEndpoint() );

Output:

Endpoint is: https://s3.amazonaws.com

HttpRequestData - getExpiry()

Returns the time in seconds before the signature expires starting when the request was signed using the generateAuth() method in the RequestAuthAPI class.

The expiration is commonly set by the third-party service you are sending the request to, which overrides any value set by the setExpiry() method. For example, if the expiration is set to 900 seconds by the third-party service and you use the setExpiry() method to set the expiration to 300 seconds, the getExpiry() method returns 900 seconds.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Time before the signature expires. Unit: Seconds</td>
</tr>
</tbody>
</table>

var endpoint = "https://s3.amazonaws.com";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.addQueryParam("list-type", "2");
httpRequestData.setService('s3');
httpRequestData.setHttpMethod('get');
httpRequestData.setDirective("query");
httpRequestData.setExpiry(300);
Signature expires in: 300

HttpRequestData - getHeader(String key)
Returns the value associated with the given HTTP header.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of HTTP header.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the given HTTP header.</td>
</tr>
</tbody>
</table>

Header value is: public-read

HttpRequestData - getHeaderMap()
Returns an object containing the headers included in the request.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Object | Key-value pairs that define all of the headers associated with the request. Each key-value pair includes these parts:  
  • Key: String. Name of the header.  
  • Value: String. Value of the header. |

```javascript
var endpoint = "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod('put');
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addHeader('x-amz-acl', 'public-read');
httpRequestData.addHeader('content-type', 'application/json');

var map = httpRequestData.getHeaderMap();

for(var x in map) {
    var y = map[x]
    gs.info(x, y);
}
```

Output:

```javascript
*** Script: x-amz-acl
*** Script: content-type
```

**HttpRequestData - getHost()**

Returns the host used in the request.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Host used in the request.</td>
</tr>
</tbody>
</table>

```javascript
var endpoint = "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setHost('myHost')
httpRequestData.setService('s3')
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod('put');
var content = "Action=SendMessage\MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addHeader('x-amz-acl', 'public-read');

gs.info("Host is: " + httpRequestData.getHost());
```

Output:

Host is: myHost

**HttpRequestData - getHttpMethod()**

Returns the HTTP method used in the request.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>HTTP method used in the request.</td>
</tr>
</tbody>
</table>
var endpoint= "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod('put');
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addHeader('x-amz-acl', 'public-read');

gs.info("HTTP method is: " + httpRequestData.getHttpMethod());

Output:

HTTP method is: PUT

HttpRequestData - getQueryParam(String key)

Returns the value of the specified query parameter.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the query parameter to get the value of.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the query parameter.</td>
</tr>
</tbody>
</table>

var endpoint= "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod('put');
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addQueryParam('api_version', 'v2');

gs.info("API version is: " + httpRequestData.getQueryParam('api_version'));

Output:
**HttpRequestData - getQueryParamMap()**

Returns an object containing the query parameters included in the request.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Object | Key-value pairs that define the query parameters associated with the request. Each key-value pair includes these parts:  
• Key: String. Name of the query parameter.  
• Value: String. Value of the query parameter. |

```javascript
var endpoint = "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod('put');
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addQueryParam('api_version', 'v2');
httpRequestData.addQueryParam('limit', '1000');
httpRequestData.addQueryParam('offset', '1');

var map = httpRequestData.getQueryParamMap();
gs.info("api_version is: " + map['api_version']);
```

### Output:

`api_version is: v2`
HttpRequestData - getRegion()

Returns the region for the request that will be used in signing.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the region.</td>
</tr>
</tbody>
</table>

```javascript
var endpoint = "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod('put');
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);

gs.info("Region is: " + httpRequestData.getRegion());
```

Output:

Region is: us-east-1

HttpRequestData - getService()

Returns the service defined in the request.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Service defined in the request.</td>
</tr>
</tbody>
</table>
var endpoint = "https://s3.amazonaws.com";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');

gs.info("Service is: " + httpRequestData.getService());

**Output:**

```
Service is: s3
```

**HttpRequestData - setContent(String content)**

Defines the content to send in the request.

† **Note:** This method is not supported in script includes or MID server scripts.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>content</td>
<td>String</td>
<td>Content to send in the request.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var endpoint = "https://s3.amazonaws.com";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod('put');
var content = "This is a test message";
httpRequestData.setContent(content);

**HttpRequestData - setDate(Number date)**

Defines the date of the request used for signing.

Use this method only when testing whether the signature is created correctly based on a timestamp. In production, the signing request uses the time that you generated the signature; manually setting this value may cause the signing request to fail.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| date   | Number| The difference between January 1, 1970 UTC and the date used to sign the request.  
        |       | Unit: Milliseconds                                                          |

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var endpoint = "https://s3.amazonaws.com";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.addQueryParam("list-type", "2");
httpRequestData.setService('s3');
httpRequestData.setHttpMethod('get');
httpRequestData.setDate(1349333576093);
```

### HttpRequestData - setDirective(String directive)

Defines whether to apply the signature to the request in a header or as a query parameter.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| directive | String | Whether to apply the signature to the header or as a query parameter.        
        |        | Values include:                                                            |
        |        | • query: Applies signature as a query parameter.                            |
        |        | • header: Applies signature to the request header.                          |
        |        | Default: header                                                            |
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var endpoint= "https://s3.amazonaws.com";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.addQueryParam("list-type", "2");
httpRequestData.setService('s3');
httpRequestData.setHttpMethod('get');
httpRequestData.setDirective("query");
```

HttpRequestData - setEndpoint(String endpoint)

Defines the endpoint to send the request to.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>endpoint</td>
<td>String</td>
<td>Endpoint to send the request to. The endpoint can include query parameters, path, and version.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var endpoint="https://third-party-endpoint/path-parameters/version";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
```

HttpRequestData - setExpiry(Number expiry)

Sets the amount of time before the signature expires starting when the request is signed using the `generateAuth()` method in the RequestAuthAPI class.

The expiration is commonly set by the third-party service you are sending the request to, which overrides any value set by the `setExpiry()` method. For example, if the expiration is set to 900 seconds by the third-party service and you...
use the `setExpiry()` method to set the expiration to 300 seconds, the `getExpiry()` method returns 900 seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>expiry</td>
<td>Number</td>
<td>Amount of time before the signature expires.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unit: Seconds</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var endpoint= "https://s3.amazonaws.com";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.addQueryParam("list-type", "2");
httpRequestData.setService('s3');
httpRequestData.setHttpMethod('get');
httpRequestData.setExpiry(300);
```

HttpRequestData - `setHost(String host)`
Definitions the host used in the request.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>String</td>
<td>Name of the HTTP request host.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var endpoint= "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setHost('myHost');
```
HttpRequestData.setService('s3');
HttpRequestData.setRegion('us-east-1');
HttpRequestData.setHttpMethod('put');
var content = "Action=SendMessage&MessageBody=This is a test message";
HttpRequestData.setContent(content);
HttpRequestData.addHeader('x-amz-acl', 'public-read');

**HttpRequestData - setHttpMethod(String httpMethod)**

Sets the HTTP method to use in the request.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>httpMethod</td>
<td>String</td>
<td>HTTP method to use in the request.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var endpoint= "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
HttpRequestData.setRegion('us-east-1');
HttpRequestData.setHttpMethod('put');

**HttpRequestData - setRegion(String region)**

Defines the region to use for the request in signing.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>region</td>
<td>String</td>
<td>The region to use to sign the request. Available values are determined by the third-party API's requirements.</td>
</tr>
</tbody>
</table>
### HttpRequestData - setService(String service)

Defines the service for the request.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>service</td>
<td>String</td>
<td>The service for the request.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### i18NV3 - Client

Provides methods to get and format translated messages.

The i18N methods are accessed using the g_i18n global object.

#### i18NV3 - getMessage(String msgKey, Function callback)

Retrieves a translated message.
If the specified string exists in the database for the current language, then the translated message is returned. If the specified string does not exist for the current language, then the English version of the string is returned. If the string does not exist at all in the database, then the ID itself is returned.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msgKey</td>
<td>String</td>
<td>The message to be retrieved.</td>
</tr>
<tr>
<td>callback</td>
<td>Function</td>
<td>The function to be called when the message has been retrieved. The callback function has one argument, a string that is the translated message.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### i18NV3 - getMessages(Array msgKeys, Function callback)

Retrieves a set of messages.

If the specified string exists in the database for the current language, then the translated message is returned. If the specified string does not exist for the current language, then the English version of the string is returned. If the string does not exist at all in the database, then the ID itself is returned.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msgKeys</td>
<td>Array</td>
<td>An array of keys specifying the messages to be retrieved.</td>
</tr>
<tr>
<td>callback</td>
<td>Function</td>
<td>The function to be called when the messages have been retrieved. The callback function has one argument, an object containing key-value pairs, where key is the requested message key, and the value is the translated string.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

i18NV3 - format(String message, Object map)

Formats a string containing named tokens with values from a map.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The message to have the tokens added.</td>
</tr>
<tr>
<td>map</td>
<td>Object</td>
<td>The map of name/value pairs to replace in the message.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The formatted string</td>
</tr>
</tbody>
</table>

// Returns: "The rich young ruler was very very rich"
nowapi.g_i18n.format("The {p1} {p2} {p3} was very very {p1}", {p1: "rich", p2: "young", p3: "ruler"});

ICalUtilSNC - Global

The ICalUtilSNC API provides functions to generate iCalendar compliant events.

ICalUtilSNC - formatICalComponent(Array arrEvents)

Generates a formatted VCALENDAR component.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>arrEvents</td>
<td>Array</td>
<td>Contains the properties that make up a VCALENDAR component.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Contains properties needed to form a VCALENDAR.</td>
</tr>
</tbody>
</table>

ICalUtilSNC - formatICalEvent(Array arrEvent, Boolean useAlarm)

Generates a formatted VEVENT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>arrEvent</td>
<td>Array</td>
<td>Contains the individual properties that make up a VEVENT.</td>
</tr>
<tr>
<td>useAlarm</td>
<td>Boolean</td>
<td>If true, adds a VALARM to the VEVENT.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The formatted VEVENT.</td>
</tr>
</tbody>
</table>

ICalUtilSNC - formatRecurringRule (Object ruleObj)

Formats the RRULE property for a VEVENT.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ruleObj</td>
<td>Object</td>
<td>Contains the properties for the RRULE property.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The formatted RRULE property.</td>
</tr>
</tbody>
</table>

ICalUtilSNC - getDateFromScheduleDateTime (GlideScheduleDateTime scheduleDateTime)

Returns the date from the GlideScheduleDateTime.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheduleDateTime</td>
<td>GlideScheduleDateTime</td>
<td>A GlideScheduleDateTime representation of the date and time.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Formatted date.</td>
</tr>
</tbody>
</table>

**ICalUtilSNC - getSDT(String sdtStr, String timeZone)**

If provided with a formatted date time string, it returns a GlideScheduleDateTime object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sdtStr</td>
<td>String</td>
<td>Formatted ScheduleDateTime.</td>
</tr>
<tr>
<td>timeZone</td>
<td>String</td>
<td>If provided, will be used for the generated GlideScheduleDateTime.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideScheduleDateTime</td>
<td>A GlideScheduleDateTime representation of the sdtStr.</td>
</tr>
</tbody>
</table>

**ICalUtilSNC - getTimeFromScheduleDateTime (GlideScheduleDateTime scheduleDateTime)**

Returns the time from the GlideScheduleDateTime.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheduleDateTime</td>
<td>GlideScheduleDateTime</td>
<td>A GlideScheduleDateTime representation of the date and time.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Formatted time.</td>
</tr>
</tbody>
</table>

### IdentificationEngine - Scoped

The `IdentificationEngine` API uses the Identification and Reconciliation framework to minimize the creation of duplicate configuration items (CIs) and to reconcile CI attributes by only accepting information from authorized data sources when updating the Configuration Management Database (CMDB).

When using this class in a scoped application, use the `sn_cmdb` namespace identifier.

### IdentificationEngine - createOrUpdateCI(String source, String input)

Inserts or updates configuration items and non-Configuration Management Database (CMDB) configuration items (classes not extending from `cmdb_ci`) in the CMDB based on identification and reconciliation rules. Use this API instead of updating the CMDB directly.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>String</td>
<td>Required. A JSON formatted string of configuration items to add or update.</td>
</tr>
</tbody>
</table>


### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items</td>
<td>Array</td>
<td>Array of objects that define the items to add or update.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;items&quot;:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;className&quot;:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;display_values&quot;:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>{Object}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;internal_id&quot;:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;lookup&quot;: [Array]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;related&quot;: [Array]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;settings&quot;: {Object}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;sys_object_source_info&quot;: {Object}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;values&quot;: {Object}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>input.items.className</td>
<td>String</td>
<td>Required. Class/table name, sys_class_name, of the configuration item (CI) to create or update. This value can be any CMDB class/table such as cmdb_ci_linux_server or cmdb_ci_win_server.</td>
</tr>
<tr>
<td>input.items.display_values</td>
<td>Object</td>
<td>Reference fields to create or update for this related item as name-value pairs, where the name is the field name and the value is the referenced display value. If you want to use the sys_id instead of the display value for reference fields, pass the information in the <code>input.items.lookup.values</code> parameter instead of in this parameter.</td>
</tr>
</tbody>
</table>
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>input.items.internal_id</strong></td>
<td>String</td>
<td>Unique item identifier for the associated payload. This can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td><strong>input.items.lookup</strong></td>
<td>Array</td>
<td>Identifies the top-level item containing the lookup (lookup-based identification). These records are used to identify the configuration item based on a lookup table that has a reference back to cmdb_ci. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For example:

```json
"lookup": [
  {
    "className": String,
    "internal_id": String,
    "sys_object_source_info": {Object},
    "values": {Object}
  }
]
```

| **input.items.lookup.className** | String | Required. Class/table name, sys_class_name, of the configuration item (CI) to create or update. |

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.lookup.internal_id</td>
<td>String</td>
<td>Unique item identifier for the associated payload. This can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info</td>
<td>Object</td>
<td>Unique CI identifier for a specific source.</td>
</tr>
</tbody>
</table>

```
"source_feed": "String",
"source_name": "String",
"source_native_key": "String"
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.lookup.sys_object_source_info.source_feed</td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide the name of the feed sending this item. The data source generates this feed name. It can be any string that uniquely identifies the source feed.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_name</td>
<td>String</td>
<td>Data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_native_key</td>
<td>String</td>
<td>Unique key/id for the item from the source.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date and time that the item was scanned. Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>input.items.lookup.values</td>
<td>Object</td>
<td>Fields to create or update for this related item as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced sys_id. If you want to use the display value instead of the sys_id for reference fields, pass this information in a display_values object instead of in the values object. Field names and types depend on the fields selected, as:</td>
</tr>
<tr>
<td>input.items.related</td>
<td>Array</td>
<td>Reference to the top-level item that contains the related list. Rules in the Related Entry [cmdb_related_entry]</td>
</tr>
</tbody>
</table>
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| input.items.related.className              | String  | Required. Class/table name, sys_class_name, of the configuration item (CI) to create or update. This value can be any CMDB class/table, such as `cmdb_software_instance` or `cmdb_key_value`.
| input.items.related.internal_id           | String  | Unique item identifier for the associated payload. This can be any value, but must be unique within the payload. |
| input.items.related.sys_object_source_info| Object  | Unique CI identifier for a specific source. |
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>input.items.related.sys_object_source_info.source_feed</code></td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide the name of the feed sending this item. The data source generates this feed name. It can be any string that uniquely identifies the source feed.</td>
</tr>
<tr>
<td><code>input.items.related.sys_object_source_info.source_name</code></td>
<td>String</td>
<td>Data source of the CI information. This value must be one of the values defined for the discovery_source field of the Configuration Item <code>[cmdb_ci]</code> table.</td>
</tr>
<tr>
<td><code>input.items.related.sys_object_source_info.source_native_key</code></td>
<td>String</td>
<td>Unique key/id for the item from the source. The data source generates this key. It can be any string that is unique to the item.</td>
</tr>
<tr>
<td><code>input.items.related.sys_object_source_info.source_recency_timestamp</code></td>
<td>String</td>
<td>UTC date and time that the item was scanned. Format: <code>YYYY-MM-DD hh:mm:ss</code></td>
</tr>
<tr>
<td><code>input.items.related.values</code></td>
<td>Object</td>
<td>Fields to create or update for this related item as name/value pairs, where the name is the field name and the value is the field value.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name. For a reference field, the value must be the referenced sys_id. If you want to use the display value instead of the sys_id for reference fields, pass this information in a display_values object instead of in the values object. Field names and types depend on the fields selected as:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>input.items.settings</td>
<td>Object</td>
<td>Parameters that define the types of updates that are permitted:</td>
</tr>
<tr>
<td>input.items.settings.skipReclassificationRestrictionRules</td>
<td>Boolean</td>
<td>Flag that indicates whether IRE should not run the:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>input.items.settings.updateWithoutDowngrade</strong></td>
<td>Boolean</td>
<td>Flag that indicates whether update and downgrade are both permitted for this item. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Update the item but downgrade is not permitted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Both item update and downgrade are permitted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td><strong>input.items.settings.updateWithoutUpgrade</strong></td>
<td>Boolean</td>
<td>Flag that indicates whether update and upgrade are both permitted for this item. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Update the item but upgrade is not permitted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Both item update and upgrade are permitted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td><strong>input.items.settings.updateWithoutSwitch</strong></td>
<td>Boolean</td>
<td>Flag that indicates whether the item can be updated and the class switched. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: The item can be updated and the class switched.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: The item cannot be updated and the class cannot be switched.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>input.items.sys_object_source_info</strong></td>
<td>Object</td>
<td>Unique CI identifier for a specific source.</td>
</tr>
<tr>
<td><strong>input.items.sys_object_source_info.source_feed</strong></td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide the feed senting this item. The data source generates this feed name. It can be any string that uniquely identifies the source feed.</td>
</tr>
<tr>
<td><strong>input.items.sys_object_source_info.source_name</strong></td>
<td>String</td>
<td>Data source of the CI information. This value must be one of the values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td><strong>input.items.sys_object_source_info.source_native_key</strong></td>
<td>String</td>
<td>Unique key/id for the item from the source.</td>
</tr>
</tbody>
</table>

- true: Update the item but class switching is not permitted.
- false: Both item update and class switching are permitted.

Default: false

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.

Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>input.items.sys_object_source_info.source_recency_timestamp</strong>&lt;br&gt;The data source generates this key. It is any string that is unique to the item.</td>
</tr>
<tr>
<td></td>
<td>String</td>
<td>UTC date and time that the item was scanned.&lt;br&gt;<strong>Format:</strong> Year-MM-DD hh:mm:ss</td>
</tr>
</tbody>
</table>
|      |            | **input.items.values**<br>Fields to create or update for this related item as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced sys_id.<br>If you want to use the display value instead of the sys_id for reference fields, pass this information in a `display_values` object instead of in the `values` object.<br>Field names and types depend on the fields selected by the user, such as:
```
"values": {
  "host_name": "String",
  "ip_address": "String",
  "name": "String",
  "os_name": "String",
  "sys_class_name": "String"
}
```
|      | Object     | **input.referenceItems**<br>Array of objects that define references between items in the input payload. |
|      | Array      | Array of objects that define references between items in the input payload. |
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>input.referencetItems.referenced</code></td>
<td>String</td>
<td>The <code>internal_id</code> defined for the item being referenced by another item.</td>
</tr>
<tr>
<td><code>input.referencetItems.referencedBy</code></td>
<td>String</td>
<td>The <code>internal_id</code> defined for the item that references another item.</td>
</tr>
<tr>
<td><code>input.referencetItems.referenceField</code></td>
<td>String</td>
<td>Name of the reference field in the class/table for the <code>referencedBy</code> item.</td>
</tr>
<tr>
<td><code>input.relations</code></td>
<td>Array</td>
<td>Array of objects that specify relationships between items in the payload.</td>
</tr>
</tbody>
</table>

An object in this array can use either of two formats:

- The object can define a relationship between two top-level items (only) using name-value pairs, with values representing item indexes from the payload items array.

```json
"relations": [
  {
    "child": Number,
    "parent": Number,
    "sys_rel_source_info": Object
  }
]
```
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;type&quot;</td>
<td>String</td>
<td>The object can define a relationship between any two items, including top-level, related, or lookup items, using <code>parent_id</code> and <code>child_id</code> key/value pairs, with values representing <code>internal_id</code> values defined for those items.</td>
</tr>
</tbody>
</table>

- The object can define a relationship between any two items, including top-level, related, or lookup items, using `parent_id` and `child_id` key/value pairs, with values representing `internal_id` values defined for those items.

```json
"relations": [
  {
    "child_id": "String",
    "parent_id": "String",
    "sys_rel_source_info": {Object},
    "type": "String"
  }
]
```

- `input.relations.child`: Number Integer index of the CI object in the array that represents the child in the relationship (items, items.related, or items.lookup).

- `input.relations.child_id`: String The `internal_id` of the child item in the relationship (items, items.related, or items.lookup).

- `input.relations.parent`: Number Integer index of the parent item in the relationship (items, items.related, or items.lookup).
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>input.relations.parent_id</code></td>
<td>String</td>
<td>The <em>internal_id</em> of the parent item in the relation (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td><code>input.relations.sys_rel_source_info</code></td>
<td>Object</td>
<td>Discovery information for the relationship. For non-dependency relationships, this information is saved in the Relationship Sources table (not persisted for <code>identifyCIEnhanced()</code> or <code>identifyCI()</code> methods.)</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Data type:</strong></td>
</tr>
</tbody>
</table>
|                                           |          | `{  "source_name": "String",    
|                                           |          |    "source_feed": "String"  }                                              |
| `input.relations.sys_rel_source_info.source_name` | String   | Discovery source name.                                                                                                                     |
|                                           |          | Default: Discovery source passed in the API method parameter.                                                                            |
| `input.relations.sys_rel_source_info.source_feed` | String   | Any string that is a sub-discovery/scan within the discovery source.                                                                      |
|                                           |          | Default: ‘UNKNOWN’ is stored in the `source_feed` column when creating a record in the `sys_rel_source` table.                             |
| `input.relations.type`                    | String   | Type of relationship that exists between the parent and child items.                                                                       |
|                                           |          | Must be a name field value from items, items.related, or items.lookup.                                                                   |
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>the CI Relationship Type</td>
<td>[cmdb_rel_type]</td>
<td>Table of the CI Relationship Type</td>
</tr>
<tr>
<td>source</td>
<td>String</td>
<td>Identifies the data source of the CI information. This value must be one of</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;</td>
<td>JSON formatted string that is a list of results for the configuration items in the input string. Each result string is in the format 'items: [{}], relations: [{}]', where each item within the items and relations lists contains name-value pairs. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems</td>
<td>No values are currently returned.</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedRelations</td>
<td>No values are currently returned.</td>
</tr>
<tr>
<td>&lt;String&gt;.items</td>
<td>List of objects that describe created or updated CIs. Data type: Array</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;identificationAttempts&quot;: [Array], &quot;identifierEntrySysId&quot;: &quot;String&quot;, &quot;info&quot;: [Array], &quot;inputIndices&quot;: [Array], &quot;maskedAttributes&quot;: [Array], &quot;operation&quot;: &quot;String&quot;, &quot;relatedItems&quot;: [Array], &quot;relatedSysIds&quot;: [Array], &quot;sysId&quot;: &quot;String&quot;</td>
<td></td>
</tr>
</tbody>
</table>

**<String>.items.additionalRelatedItems**

List of JSON objects that provides information about additional lookup and related items that were processed but not provided as part of the input payload. These items come from partial payloads.

This information is not currently returned.

Data type: Array

```
"additionalRelatedItems": [ 
  { 
    "className": "String", 
    "inputIndices": [Array], 
    "operation": "String", 
    "sysId": "String" 
  }
]
```

**<String>.items.additionalRelatedItems.className**

Class/table name (sys_class_name) of the CI that was created or updated.

Data type: String

**<String>.items.additionalRelatedItems.inputIndices**

Index of the corresponding input item. For top-level items, it is a list of integers. For related items, it is a list of related item indices.

Data type: Array
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>or lookup items, it is list of JSON objects.</td>
<td></td>
</tr>
<tr>
<td>Data type: Array of Number or Array of objects</td>
<td></td>
</tr>
<tr>
<td>&quot;inputIndices&quot;: [</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;mainIndex&quot;: Number,</td>
<td></td>
</tr>
<tr>
<td>&quot;subIndex&quot;: Number</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.additionalRelatedItems.inputIndices.mainIndex</td>
<td>Index value from the request body <strong>items</strong> array that corresponds to the CI parent of the additional related item.</td>
</tr>
<tr>
<td>Data type: Number</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.additionalRelatedItems.inputIndices.subIndex</td>
<td>Index value from the request body <strong>items.lookup</strong> array that corresponds to the additional related item.</td>
</tr>
<tr>
<td>Data type: Number</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.additionalRelatedItems.operation</td>
<td>Type of operation. Possible values:</td>
</tr>
<tr>
<td>Data type: String</td>
<td>• INSERT: New CI was inserted into the database.</td>
</tr>
<tr>
<td></td>
<td>• NO_CHANGE: No CI changes were made.</td>
</tr>
<tr>
<td></td>
<td>• UPDATE: Existing CI was updated.</td>
</tr>
<tr>
<td>&lt;String&gt;.items.additionalRelatedItems.sysId</td>
<td>Sys_id of the additional related items.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.className</code></td>
<td>Class/table name (sys_class_name) of the CI that was created or updated. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.duplicateIndices</code></td>
<td>List of indexes of CIs that are duplicates of the current item. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.errorCount</code></td>
<td>Number of errors. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.errors</code></td>
<td>Array of objects in which each object describes an error encountered while processing this CI. Data type: Array</td>
</tr>
<tr>
<td>&quot;errors&quot;: [</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>]</td>
<td></td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.errors.error</code></td>
<td>Type of error encountered while processing the CI. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.errors.message</code></td>
<td>Error message associated with the error. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.identificationAttempts</code></td>
<td>List of attempts that were made to identify the CIs. Data type: Array</td>
</tr>
<tr>
<td>&quot;identificationAttempts&quot;: [</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;attemptResult&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>]</td>
<td></td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;attributes&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;hybridEntryCiAttributes&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;identifierName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;searchOnTable&quot;: &quot;String&quot;</td>
</tr>
</tbody>
</table>

`<String>.items.identificationAttempts.attemptResult` Results of the attempt to identify the CI.

Possible values:

- **MATCHED**: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.

- **MULTI_MATCH**: Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.

- **NO_MATCH**: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.

- **SKIPPED**: Identification not attempted. The attributes required for this identifier rule table search were not provided, so the rule was not applied.

Data type: String
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.identificationAttempts.attributes</code></td>
<td>List of CI identifier entry attributes that were used during the identification process. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.identificationAttempts.hybridEntryCiAttributes</code></td>
<td>No values are currently returned.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.identificationAttempts.identifierName</code></td>
<td>Identifier rule used for this CI identification attempt. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.identificationAttempts.searchOnTable</code></td>
<td>Name of the table searched during the identification process. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.identifierEntrySysId</code></td>
<td>Sys_id for the identifier rule used to identify the CI. Located in the Identifier Entry [cmdb_identifier_entry] table. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.info</code></td>
<td>List of objects that contain additional information about the processing of the item. Data type: Array</td>
</tr>
</tbody>
</table>

- **Attributes**:  
  ```json
  "attributes": [  
    "serial_number": "String",  
    "serial_number_type": "String"  
  ]
  ```
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| <String>.items.info.code | Reclassification type that was skipped. Possible values:  
- SKIPPED_CLASS_SWITCH  
- SKIPPED_CLASS_DOWNGRADE  
- SKIPPED_CLASS_UPGRADE  
Data type: String |
| <String>.items.info.message | Message that provides additional insights into the reason for skipping the reclassification.  
Data type: String |
| <String>.items.info.ruleSysId | Sys_id of the reclassification restriction rule that was matched. Applicable only when IRE skips reclassification due to reclassification restriction rule. This value is empty if the reclassification is skipped due to a payload or global flag.  
Data type: String |
| <String>.items.inputIndices | Index values for CIs from the request body **items** array that correspond to this CI.  
Data type: Array |
| <String>.items.maskedAttributes | List of attributes whose update by a non-authoritative data source was skipped as defined by the Reconciliation Rules. |
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.relatedItems</td>
<td>List of JSON objects that provides information about the processed related items. Data type: Array</td>
</tr>
</tbody>
</table>

```
"relatedItems": [ 
  { 
    "className": "String",
    "errors": [Array],
    "errorCount": Number,
    "inputIndices": [Array],
    "operation": "String",
    "sysId": "String"
  }
]
```

| <String>.items.relatedItems.className | Class/table name (sys_class_name) of the related item. Data type: String |
| <String>.items.relatedItems.errors | List of errors that occurred during processing. Data type: Array |

```
"errors": [ 
  { 
    "error": "String",
    "message": "String"
  }
]
```

<p>| &lt;String&gt;.items.relatedItems.errorCount | Number of errors detected during processing. Data type: Number |
| &lt;String&gt;.items.relatedItems.inputIndices | Indexes of the corresponding related items. Data type: Array or Numbers |</p>
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| "inputIndices": [  
  {  
    "mainIndex": Number,  
    "subIndex": Number  
  }  
] | Integer value from the request body `items` array that corresponds to the CI parent of the related item. Data type: Number |
| `<String>.items.relatedItems.inputIndices.mainIndex` | Integer value from the request body `items.lookup` array that corresponds to the related item. Data type: Number |
| `<String>.items.relatedItems.inputIndices.subIndex` | Type of operation. Possible values:  
  • INSERT: New CI was inserted into the database.  
  • NO_CHANGE: No CI changes were made.  
  • UPDATE: Existing CI was updated.  
Data type: String |
| `<String>.items.relatedSysIds` | List of the sys_id values for related items (table lookup items) from the request body `items.lookup` array. Notable values:  
  • null: No sys_id was identified for this related item.  
Data type: Array |
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.sys_id</code></td>
<td>Sys_id of the CI that was updated or created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations</code></td>
<td>List of JSON objects that provides information about processed relations.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>&quot;relations&quot;: [</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;className&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;errorCount&quot;: Number,</td>
<td></td>
</tr>
<tr>
<td>&quot;inputIndices&quot;: [Array],</td>
<td></td>
</tr>
<tr>
<td>&quot;operation&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sysId&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations.className</code></td>
<td>Sys_class_name of this dependent relationship CI. Only supported value:</td>
</tr>
<tr>
<td></td>
<td>• cmdb_rel_ci: CI Relationship table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations.errorCount</code></td>
<td>Number of errors.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations.inputIndices</code></td>
<td>Indexes of the corresponding input relations.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations.operation</code></td>
<td>Type of operation performed. Possible values:</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• INSERT</td>
<td></td>
</tr>
<tr>
<td>• UPDATE</td>
<td></td>
</tr>
<tr>
<td>• NO_CHANGE</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

<String>.relations.sysId

Sys_id of the dependent relationship CI.
Data type: String

The following shows how to reclassify a configuration item.

```javascript
var payload = {
    "items": [
        {
            "className": "cmdb_ci_win_server",
            "values": {
                "short_description": "Linux server description",
                "name": "Linux Server 1"
            }
        }
    ]
};

var input = JSON.stringify(payload);
var output = sn_cmdb.IdentificationEngine.createOrUpdateCI('ServiceNow', input);

gs.info(JSON.stringify(JSON.parse(output), null, 2));
```

Output:

```json
{
    "items": [
        {
            "className": "cmdb_ci_linux_server",
            "operation": "NO_CHANGE",
            "sysId": "440577800f321010150efc91ff76e94",
            "identifierEntrySysId": "556eb250c3400200d8d4bea192d3ae92",
            "identificationAttempts": []
        }
    ]
}
```
"attributes": [],
"info": "sys_object_source SKIPPED",
"identifierName": "",
"attemptResult": "SKIPPED",
"hybridEntryCiAttributes": []
},

{ "attributes": [
    "serial_number",
    "serial_number_type"
],
"identifierName": "Hardware Rule",
"attemptResult": "SKIPPED",
"searchOnTable": "cmdb_serial_number",
"hybridEntryCiAttributes": []
},

{ "attributes": [
    "serial_number"
],
"identifierName": "Hardware Rule",
"attemptResult": "SKIPPED",
"searchOnTable": "cmdb_ci_hardware",
"hybridEntryCiAttributes": []
},

{ "attributes": [
    "name"
],
"identifierName": "Hardware Rule",
"attemptResult": "MATCHED",
"searchOnTable": "cmdb_ci_hardware",
"hybridEntryCiAttributes": []
}
],
"info": [
    
    "message": "CI Reclassification not allowed from class: [cmdb_ci_linux_server] to
[cmdb_ci_win_server] by a reclassification restriction rule",
    "code": "SKIPPED_CLASS_SWITCH",
    "ruleSysId": "b3d4b3800f321010150efc91ff767eab"
  ]
},
"errorCount": 0,
The following shows how to update a configuration item.

```javascript
var payload = {
    "items": [
        {
            "className": "cmdb_ci_win_server",
            "values": {
                "chassis_type": "Desktop",
                "os": "Windows 2012 R2 Datacenter",
                "name": "Windows2012Server1",
                "serial_number": "0000-0011-1690-8730-8636-5722-52",
                "cpu_count": "1"
            },
            "lookup": [
                {
                    "values": {
                        "valid": "true",
                        "serial_number": "0000-0011-1690-8730-8636-5722-52",
                        "serial_number_type": "bios"
                    },
                    "className": "cmdb_serial_number"
                },
                {
                    "values": {
                        "valid": "true",
                        "serial_number": "3311-9736-4988-9744-1749-4183-41",
                        "serial_number_type": "chassis"
                    },
                    "className": "cmdb_serial_number"
                }
            ]
        }
    ]
};
```
"internal_id": "16777219",
"sys_object_source_info": {
  "source_feed": "SN Discovery Feed 1",
  "source_name": "ServiceNow",
  "source_native_key": "16777219",
  "source_recency_timestamp": "2019-10-18 08:31:23"
},
{
  "className": "cmdb_ci_spkg",
  "values": {
    "name": "Windows 2012 R2 Datacenter",
    "key": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL"
  }
},
"related": [
  {
    "internal_id": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL|16777219",
    "values": {
      "name": "Windows 2012 R2 Datacenter-SAMLABVM52"
    },
    "className": "cmdb_software_instance",
    "sys_object_source_info": {
      "source_feed": "SN Discovery Feed 1",
      "source_name": "ServiceNow",
      "source_native_key": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL|16777219"
    }
  },
  {
    "className": "cmdb_ci_app_server_tomcat",
    "values": {
      "running_process_key_parameters": "/opt/OV/nonOV/tomcat/b/temp org.apache.catalina.startup.Bootstrap start",
      "install_directory": "/opt/OV/nonOV/tomcat/b",
      "name": "Tomcat@hpom9:3443",
      "server_port": "8006",
      "sys_class_name": "cmdb_ci_app_server_tomcat"
    },
    "internal_id": "tomcat_id"
  }
],
"relations": [
  {
    "parent_id": "tomcat_id",
    "internal_id": "16777219"
  }
]
"child_id": "16777219",
"type": "Runs on::Runs"
},

"referenceItems": [ { "referenceField": "installed_on",
"referenced": "16777219",
"referencedBy": "Microsoft Windows Server 2012 R2 Datacenter:::NULL\|16777219"
} ];

var input = JSON.stringify(payload);
var output = sn_cmdb.IdentificationEngine.createOrUpdateCI('ServiceNow', input);

gs.info(JSON.stringify(JSON.parse(output), null, 2));

Output:

{ "items": [ { "className": "cmdb_ci_win_server",
"operation": "INSERT",
"sysId": "d56ab6eadbd510102f67dfea5e96194e",
"relatedSysIds": [ "dd6af62adb1910102f67dfea5e96197f",
"996af62adb1910102f67dfea5e961980"
],
"relatedItems": [ { "errors": [],
"operation": "INSERT",
"errorCount": 0,
"mergedPayloadIds": [],
"warningCount": 0,
"sysId": "dd6af62adb1910102f67dfea5e96197f",
"markers": [],
"inputIndices": [ { "mainIndex": 0,
"subIndex": 0
} ],
"className": "cmdb_serial_number" } ] }
{"errors": [],
"operation": "INSERT",
"errorCount": 0,
"mergedPayloadIds": [],
"warningCount": 0,
"sysId": "996af62adb1910102f67dfea5e961980",
"markers": [],
"inputIndices": [
{
  "mainIndex": 0,
  "subIndex": 1
}
],
"className": "cmdb_serial_number"
],
"identifierEntrySysId": "Unknown",
"identificationAttempts": [
{
  "info": "sys_object_source NO_MATCH",
  "identifierName": "",
  "attemptResult": "NO_MATCH",
  "attributes": [],
  "hybridEntryCiAttributes": []
},
{
  "identifierName": "Hardware Rule",
  "attemptResult": "NO_MATCH",
  "attributes": ["serial_number",
                  "serial_number_type"],
  "searchOnTable": "cmdb_serial_number",
  "hybridEntryCiAttributes": []
},
{
  "identifierName": "Hardware Rule",
  "attemptResult": "NO_MATCH",
  "attributes": ["serial_number"],
  "searchOnTable": "cmdb_ci_hardware",
...}
"hybridEntryCiAttributes": [],

{
   "identifierName": "Hardware Rule",
   "attemptResult": "NO_MATCH",
   "attributes": [
      "name"
   ],
   "searchOnTable": "cmdb_ci_hardware",
   "hybridEntryCiAttributes": []
},

{
   "identifierName": "Hardware Rule",
   "attemptResult": "SKIPPED",
   "attributes": [
      "mac_address",
      "name"
   ],
   "searchOnTable": "cmdb_ci_network_adapter",
   "hybridEntryCiAttributes": []
}
],
"errorCount": 0,
"mergedPayloadIds": [],
"warningCount": 0,
"markers": [],
"inputIndices": [0]
},
{
   "className": "cmdb_ci_spkg",
   "operation": "INSERT",
   "sysId": "116af62adb1910102f67dfe6e5961981",
   "relatedSysIds": [
      "6d6af62adb1910102f67dfe961984"
   ],
   "relatedItems": [
      {
         "errors": [],
         "operation": "INSERT",
         "errorCount": 0,
         "mergedPayloadIds": [],
         "warningCount": 0,
      }
   ]
}
"sysId": "6d6af62ad6b910102f67dfe5e961984",
"markers": [],
"inputIndices": [
  {
    "mainIndex": 1,
    "subIndex": 0
  }
],
"className": "cmdb_software_instance",
"identifierEntrySysId": "Unknown",
"identificationAttempts": [
  {
    "info": "sys_object_source SKIPPED",
    "identifierName": "",
    "attemptResult": "SKIPPED",
    "attributes": [],
    "hybridEntryCiAttributes": []
  },
  {
    "identifierName": "Software",
    "attemptResult": "NO_MATCH",
    "attributes": [
      "key"
    ],
    "searchOnTable": "cmdb_ci_spkg",
    "hybridEntryCiAttributes": []
  }
],
"errorCount": 0,
"mergedPayloadIds": [],
"warningCount": 0,
"markers": [],
"inputIndices": [1]
],
"className": "cmdb_ci_app_server_tomcat",
"operation": "INSERT",
"sysId": "e96af62ad6b910102f67dfe5e961985",
"identifierEntrySysId": "Unknown",
"identificationAttempts": [
Identify a dependent CI.

```javascript
var payload = {
    items: [
        {className: 'cmdb_ci_web_server',
         values: {
             name: 'apache linux den 200',
             running_process_command: 'xyz',
             running_process_key_parameters: 'abc',
         }
    ]
}
```
```javascript
var input = JSON.stringify(payload);
var output = sn_cmdb.IdentificationEngine.createOrUpdateCI('ServiceNow', input);

gs.info(JSON.stringify(JSON.parse(output), null, 2));
```

Output:

```json
{
  "items": [
    {
      "className": "cmdb_ci_web_server",
      "operation": "INSERT",
      "sysId": "b9bb766adb1910102f67d6e5e961962",
      "identifierEntrySysId": "Unknown",
      "identificationAttempts": [
        {
          "info": "sys_object_source SKIPPED",
          "identifierName": "",
          "attemptResult": "SKIPPED",
          "attributes": [],
          "hybridEntryCiAttributes": []
        },
        {
          "info": "sys_object_source SKIPPED",
          "identifierName": "",
          "attemptResult": "SKIPPED",
          "attributes": [],
          "hybridEntryCiAttributes": []
        }
      ],
      "errorCount": 0,
      "mergedPayloadIds": [],
      "warningCount": 0,
      "markers": [],
      "inputIndices": [0]
    }
  ],
  "identifierEntrySysId": "Unknown",
  "identificationAttempts": [
    {
      "info": "sys_object_source SKIPPED",
      "identifierName": "",
      "attemptResult": "SKIPPED",
      "attributes": [],
      "hybridEntryCiAttributes": []
    }
  ]
}
```
{"className": "cmdb_ci_linux_server",
"operation": "INSERT",
"sysId": "a5bb766adb1910102f67dfea5e96195b",
"identifierEntrySysId": "Unknown",
"identificationAttempts": [
{
    "info": "sys_object_source SKIPPED",
    "identifierName": "",
    "attemptResult": "SKIPPED",
    "attributes": [],
    "hybridEntryCiAttributes": []
},
{
    "identifierName": "Hardware Rule",
    "attemptResult": "SKIPPED",
    "attributes": [
        "serial_number",
        "serial_number_type"
    ],
    "searchOnTable": "cmdb_serial_number",
    "hybridEntryCiAttributes": []
},
{
    "identifierName": "Hardware Rule",
    "attemptResult": "SKIPPED",
    "attributes": [
        "serial_number"
    ],
    "searchOnTable": "cmdb_ci_hardware",
    "hybridEntryCiAttributes": []
},
{
    "identifierName": "Hardware Rule",
    "attemptResult": "NO_MATCH",
    "attributes": [
        "name"
    ],
    "searchOnTable": "cmdb_ci_hardware",
    "hybridEntryCiAttributes": []
}]
}
Identify an independent CI with lookup-based identification.

```javascript
var payload = {
    items: [
        {
            className: 'cmdb_ci_netgear',
            values: {name: 'ny8500-nbxs08',
                     ports: '1200'},
            lookup: {
                   className: 'cmdb_serial_number',

```
```
var input = JSON.stringify(payload);
var output = sn_cmdb.IdentificationEngine.createOrUpdateCI('ServiceNow', input);

gs.info(JSON.stringify(JSON.parse(output), null, 2));
```

Output:

```
{
  "items": [
    {
      "className": "cmdb_ci_netgear",
      "operation": "INSERT",
      "sysId": "787c7e6adb1910102f67dfea5e96196e",
      "relatedSysIds": [
        "f47c7e6adb1910102f67dfea5e961977",
        "3c7c7e6adb1910102f67dfea5e961977"
      ],
      "relatedItems": [
        {
          "errors": [],
          "operation": "INSERT",
          "errorCount": 0,
          "mergedPayloadIds": [],
          "warningCount": 0,
          "sysId": "f47c7e6adb1910102f67dfea5e961977",
          "markers": [],
          "inputIndices": [
            {
              "mainIndex": 0,
              "subIndex": 0
            }
          ],
          "className": "cmdb_serial_number"
        }
      ]
    }
  ]
```

"mergedPayloadIds": [],
"warningCount": 0,
"sysId": "3c7c7e6adb1910102f67dfe5e961977",
"markers": [],
"inputIndices": [
{
    "mainIndex": 0,
    "subIndex": 1
}
],
"className": "cmdb_serial_number"
],
"identifierEntrySysId": "Unknown",
"identificationAttempts": [
{
    "info": "sys_object_source SKIPPED",
    "identifierName": "",
    "attemptResult": "SKIPPED",
    "attributes": [],
    "hybridEntryCiAttributes": []
},
{
    "identifierName": "Hardware Rule",
    "attemptResult": "NO_MATCH",
    "attributes": [
        "serial_number",
        "serial_number_type"
    ],
    "searchOnTable": "cmdb_serial_number",
    "hybridEntryCiAttributes": []
},
{
    "identifierName": "Hardware Rule",
    "attemptResult": "SKIPPED",
    "attributes": [
        "serial_number"
    ],
    "searchOnTable": "cmdb_ci_hardware",
    "hybridEntryCiAttributes": []
},
{
    "identifierName": "Hardware Rule",
    "attemptResult": "NO_MATCH",
    "attributes": []
}
IdentificationEngine - createOrUpdateCIEnhanced(String source, String input, Object options)

Inserts or updates configuration items and non-Configuration Management Database (CMDB) configuration items (classes not extending from cmdb_ci) in the CMDB based on identification and reconciliation rules. Use this API instead of updating the CMDB directly.

In addition to providing the functionality of the createOrUpdateCI() method, this method also supports:
• Handling partial payloads
• Handling partial commits
• Removing duplicate items within a payload
• Generating output summaries

For additional information on IRE and more detailed explanations of the data used by this method, see Identification and Reconciliation (IRE).

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>String</td>
<td>Required. A JSON formatted string of configuration items to add or update.</td>
</tr>
<tr>
<td>input.items</td>
<td>Array</td>
<td>Array of objects that define the items to add or update.</td>
</tr>
</tbody>
</table>

#### input.items

- className: String
- display_values: Object
- internal_id: String
- lookup: Array
- related: Array
- settings: Object
- sys_object_source_info: Object
- values: Object

#### input.items.className

- Required. Class/table name, sys_class_name, of the items.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.display_values</td>
<td>Object</td>
<td>Reference fields to create or update for this related item as name-value pairs, where the name is the field name and the value is the referenced display value. If you want to use the <code>sys_id</code> instead of the display value for reference fields, pass the information in the <code>input.items.lookup.values</code> parameter instead of in this parameter. Reference field names depend on the fields selected as:</td>
</tr>
<tr>
<td>input.items.internal_id</td>
<td>String</td>
<td>Unique identifier for the associated payload. This can be any value but must be unique within the payload.</td>
</tr>
<tr>
<td>input.items.lookup</td>
<td>Array</td>
<td>Identifies the top-level item containing the lookup (lookup-based identification). These</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.lookup.className</td>
<td>String</td>
<td>Required. Class/table name, sys_class_name, of the configuration item (CI) to create. This value can be any CMDB class/table, such as cmdb_serial_number or cmdb_ci.</td>
</tr>
<tr>
<td>input.items.lookup.internal_id</td>
<td>String</td>
<td>Unique lookup item identifier for the associated payload. This can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info</td>
<td>Object</td>
<td>Defines a unique CI identifier for a specific data source. Different sources may have different name-value pairs.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_feed</td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide the name of the feed sending this item. The data source generates this field. It can be any string that uniquely identifies the source feed.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_name</td>
<td>String</td>
<td>Data source information of the CI. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_native_key</td>
<td>String</td>
<td>Unique key-id for the item from the source. The data source generates this key. It can be any string that is unique to the item.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date/time that the item was scanned. Format: YYYY-MM-DD hh:mm:ss.</td>
</tr>
<tr>
<td>input.items.lookup.values</td>
<td>Object</td>
<td>Field information as name-value pairs, where the name is the field name. When updating reference fields, the value must be the referenced sys_id.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field names and types</td>
<td></td>
<td>Field names and types depend on the fields selected by the user, such as:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>values</em>: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>serial_number</em>: String,</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>serial_number_type</em>: String,</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>valid</em>: String,</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>ip_address</em>: String,</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>mac_address</em>: String,</td>
</tr>
<tr>
<td>input.items.related</td>
<td>Array</td>
<td>Reference to the top-level item in the related list. Rules in the Related</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Entry [cmdb_related_entry] define what type of records can be in this</td>
</tr>
<tr>
<td></td>
<td></td>
<td>array. These records are used to add items based on a related table that</td>
</tr>
<tr>
<td></td>
<td></td>
<td>has a reference to the CI that is being identified. The related table may</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or may not extend cmdb_ci. These records are not used to identify the CI.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>related</em>: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>className</em>: String,</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>internal_id</em>: String,</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>sys_object_source_info</em>: {Object},</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>values</em>: {Object}</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.related.className</td>
<td>String</td>
<td>Required. Class name, sys_class_name, of the configuration item (CI) to create or update. This value can be any CMDB class/table, such as cmdb_software_instance or cmdb_key_value.</td>
</tr>
<tr>
<td>input.items.related.internal_id</td>
<td>String</td>
<td>Unique identifier for this related item in this payload. Can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info</td>
<td>Object</td>
<td>Object that makes up a unique CI identifier for a specified data source. Different data sources may have different name-value pairs for the same CI.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info.source_feed</td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide the name of the feed sending this item. The data source generates this feed name.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.related.sys_object_source_info.source_name</td>
<td>String</td>
<td>any string that uniquely identifies the source feed.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info.source_native_key</td>
<td>String</td>
<td>Unique key/ID from the source for the related item. The data source generates this key. It can be any string that is unique to the item.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date and time that the item was scanned. Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>input.items.related.values</td>
<td>Object</td>
<td>Fields to create or update for this related item as name/value pairs, where the name is the field name. For reference fields, the value must be the referenced sys_id. If you want to use the display value instead of the sys_id for reference fields, pass this information in a display_values object instead of in the values object. Field names and types depend on the fields selected by the user, such as:</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.settings</td>
<td>Object</td>
<td>Parameters that define the types of updates that are permitted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>input.items.settings.skipReclassificationRestrictionRules</td>
<td>Boolean</td>
<td>Flag that indicates whether IRE should not run the Reclassification Restriction rule that matches the class for the payload item. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Skip running the rule.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Run the rule.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>input.items.settings.updateWithoutDowngrade</td>
<td>Boolean</td>
<td>Flag that indicates whether update and downgrade are both permitted for this item. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Update and downgrade are both permitted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Update is permitted, downgrade is not permitted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| input.items.settings.updateWithoutSwitch | Boolean | Flag that indicates whether the item can be updated and the class switched. Valid values:  
- true: Update the item but class switching is not permitted.  
- false: Both item update and class switching are permitted.  
Default: false |
| input.items.settings.updateWithoutUpgrade | Boolean | Flag that indicates whether update and upgrade are both permitted for this item. Valid values:  
- true: Update the item but upgrade is not permitted.  
- false: Both item update and upgrade are permitted.  
Default: false |
| input.items.sys_object_source_info | Object | Unique CI identifier for a specific source. |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.

2996
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>input.items.sys_object_source_info.source_feed</code></td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide a name for the feed sending this item. The data source generates this feed name. It can be any string that uniquely identifies the source feed.</td>
</tr>
<tr>
<td><code>input.items.sys_object_source_info.source_name</code></td>
<td>String</td>
<td>Data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the <code>cmdb_ci</code> table.</td>
</tr>
<tr>
<td><code>input.items.sys_object_source_info.source_native_key</code></td>
<td>String</td>
<td>Unique key/id for the item from the source. The data source generates this key. It can be any string that is unique to the item.</td>
</tr>
<tr>
<td><code>input.items.sys_object_source_info.source_recency_timestamp</code></td>
<td>String</td>
<td>UTC date and time that the item was scanned. Format: <code>YYYY-MM-DD hh:mm:ss</code></td>
</tr>
<tr>
<td><code>input.items.values</code></td>
<td>Object</td>
<td>Fields to create or update for this related item as name/value pairs, where the name is the field name. For a reference field, the value must be the name of the target object.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>the referenced sys_id.</td>
<td></td>
<td>If you want to use the display value instead of the sys_id for reference fields, pass this information in a <code>display_values</code> object instead of in the <code>values</code> object. Field names and types depend on the fields selected as:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;values&quot;:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;host_name&quot;: &quot;String&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;ip_address&quot;: &quot;String&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;name&quot;: &quot;String&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;os_name&quot;: &quot;String&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;sys_class_name&quot;: &quot;String&quot;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**input.referencelItems**

Array

Array of objects that define references between items in the input:

```
{"reference": {
  "referenced": "String",
  "referencedBy": "String",
  "referenceField": "String"
}}
```

**input.referencelItems.referenced**

String

The `internal_id` defined for the item being referenced by another item.

**input.referencelItems.referencedBy**

String

The `internal_id` defined for the item that references another item.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.referenceltems.referenceField</td>
<td>String</td>
<td>Name of the reference field in the class/table for the referencedBy item.</td>
</tr>
</tbody>
</table>
| input.relations                        | Array    | Array of objects that specify relationships between items in the input payload. An object can use either of the following formats:

- The object can define a relationship between two top-level items using `parent` and `child` name-value pairs, with values representing item indexes from the payload:

  ```json
  "relations": [
    {
      "child": Number,
      "parent": Number,
      "sys_rel_source_info": {Object},
      "type": "String"
    }
  ]
  ```

- The object can define a relationship between any two items, including top-level, related, or lookup items, using `parent_id` and `child_id` keys, with values representing `internal_id` values defined for those items:

  ```json
  "relations": {
    "parent_id": Number,
    "child_id": Number
  }
  ```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.relations.child</td>
<td>Number</td>
<td>Integer index of the CI object that defines the child in the relationship. (items, items.related, or items.lookup.)</td>
</tr>
<tr>
<td>input.relations.child_id</td>
<td>String</td>
<td>The internal_id of the child item in the relationship. (items, items.related, or items.lookup.)</td>
</tr>
<tr>
<td>input.relations.parent</td>
<td>Number</td>
<td>Integer index of the parent item in the relationship. (items, items.related, or items.lookup.)</td>
</tr>
<tr>
<td>input.relations.parent_id</td>
<td>String</td>
<td>The internal_id of the parent item in the relationship. (items, items.related, or items.lookup.)</td>
</tr>
<tr>
<td>input.relations.sys_rel_source_info</td>
<td>Object</td>
<td>Discovery source information for the relationship. For non-dependency relationships, this information is saved in the Relationship Sources [sys_rel_source] table (not persisted for identifyCIEnhanced() or identifyCI() methods.)</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>input.relations.sys_rel_source_info.source_name</td>
<td>String</td>
<td>Discovery source name. Default: Discovery source passed in the API method parameter.</td>
</tr>
<tr>
<td>input.relations.sys_rel_source_info.source_feed</td>
<td>String</td>
<td>Any string that is a sub-discovery/scan within the discovery source. Default: 'UNKNOWN' is stored in the source_feed column when creating a record in the sys_rel_source table.</td>
</tr>
<tr>
<td>input.relations.type</td>
<td>String</td>
<td>Type of relationship that exists between the parent and child items. Must be a name field value from the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional, but {} or null must be passed-in. Options to enable or disable features.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options.deduplicate_payloads</td>
<td>Boolean</td>
<td>Flag that indicates whether duplicate items are merged or considered errors. Valid values: true: Merge duplicate items.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Consider duplicate items as errors. Default: true</td>
</tr>
<tr>
<td>options.generate_summary</td>
<td>Boolean</td>
<td>Flag that indicates whether the returned results contain summary information. The details of the returned summary can be accessed.</td>
</tr>
</tbody>
</table>

**Note:**
- If `partial_payloads` is set to true, both `partial_commits` and `deduplicate_payloads` are enabled, even if they are set to false, as those features are essential for partial payloads functionality.

```
"options": {
  "deduplicate_payloads": "Boolean",
  "generate_summary": "Boolean",
  "partial_commits": "Boolean",
  "partial_payloads": "Boolean"
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.summary</td>
<td>&lt;String&gt;</td>
<td>return results table. Valid values: • true: Include summary information. • false: Do not include summary information. Default: false</td>
</tr>
<tr>
<td>options.partial_commits</td>
<td>Boolean</td>
<td>Flag that indicates whether partial commit support is enabled. For additional information on partial commits, see Enhanced IRE features. Valid values: • true: Partial commit enabled. • false: Partial commit disabled. Default: true</td>
</tr>
<tr>
<td>options.partial_payloads</td>
<td>Boolean</td>
<td>Flag that indicates whether partial payload support is enabled. For additional information on partial payloads, see Enhance IRE features and Create an IRE data source rule. Valid values: • true: Partial payload enabled. • false: Partial payload disabled. Default: true</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>String</td>
<td>Data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;</td>
<td>JSON formatted string that is a list of results for the configuration items in the input string.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;additionalCommittedItems&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;className&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems</td>
<td>List of CIs that were committed during the IRE processing of the current payload, but were not present in the current input payload.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;errors&quot;</td>
<td>Object that describes errors encountered while processing this additional CI. Data type: Object</td>
</tr>
<tr>
<td>&quot;error&quot;</td>
<td>Type of error encountered while processing the additional CI. Data type: String</td>
</tr>
<tr>
<td>&quot;message&quot;</td>
<td>Error message encountered while processing the additional CI. Data type: String</td>
</tr>
<tr>
<td>&quot;sysId&quot;</td>
<td>String representing the system identifier of the additional CI. Data type: String</td>
</tr>
</tbody>
</table>

<String>.additionalCommittedItems.className

Sys_class_name of this additional CI. Data type: String

<String>.additionalCommittedItems.errorCount

Number of errors encountered while processing this additional CI. Data type: Number

<String>.additionalCommittedItems.errors

Array of objects that describes errors encountered while processing this additional CI. Data type: Array

<String>.additionalCommittedItems.errors.error

Type of error encountered while processing the additional CI. Data type: String

<String>.additionalCommittedItems.errors.message

Error message encountered while processing the additional CI. Data type: String

<String>.additionalCommittedItems.identificationAttempts

Array of objects in which each object describes an attempt made to identify this additional CI. Data type: Array
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td></td>
<td>`{</td>
</tr>
<tr>
<td></td>
<td><strong>attemptResult</strong>: String</td>
</tr>
<tr>
<td></td>
<td><strong>attributes</strong>: [Array]</td>
</tr>
<tr>
<td></td>
<td><strong>hybridEntryCiAttributes</strong>: [Array]</td>
</tr>
<tr>
<td></td>
<td><strong>identifierName</strong>: String</td>
</tr>
<tr>
<td></td>
<td><strong>searchOnTable</strong>: [Array]</td>
</tr>
<tr>
<td><strong>&lt;String&gt;.additionalCommittedItems.identificationAttempts.attemptResult</strong></td>
<td><strong>Outcome of this additional CI identification attempt.</strong></td>
</tr>
<tr>
<td>Data type: String</td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• MATCHED: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• MULTI_MATCH: Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• NO_MATCH: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• SKIPPED: Identification not attempted. The attributes required for this identifier rule table search were not provided, so the rule was not applied.</td>
</tr>
<tr>
<td><strong>&lt;String&gt;.additionalCommittedItems.identificationAttempts.attributes</strong></td>
<td><strong>Array of CI identifier entry attributes used during this additional CI identification attempt.</strong></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Returns (continued)</td>
<td></td>
</tr>
<tr>
<td><strong>Parameter</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>Attribute names and types depend on the request body data and the identifier in use, such as:</td>
<td></td>
</tr>
<tr>
<td><em>attributes</em>:</td>
<td></td>
</tr>
<tr>
<td><em>serial_number</em>:  &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td><em>serial_number_type</em>:  &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.identificationAttempts.hybridEntryCiAttributes</td>
<td>Array of CI identifier entry attributes used during this additional CI identification attempt.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>Attribute names and types depend on the request body data and the identifier in use, such as:</td>
<td></td>
</tr>
<tr>
<td><em>hybridEntryCiAttributes</em>:</td>
<td></td>
</tr>
<tr>
<td><em>name</em>:  &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td><em>serial_number</em>:  &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.identificationAttempts.identifierName</td>
<td>Identifier rule used for this additional CI identification attempt.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.identificationAttempts.searchOnTable</td>
<td>Name of the table searched for this additional CI identification attempt.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.identifierEntrySysId</td>
<td>Sys_id for the identifier rule used to identify this additional CI.</td>
</tr>
<tr>
<td>Notable values:</td>
<td></td>
</tr>
<tr>
<td>• Unknown: Identification of this additional CI failed. See errors for details.</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.inputIndices</td>
<td>Array of index values for CIs from the request body items array that correspond to this additional CI.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| ![](image-url) | Data type: Array
| *inputIndices:* |
| ![](image-url) | Number |
| ![](image-url) | Array of marker values for internal use. |
| ![](image-url) | Data type: Array |
| *markers:* |
| ![](image-url) | Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this additional CI. |
| ![](image-url) | Data type: Array |
| *mergedPayloadIds:* |
| ![](image-url) | "String" |
| ![](image-url) | Array of operation values for this additional CI. |
| ![](image-url) | Data type: String |
| *operation:* |
| ![](image-url) | "DELETE": An existing CI is removed from the target table. |
| ![](image-url) | "INSERT": The additional CI is inserted into the target table as a new record. |
| ![](image-url) | "NO_CHANGE": No operation is performed for the additional CI. |
| ![](image-url) | "UPDATE": An existing CI in the target table is updated. |
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• UPDATE_WITH_DOWNGRADE: An existing CI in the target table is updated and its class is changed to a more generic class (ancestor class).</td>
<td></td>
</tr>
<tr>
<td>• UPDATE_WITH_SWITCH: An existing CI in the target table is updated and its class is changed to another class which is not an ancestor or descendent class.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE_WITH_UPGRADE: An existing CI in the target table is updated and its class is changed to a more specialized class (descendent class).</td>
<td></td>
</tr>
</tbody>
</table>

**<String>.additionalCommittedItems.sysId**
Sys_id found for this additional CI through identification.
Data type: String
Notable values:
- Unknown: Identification of this additional CI failed. See errors for details.

**<String>.additionalCommittedItems.warnings**
Array of objects that describe a warning encountered while processing this additional CI.
Data type:

```json
  "warnings": [
    {
      "error": "String",
      "message": "String"
    }
  ]
```

**<String>.additionalCommittedItems.warnings.error**
Type of warning encountered while processing this additional CI.
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.additionalCommittedItems.warnings.message</code></td>
<td>Warning message encountered while processing this additional CI. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations</code></td>
<td>Array of objects that describe a dependent relationship CI that was not included in the request body relations list to insert or update. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations.className</code></td>
<td>The sys_class_name of this additional dependent CI. Data type: String Only supported values: • cmdb_rel_ci: The CI Relationship table</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations.errorCount</code></td>
<td>Number of errors encountered while processing this additional dependent CI. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations.errors</code></td>
<td>Array of objects that describe errors encountered while processing this additional dependent CI. Data type: Array</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.additionalCommittedRelations.errors.error</td>
<td>Type of error encountered while processing the additional dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedRelations.errors.message</td>
<td>The error message encountered while processing this additional dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedRelations.inputIndices</td>
<td>Index values for dependent relationship CI objects in the request body that correspond to this additional dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedRelations.markers</td>
<td>Marker values for internal use. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedRelations.mergedPayloadIds</td>
<td>Sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this additional dependent relationship CI. Data type: Array</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;mergedPayloadIds&quot;: [&quot;String&quot;]</td>
<td>-</td>
</tr>
</tbody>
</table>

**<String>.additionalCommittedRelations.operation**

Operation that was performed for the additional dependent relationship CI. Possible values:
- INSERT: The dependent relationship CI is inserted into the target table as a new record.
- INSERT_AS_INCOMPLETE: The dependent relationship CI had errors and is inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.
- INSERT_AS_PARTIAL: The dependent relationship CI had errors and is inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.
- NO_CHANGE: No operation is performed for the dependent relationship CI.
- UPDATE: An existing dependent relationship CI in the target table is updated.

Data type: String

**<String>.hasError**

Flag that indicates whether any item or relation has errors.

Data type: Boolean

**<String>.hasWarning**

Flag that indicates whether any item or relation has warnings.

Data type: Boolean

© 2021 ServiceNow, Inc. All rights reserved.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items</td>
<td>Array of objects that describe the created or updated CIs. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.items.additionalRelatedItems</td>
<td>List of JSON objects that provide information about additional lookup and related items that were processed but not provided as part of the input payload. These items are from partial payloads. Data type: Array</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.additionalRelatedItems.className</code></td>
<td>Class/table name (sys_class_name) of the CI that was created or updated. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.additionalRelatedItems.inputIndices</code></td>
<td>Index values for CIs from the request body items array that correspond to this related item. Data type: Array of Numbers</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.additionalRelatedItems.mergedPayloadIds</code></td>
<td>List of sys_ids of the partial payloads that were merged into the related item. Located in the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.additionalRelatedItems.operation</code></td>
<td>Type of operation. Possible values: INSERT, NO_CHANGE, UPDATE. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.additionalRelatedItems.sysId</code></td>
<td>Sys_id of the CI that was updated or created. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.className</code></td>
<td>Class/table name (sys_class_name) of the CI that was created or updated. Data type: String</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.duplicateIndices</code></td>
<td>List of indexes of CIs that are duplicates of the current item. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.errorCount</code></td>
<td>Number of errors encountered while processing the CI. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.errors</code></td>
<td>Array of objects in which each object describes an error encountered while processing this CI. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.errors.error</code></td>
<td>Type of error encountered while processing the CI. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.errors.message</code></td>
<td>Error message associated with the error. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.identificationAttempts</code></td>
<td>List of attempts that were made to identify the CIs. Data type: Array</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.attemptResult</td>
<td>Results of the attempt to identify the CI. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• MATCHED: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• MULTI_MATCH: Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• NO_MATCH: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• SKIPPED: Identification not attempted. The attributes required for this identifier rule table search were not provided, so the rule was not applied.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.attributes</td>
<td>List of CI identifier entry attributes that were used during the identification process.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>

Attribute names and types depend on the request body data and the identifier in use, such as:
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;items.identificationAttempts.hybridEntryCiAttributes</code></td>
<td>List of CI attributes that were used during the identification process.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>Attribute names and types depend on the request body data and the identifier used.</td>
</tr>
<tr>
<td><code>&lt;String&gt;items.identificationAttempts.identifierName</code></td>
<td>Identifier rule used for this CI identification attempt.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;items.identificationAttempts.searchOnTable</code></td>
<td>Name of the table searched during the identification process.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;items.identifierEntrySysId</code></td>
<td>Sys_id for the identifier used to identify the CI.</td>
</tr>
<tr>
<td></td>
<td>Located in the Identifier Entry table. [cmdb_identifier_entry].</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;items.incompleteSysIds</code></td>
<td>If the item was saved as an incomplete payload, contains the sys_id of the CMDB IRE Incomplete Payloads table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&lt;String&gt;.items.info</td>
<td>List of objects that contains additional information about the processing of the item. Data type: Array.</td>
</tr>
<tr>
<td></td>
<td><strong>info</strong>: [</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&lt;String&gt;.items.maskedAttributes</td>
<td>List of attributes whose update by a non-authoritative data source was skipped as defined by the Reconciliation Rules.</td>
</tr>
<tr>
<td>&lt;String&gt;.items.operation</td>
<td>Operation that took place. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• INSERT: New CI was inserted into the database.</td>
</tr>
<tr>
<td></td>
<td>• INSERT_AS_INCOMPLETE: Item was saved in cmdb_ire_incomplete_payloads table.</td>
</tr>
<tr>
<td></td>
<td>• INSERT_AS_PARTIAL: Item was saved in cmdb_ire_partial_payloads table.</td>
</tr>
<tr>
<td></td>
<td>• NO_CHANGE: No CI changes were made.</td>
</tr>
<tr>
<td></td>
<td>• UPDATE: Existing CI was updated.</td>
</tr>
<tr>
<td></td>
<td>• UPDATE_WITH_DOWNGRADE: CI was updated and the class changed to a more generic class (ancestor class).</td>
</tr>
<tr>
<td></td>
<td>• UPDATE_WITH_SWITCH: CI was updated and the class changed to a class that is not ancestor or descendent.</td>
</tr>
<tr>
<td></td>
<td>• UPDATE_WITH_UPGRADE: CI was updated and the class changed to a more specialized class (descendent class).</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.partialSysIds</code></td>
<td>If the item had errors and was saved as a partial payload, this parameter contains the sys_id of the partial payload record.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems</code></td>
<td>List of JSON objects that describe a related CI (table lookup CI) from the request body <code>items.lookup</code> array.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.className</code></td>
<td>Class/table name (sys_class_name) of the related item.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.errors</code></td>
<td>List of errors that occurred during processing of the related item.</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.errors.error</code></td>
<td>Type of error encountered while processing the related item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.errors.message</code></td>
<td>Error message associated with the error. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.errorCount</code></td>
<td>Number of errors detected while processing the related items. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.incompleteSysIds</code></td>
<td>If the relation had errors and was saved as an incomplete payload, this value is the sys_id of the record in the CMDB IRE Incomplete Payloads table. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.inputIndices</code></td>
<td>Index of the corresponding input item. For top-level items, it is a list of integers. For related or lookup items, it is a list of JSON objects. Data type: Array of Numbers or Array of Objects</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.inputIndices.mainIndex</code></td>
<td>Index value from the request body items array that corresponds to the CI parent of the related item. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.inputIndices.subIndex</code></td>
<td>Index value from the request body items.lookup array that corresponds to the related item. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.mergedPayloadIds</code></td>
<td>List of sys ids of the partial payloads that were merged into the CI during processing. Data type: Array</td>
</tr>
</tbody>
</table>
| `<String>.items.relatedItems.operation` | Operation that took place. Possible values:  
  - INSERT: New related CI was inserted into the database.  
  - INSERT_AS_INCOMPLETE: Item was saved in cmdb_ire_incomplete_payloads table.  
  - INSERT_AS_PARTIAL: Item was saved in cmdb_ire_partial_payloads table.  
  - NO_CHANGE: No related CI changes were made.  
  - UPDATE: Existing related CI was updated.  
  - UPDATE_WITH_DOWNGRADE: Related CI was updated and the class changed to a more generic class (ancestor class).  
  - UPDATE_WITH_SWITCH: Related CI was updated and the class changed. |
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• UPDATE_WITH_UPGRADE: Related CI was updated and the class changed to a more specialized class (descendent class). Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.partialSysIds</td>
<td>If the related item had errors and was saved as a partial payload, this contains a list of the sys_ids of the associated records in the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.sysId</td>
<td>The sys_id of the related item. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.warningCount</td>
<td>Number of warnings encountered when processing the related items. Data type: Number</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.warnings</td>
<td>Array of objects that describes a warning encountered while processing the related items. Data type: Array</td>
</tr>
</tbody>
</table>
| `warnings: [
  {
    "error": "String",
    "message": "String"
  }
]` | Type of warning encountered while processing the related item. Data type: String |
### Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.warnings.message</code></td>
<td>Message associated with the warning. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedSysIds</code></td>
<td>List of the CIs used during lookup-based identification. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.sys_id</code></td>
<td>Sys_id of the CI that was updated or created. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations</code></td>
<td>List of JSON objects that describe a dependent relationship CI from the request body <code>relations</code> array. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations.className</code></td>
<td>Sys_class_name of this dependent relationship CI. Only supported value: <code>cmdb_rel_ci</code>: CI Relationship table. Data type: String</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.relations.errorCount</td>
<td>Number of errors encountered when processing the dependent relationship CI. Data type: Number</td>
</tr>
<tr>
<td>&lt;String&gt;.relations.errors</td>
<td>Array of errors that occurred while processing this dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td>&quot;errors&quot;: [</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>&lt;String&gt;.relations.errors.error</td>
<td>Type of error encountered while processing the dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.relations.errors.message</td>
<td>Error message encountered while processing this dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.relations.incompleteSysIds</td>
<td>If the relation was saved as an incomplete payload, this value is the sys_id of the record in the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.relations.inputIndices</td>
<td>Indexes for the dependent relationship CI objects in the request body relations array that...</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.relations.operation</td>
<td>Type of operation performed. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• INSERT: The dependent relationship CI was inserted into the target table as a new record.</td>
</tr>
<tr>
<td></td>
<td>• INSERT_AS_INCOMPLETE: The dependent relationship CI had errors and was inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.</td>
</tr>
<tr>
<td></td>
<td>• INSERT_AS_PARTIAL: The dependent relationship CI had errors and was inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.</td>
</tr>
<tr>
<td></td>
<td>• NO_CHANGE: No operation was performed.</td>
</tr>
<tr>
<td></td>
<td>• UPDATE: An existing dependent relationship CI in the target table was updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.relations.partialSysIds</td>
<td>If the relationship is saved as a partial payload, this value is the sys_id of the record in the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.relations.sysId</td>
<td>Sys_id of the dependent relationship CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.summary</td>
<td>List of JS object statistics: inserted, updated, and such, per class. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.summary.&lt;class_name&gt;</td>
<td>Statistics for a specific class. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&lt;class_name&gt;:</td>
</tr>
<tr>
<td></td>
<td>- additionalInsertedItemCount: Number</td>
</tr>
<tr>
<td></td>
<td>- errorCount: Number</td>
</tr>
<tr>
<td></td>
<td>- incompleteItemCount: Number</td>
</tr>
<tr>
<td></td>
<td>- insertedItemCount: Number</td>
</tr>
<tr>
<td></td>
<td>- partialItemCount: Number</td>
</tr>
<tr>
<td></td>
<td>- skippedItemCount: Number</td>
</tr>
<tr>
<td></td>
<td>- unchangedItemCount: Number</td>
</tr>
<tr>
<td></td>
<td>- updatedItemCount: Number</td>
</tr>
<tr>
<td></td>
<td>- warningCount: Number</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.partialItemCount</code></td>
<td>Number of items saved in the Partial Payload table [cmdb_ire_partial_payloads]. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.skippedItemCount</code></td>
<td>Number of items that were skipped. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.unchangedItemCount</code></td>
<td>Number of items that had entries but were not modified. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.updatedItemCount</code></td>
<td>Number of items updated. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.warningCount</code></td>
<td>Number of items that generated a warning when processed. Data type: Number</td>
</tr>
</tbody>
</table>

This example shows how to import multiple CI entries.

```javascript
var payload = {
    "items": [{
        "className": "cmdb_ci_win_server",
        "values": {
            "chassis_type": "Desktop",
            "os": "Windows 2012 R2 Datacenter",
            "name": "Windows2012Server1",
            "serial_number": "0000-0011-1690-8730-8636-5722-52",
            "cpu_count": "1"
        },
        "lookup": [{
            "values": {
                "valid": "true",
                "serial_number": "0000-0011-1690-8730-8636-5722-52",
                "serial_number_type": "bios"
            }
        },
        "className": "cmdb_serial_number"
    }
};
```


```json
{
  "values": {
    "valid": "true",
    "serial_number": "3311-9736-4988-9744-1749-4183-41",
    "serial_number_type": "chassis"
  },
  "className": "cmdb_serial_number"
},

"internal_id": "16777219",
"sys_object_source_info": {
  "source_feed": "SN Discovery Feed 1",
  "source_name": "ServiceNow",
  "source_native_key": "16777219",
  "source_recency_timestamp": "2019-10-18 08:31:23"
},

"className": "cmdb_ci_spkg",
"values": {
  "name": "Windows 2012 R2 Datacenter",
  "key": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL"
},

"related": [

  "internal_id": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL\|16777219",
  "values": {
    "name": "Windows 2012 R2 Datacenter-SAMLABVM52"
  },
  "className": "cmdb_software_instance",
  "sys_object_source_info": {
    "source_feed": "SN Discovery Feed 1",
    "source_name": "ServiceNow",
    "source_native_key": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL\|16777219"
  }
},

"settings": {
  "skipReclassification Restriction Rules": "false",
  "updateWithoutDowngrade": "true",
  "updateWithoutUpgrade": "true",
  "updateWithoutSwitch": "true"
}
```
 className: "cmdb.ci.app_server_tomcat",
values: {
  "running_process_key_parameters": "/opt/OV/nonOV/tomcat/b/temp org.apache.catalina.startup.Bootstrap start",
  "install_directory": "/opt/OV/nonOV/tomcat/b",
  "name": "Tomcat@hpom9:3443",
  "server_port": "8006",
  "sys_class_name": "cmdb.ci.app_server_tomcat"
},

"internal_id": "tomcat_id"
},

"relations": [
{
  "parent_id": "tomcat_id",
  "child_id": "16777219",
  "type": "Runs on::Runs"
}
],

"referenceItems": [
{
  "referenceField": "installed_on",
  "referenced": "16777219",
  "referencedBy": "Microsoft Windows Server 2012 R2 Datacenter:::_NULL|16777219"
}
];

var input = JSON.stringify((payload));
var output = sn_cmdb.IdentificationEngine.createOrUpdateCIEnhanced('ServiceNow', input, {});
gs.info(JSON.stringify(JSON.parse(output), null, 2));

Output:

{
  "items": [
  {
    "className": "cmdb.ci.win_server",
    "operation": "NO_CHANGE",
    "sysId": "65d873d2b3a0001028f6eae2c6a8dc2a",
    "relatedSysIds": ["a1d873d2b3a0001028f6eae2c6a8dc32"],
  }
}
"a1d873d2b3a0001028f6eae2c6a8dc33",

"relatedItems": [
{
  "className": "cmdb_serial_number",
  "sysId": "a1d873d2b3a0001028f6eae2c6a8dc32",
  "markers": [],
  "inputIndices": [
    {
      "mainIndex": 0,
      "subIndex": 0
    }
  ]
},
  "className": "cmdb_serial_number",
  "sysId": "a1d873d2b3a0001028f6eae2c6a8dc33",
  "markers": [],
  "inputIndices": [
    {
      "mainIndex": 0,
      "subIndex": 1
    }
  ]
},
"additionalRelatedItems": [],
"identificationAttempts": [],
"errorCount": 0,
"inputIndices": [0]
],
"relatedSysIds": [
  "8b64e971b320001028f6eae2c6a8dc47"
],
"relatedItems": [
  {
    "className": "cmdb_ci_spkg",
    "operation": "NO_CHANGE",
    "sysId": "c764e971b320001028f6eae2c6a8dc44",
    "relatedSysIds": [
      "8b64e971b320001028f6eae2c6a8dc47"
    ],
    "relatedItems": [
      {
        "className": "cmdb_software_instance",
        "sysId": "8b64e971b320001028f6eae2c6a8dc47",
        "markers": [],
        "inputIndices": [0]
      }
    ]
  }
]
"mainIndex": 1,
"subIndex": 0
]
],
"additionalRelatedItems": [],
"identifierEntrySysId": "a52a87c03746220006b216a543990e8c",
"identificationAttempts": [
{
"attributes": [
"key"
],
"identifierName": "Software",
"attemptResult": "MATCHED",
"searchOnTable": "cmdb_ci_spkg",
"hybridEntryCiAttributes": []
}],
"errorCount": 0,
"inputIndices": [1]
],
[
"className": "cmdb_ci_app_server_tomcat",
"operation": "INSERT",
"sysId": "6f29f3d2b3a0001028f6eae2c6a8dcc6",
"identifierEntrySysId": "Unknown",
"identificationAttempts": [
{
"attributes": [
"install_directory",
"running_process_key_parameters",
"sys_class_name"
],
"identifierName": "Tomcat",
"attemptResult": "NO_MATCH",
"searchOnTable": "cmdb_ci_app_server_tomcat",
"hybridEntryCiAttributes": []
},
{
"attributes": [
"cl_port",
"sys_class_name"
],
IdentificationEngine - identifyCI(String jsonString)

Determines the operation (insert/update) to perform with the specified payload without committing the operation in the database.

This works just like createOrUpdateCI(), but does not commit the result.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| jsonString | String   | A JSON formatted string of configuration items to be added or updated. Each input string is in the format 'items: [{}, relations: [{}]}, where each item within the items and relations lists contains name-value pairs. The possible name-value pairs within the items list are:  
  - className - Sys_class_name of the CI to be created or updated.  
  - values: {} - Field information for the CI as name-value pairs, where the name is the field name.  
  - lookup: [{],} - List of records with each item having name-value pairs like the items list.  
  The possible name-value pairs within the relations list are:  
  - parent - Index of the parent item in the dependency relation  
  - child - Index of the child item in the dependency relation  
  - type - Relationship type. This is one of the name field values from the cmdb_rel_type table. |

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String | A JSON formatted string that is a list of results. Each result string is in the format 'items: [{}, relations: [{}]}, where each item within the items and relations lists contains name-value pairs. The possible name-value pairs within the items list are:  
  - className - Sys_class_name for the CI that was updated or created.  
  - operation - Operation to perform, which is one of the following:  
    ◦ DELETE  
    ◦ INSERT  
    ◦ NO_CHANGE  
    ◦ UPDATE  
    ◦ UPDATE_WITH_DOWNGRADE |
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>◦ UPDATE_WITH_SWITCH</td>
<td></td>
</tr>
<tr>
<td>◦ UPDATE_WITH_UPGRADE</td>
<td></td>
</tr>
</tbody>
</table>

- `sysId` - Sys_id of the CI that was updated or created.
- `relatedSysIds` - List of sys_id values of CIs used during lookup based identification.
- `identifierEntrySysId` - Sys_id of identifier entry used during matching.
- `errors` - List of errors in the format of (error, message string)
- `duplicateIndices` - List of indexes of items that are duplicates of the current item.
- `identificationAttempts` - List of attempts in the format of (attributes, identifierName, attemptResult, searchOnTable) where
  - `attributes` - Attributes of identifier entry used during identification
  - `identifierName` - CI identifier to which this identifier entry belongs
  - `attemptResult` - One of SKIPPED, NO_MATCH, MATCHED, MULTI_MATCH
  - `searchOnTable` - Table searched during the identification process.

The possible name-value pairs within the relations list are:
- `className` - Relationship CI's class name and is always cmdb_rel_ci
- `operation` - Type of operation: INSERT, UPDATE, NO_CHANGE
- `sysId` - Sys_id of the relationship CI inserted or updated

The following shows how to reclassify a configuration item.

```javascript
var payload = {
  "items": [
    {
      "className": "cmdb_ci_win_server",
      "values": {
        "short_description": "Linux server description",
        "name": "Linux Server 1"
      }
    }
  ]
}```
Output:

```json
{
   "items": [
      {
         "className": "cmdb_ci_linux_server",
         "operation": "NO_CHANGE",
         "sysId": "440577800f321010150efc91ff767e94",
         "identifierEntrySysId": "556eb250c3400200d8d4bea192d3ae92",
         "identificationAttempts": [
            {
               "attributes": [],
               "info": "sys_object_source SKIPPED",
               "identifierName": ",",
               "attemptResult": "SKIPPED",
               "hybridEntryCiAttributes": []
            },
            {
               "attributes": [
                  "serial_number",
                  "serial_number_type"
               ],
               "identifierName": "Hardware Rule",
               "attemptResult": "SKIPPED",
               "searchOnTable": "cmdb_serial_number",
               "hybridEntryCiAttributes": []
            },
            {
               "attributes": [
                  "serial_number"
               ],
               "identifierName": "Hardware Rule",
               "attemptResult": "SKIPPED",
               "searchOnTable": "cmdb_ci_hardware",
               "hybridEntryCiAttributes": []
            }
         ]
      }
   ]
}
```
IdentificationEngine - identifyCIEnhanced(String source, String input, Object options)

Determines the Configuration Management Database (CMDB) operation (insert/update) to perform with the specified payload (request body), without committing the operations in the database.

Use this method to simulate submission of a payload.

This method is similar to the IdentificationEngine - identifyCI(String jsonString) method, however it also supports the following functionality:
- Partial payloads
  - In case of an item having a warning or error, indicates if an item operation is `INSERT_AS_PARTIAL` or `INSERT_INCOMPLETE`.
  - Returns the sys_ids of partial payloads that were merged with existing partial payloads.
- Supports payload deduplicate feature.
- Generates a summary.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>String</td>
<td>Required. A JSON formatted string of configuration items to add or update.</td>
</tr>
<tr>
<td>input.items</td>
<td>Array</td>
<td>Array of objects defining the items to add or update.</td>
</tr>
<tr>
<td>input.items.className</td>
<td>String</td>
<td>Required. Class/table name, sys_class_name, of the items added or updated.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>the configuration item (CI) to create or update</td>
<td>String</td>
<td>This value can be any CMDB class/table, such as <code>cmdb_ci_linux_server</code> or <code>cmdb_ci_win_server</code>.</td>
</tr>
<tr>
<td>input.items.display_values</td>
<td>Object</td>
<td>Reference fields to create or update for this related item as name-value pairs, where the name is the field name and the value is the referenced display value. If you want to use the sys_id instead of the display value for reference fields, pass the information in the <code>input.items.lookup.values</code> parameter instead of in this parameter. Reference field names depend on the fields selected, such as:</td>
</tr>
<tr>
<td>input.items.internal_id</td>
<td>String</td>
<td>Unique item identifier for the associated payload. This can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td>input.items.lookup</td>
<td>Array</td>
<td>Identifies the top-level item containing the lookup (lookup-based identification).</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>input.items.lookup.className</code></td>
<td>String</td>
<td>Required. Class/table name, sys_class_name, of the configuration item (CI) to create or update. This value can be any CMDB class/table, such as <code>cmdb_serial_number</code> or <code>cmdb_ci</code>.</td>
</tr>
<tr>
<td><code>input.items.lookup.internal_id</code></td>
<td>String</td>
<td>Unique lookup item identifier for the associated payload. This can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td><code>input.items.lookup.sys_object_source_info</code></td>
<td>Object</td>
<td>Defines a unique CI identifier for a specific data source. Different sources may have different name-value pairs for the same CI.</td>
</tr>
</tbody>
</table>

records are used to identify the configuration item based on a lookup table that has a reference back to cmdb_ci.

For example:

```
"lookup": [
  {
    "className": "String",
    "internal_id": "String",
    "sys_object_source_info": {
      (Object)
      "sys_object_source_info": {
        "source_feed": "String"
      }
    }
  }
]
```
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.lookup.sys_object_source_info.source_feed</td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide the name of the feed sending this item.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The data source generates this feed name. It can be any string that uniquely identifies the source.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_name</td>
<td>String</td>
<td>Data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the CI.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The data source generates this field. It can be any string that uniquely identifies the source information.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_native_key</td>
<td>String</td>
<td>Unique key-id for the item from the source.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The data source generates this key. It can be any string that uniquely identifies the source information.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date/time that the item was scanned.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>input.items.lookup.values</td>
<td>Object</td>
<td>Field information as name-value pairs, where the name is the field name. When updating reference fields, the value must be the referenced sys_id.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Field names and types depend on the fields selected by the user, such as:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>{ *values*: { *serial_number*: *String, *serial_number_type*: *String, *valid*: *String, *ip_address*: *String, *mac_address*: *String} }</td>
</tr>
<tr>
<td>input.items.related</td>
<td>Array</td>
<td>Reference to the top-level item that contains the related list. Rules in the Related Entry (cmdb_related_entry) define what type of records can be in this array. These records are used to add items based on a related table that has a reference to the CI that is being identified. The related table may or may not extend cmdb_ci. These records are not used to identify the CI item.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>{ *related*: { *class*: *String, *internal_id*: *String, *sys_object_source_info*: {Object}, *values*: {Object} }</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.related.className</td>
<td>String</td>
<td>Required. Class/Table name, sys_class_name, of the configuration item (CI) to create or update. This value can be any CMDB class/table, such as cmdb_software_instance or cmdb_key.</td>
</tr>
<tr>
<td>input.items.related.internal_id</td>
<td>String</td>
<td>Unique identifier for this related item in this payload. Can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info</td>
<td>Object</td>
<td>Object that makes up a unique CI identifier for a specified data source. Different data sources may have different values for the same object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>&quot;sys_object_source_info&quot;:</strong> {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;sys_object_source_info&quot;: <strong>String</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;source_feed&quot;: <strong>String</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;source_name&quot;: <strong>String</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;source_native_key&quot;: <strong>String</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;source_recency_timestamp&quot;: <strong>String</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info.source_feed</td>
<td>String</td>
<td>If the source has multiple feeds, use this field to provide the name of the feed sending this item. The data source generates this feed name.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.related.sys_object_source_info.source_name</td>
<td>String</td>
<td>Identifies the data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info.source_native_key</td>
<td>String</td>
<td>Unique key/ID from the source for the related item. The data source generates this key. It can be any string that is unique to the item.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date and time that the item was scanned. Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>input.items.related.values</td>
<td>Object</td>
<td>Fields to create or update for this related item as name/value pairs, where the name is the field name. For reference fields, the value must be the referenced sys_id. If you want to use the display value instead of the sys_id for reference fields, pass this information in a display_values object instead of in the values object. Field names and types depend on the fields selected, such as:</td>
</tr>
</tbody>
</table>
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>input.items.settings</code></td>
<td>Object</td>
<td>Parameters that define the types of updates that are permitted.</td>
</tr>
</tbody>
</table>
| `input.items.settings.skipReclassificationRestrictionRules` | Boolean    | Flag that indicates whether the IRE should not run the Reclassification Restriction rule that matches the class for the payload item. Valid values:  
  - true: Skip running the rule.  
  - false: Run the rule.  
  Default: false. |
| `input.items.settings.updateWithoutDowngrade` | Boolean    | Flag that indicates whether update and downgrade are both permitted for this item. Valid values:  
  - true: Permitted  
  - false: Not permitted  
  Default: true. |
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| input.items.settings.updateWithoutSwitch               | Boolean | Flag that indicates whether update and class switching are both permitted for this item. Valid values:  
|                                                       |         | • true: Update the item but class switching is not permitted.  
|                                                       |         | • false: Both item update and class switching are permitted.  
|                                                       |         | Default: false                                                                                   |
| input.items.settings.updateWithoutUpgrade               | Boolean | Flag that indicates whether update and upgrade are both permitted for this item. Valid values:  
|                                                       |         | • true: Update the item but upgrade is not permitted.  
|                                                       |         | • false: Both item update and upgrade are permitted.  
<p>|                                                       |         | Default: false                                                                                   |
| input.items.sys_object_source_info                      | Object  | Unique CI identifier for a specific source                                                     |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.sys_object_source_info.source_feed</td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide the name of the feed sending this item. The data source generates this feed name. It can be any string that uniquely identifies this feed.</td>
</tr>
<tr>
<td>input.items.sys_object_source_info.source_name</td>
<td>String</td>
<td>Data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td>input.items.sys_object_source_info.source_native_key</td>
<td>String</td>
<td>Unique key/id for the item from the source. The data source generates this key. It can be any string that is unique to the item.</td>
</tr>
<tr>
<td>input.items.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date and time that the item was scanned. Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>input.items.values</td>
<td>Object</td>
<td>Fields to create or update for this related item as name/value pairs, where the name is the field name. For a reference field, the value must be the reference name.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>input.referenceltems</td>
<td>Array</td>
<td>Array of objects that define references between items in the input payload.</td>
</tr>
<tr>
<td>input.referenceltems.referenced</td>
<td>String</td>
<td>The <code>internal_id</code> defined for the item being referenced by another item.</td>
</tr>
<tr>
<td>input.referenceltems.referencedBy</td>
<td>String</td>
<td>The <code>internal_id</code> defined for the item that references another item.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.referenceItems.referenceField</td>
<td>String</td>
<td>Name of the field in the class/table for the referencedBy item.</td>
</tr>
<tr>
<td>input.relations</td>
<td>Array</td>
<td>Array of objects that specify relationships between items in the input payload. An object in this array can use either of the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The object can define a relationship between two top-level items using parent and child name-value pairs, with values representing item indexes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The object can define a relationship between any two items, including top-level, related, or lookup items, using parent_id and child_id key-value pairs, with values representing internal_id values defined for those items.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>input.relations.child</code></td>
<td>Number</td>
<td>Integer index of the CI object that represents the child in the relationship array (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td><code>input.relations.child_id</code></td>
<td>String</td>
<td>The internal_id of the child item in the relationship (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td><code>input.relations.parent</code></td>
<td>Number</td>
<td>Integer index of the parent item in the relationship (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td><code>input.relations.parent_id</code></td>
<td>String</td>
<td>The internal_id of the parent item in the relationship (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td><code>input.relations.sys_rel_source_info</code></td>
<td>Object</td>
<td>Discovery source information for the relationship. For non-dependency relationships, this information is saved in the Relationship Sources [sys_rel_source] table (not persisted for identifyCIEnhanced() or identifyCI() methods).</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>input.relations.sys_rel_source_info.source_name</strong></td>
<td>String</td>
<td>Discovery source name. Default: Discovery source passed in API method parameter.</td>
</tr>
<tr>
<td><strong>input.relations.sys_rel_source_info.source_feed</strong></td>
<td>String</td>
<td>Any string that is a sub-discovery/scan within the discovery source. Default: 'UNKNOWN' is stored in the source_feed column when creating a record in sys_rel_source table.</td>
</tr>
<tr>
<td><strong>input.relations.type</strong></td>
<td>String</td>
<td>Type of relationship that exists between parent and child items. Must be a name field value from the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td><strong>options</strong></td>
<td>Object</td>
<td>Optional, must be passed to enable or disable features.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options.deduplicate_payloads</td>
<td>Boolean</td>
<td>Flag that indicates whether duplicate items are merged or considered errors.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Merge duplicate items.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Consider duplicate items as errors.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: true</td>
</tr>
</tbody>
</table>

| options.generate_summary                  | Boolean  | Flag that indicates whether the returned results contain summary information. |
|                                           |          | Valid values:                                                               |
|                                           |          | • true: Include summary information.                                        |
|                                           |          | • false: Do not include summary information.                                |
|                                           |          | Default: false                                                              |
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>&lt;String&gt;.summary</strong></td>
<td></td>
<td>in the return results table. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Include summary information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not include summary information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td><strong>options.partial_commits</strong></td>
<td>Boolean</td>
<td>Flag that indicates whether partial commit support is enabled. For additional information on partial commits, see Enhanced IRE features. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Partial commit enabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Partial commit disabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td><strong>options.partial_payloads</strong></td>
<td>Boolean</td>
<td>Flag that indicates whether partial payload support is enabled. For additional information on partial payloads, see Enhanced IRE features and Create an IRE data source rule. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Partial payload enabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Partial payload disabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>&lt;String&gt;</strong></td>
<td>JSON formatted string that is a list of results for the configuration items in the input string. Data type: String</td>
</tr>
</tbody>
</table>

```json
{
    "additionalCommittedItems": [Array],
    "additionalCommittedRelations": [Array],
    "hasError": "Boolean",
    "hasWarning": "Boolean",
    "items": [Array],
    "relations": [Array],
    "summary": {Object}
}
```

**<String>**.additionalCommittedItems

List of CIs that were committed during the IRE processing of the current payload, but were not present in the current input payload.

Data type: Array

```json
"additionalCommittedItems": [
    {
        "className": "String",
        "errorCount": Number,
        "operation": "String",
        "identificationAttempts": [Array],
        "inputIndices": [Array],
        "markers": [Array],
    }
]
```
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;mergedPayloads&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;sysId&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
</tbody>
</table>

### `<String>.additionalCommittedRelations`

Array of objects that describe a dependent relationship CI that was not included in the request body relations list to insert or update.

Data type: Array

```
"additionalCommittedRelations": [|
|   {|
|     "className": "String",
|     "inputIndices": [Array],
|     "markers": [Array],
|     "mergedPayloadIds": [Array],
|     "operation": "String" |
|   } |
| ]
```

### `<String>.hasError`

Flag that indicates whether any item or relation has errors.

Data type: Boolean

### `<String>.hasWarning`

Flag that indicates whether any item or relation has warnings.

Data type: Boolean

### `<String>.items`

List of objects that describe the created or updated CIs.

Data type: Array

```
"items": [|
|   {|
|     "additionalRelatedItems": [Array],
|     "className": "String",
|     "duplicateIndices": [Array],
|     "errorCount": Number,
|     "errors": [Array],
|     "identificationAttempts": [Array],
|     "identifierEntrySysId": "String",
|     "info": [Array],
|     "inputIndices": [Array],
|   } |
| ]
```
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;maskedAttributes&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;operation&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;relatedSysIds&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;sysId&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

`<String>.items.additionalRelatedItems` objects that provide information about additional lookup and related items that were processed but not provided as part of the input payload. These items are from partial payloads.

Data type: Array

- "additionalRelatedItems": [ |
- "className": "String", |
- "inputIndices": [Array], |
- "mergedPayloadIds": [Array], |
- "operation": "String", |
- "sysId": "String" |
- ]

- `<String>.items.additionalRelatedItems.className`
  Class/table name (sys_class_name) of the CI that was created or updated.
  Data type: String

- `<String>.items.additionalRelatedItems.inputIndices`
  Index values for CIs from the request body `items` array that correspond to this related item.
  Data type: Array of Numbers

- `<String>.items.additionalRelatedItems.mergedPayloadIds`
  List of sys_ids of the partial payloads that were merged into the related item. Located in the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.
  Data type: Array

- `<String>.items.additionalRelatedItems.operation`
  Type of operation.
  Possible values:
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|      | • INSERT: New CI was inserted into the database.  
         • NO_CHANGE: No CI changes were made.  
         • UPDATE: Existing CI was updated.  
         Data type: String |
|<String>.items.additionalRelatedItems.sysId | Sys_id of the CI that was updated or created.  
   Data type: String |
|<String>.items.className | Class/table name (sys_class_name) of the CI that was created or updated.  
   Data type: String |
|<String>.items.duplicateIndices | List of indexes of CIs that are duplicates of the current item.  
   Data type: Array |
|<String>.items.errorCount | Number of errors encountered while processing the item.  
   Data type: Number |
|<String>.items.errors | Array of objects in which each object describes an error encountered while processing this CI.  
   Data type: Array |
|              | "errors": [ |
|              |   { |
|              |     "error": "String", |
|              |     "message": "String" |
|              |   } |
|              | ] |
|<String>.items.errors.type | Type of error encountered while processing the CI.  
   Data type: String |
|<String>.items.errors.message | Error message encountered while processing the CI.  
   Data type: String |
|<String>.items.identificationAttempts | List of attempts that were made to identify the CIs.  
   Data type: Array |
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;identificationAttempts&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;attemptResult&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;attributes&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;hybridEntryCiAttributes&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;identifierName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;searchOnTable&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

#### <String>.items.identificationAttempts.attemptResult

Possible values:

- **MATCHED**: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.
- **MULTI_MATCH**: Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.
- **NO_MATCH**: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.
- **SKIPPED**: Identification not attempted. The attributes required for this identifier rule table search were not provided, so the rule was not applied.

Data type: String

#### <String>.items.identificationAttempts.attributes

List of CI identifier entry attributes that were used during the identification process.

Data type: Array

Attribute names and types depend on the request body data and the identifier in use, such as:

```
"attributes": [ |
  "serial_number": "String", |
  "serial_number_type": "String", |
] ```
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.identifierName</td>
<td>Identifier rule used for this CI identification attempt. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.searchOnTable</td>
<td>Name of the table searched during the identification process. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.identifierEntrySysId</td>
<td>Sys_id for the identifier rule used to identify the CI. Located in the Identifier Entry [cmdb_identifier_entry] table. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.info</td>
<td>List of objects that contains additional information about the processing of the item. Data type: Array</td>
</tr>
<tr>
<td>&quot;info&quot;: [ ]</td>
<td></td>
</tr>
<tr>
<td>&quot;info&quot;: [</td>
<td></td>
</tr>
<tr>
<td>&quot;code&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;message&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;ruleSysId&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>]</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.info.code</td>
<td>Reclassification type that was skipped. Possible values:</td>
</tr>
<tr>
<td>• SKIPPED_CLASS_SWITCH</td>
<td></td>
</tr>
<tr>
<td>• SKIPPED_CLASS_DOWNGRADE</td>
<td></td>
</tr>
<tr>
<td>• SKIPPED_CLASS_UPGRADE</td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.info.message</td>
<td>Message that provides additional insights into the reason for skipping the reclassification. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.info.ruleSysId</td>
<td>Sys_id of the reclassification restriction rule that was matched. Applicable only when IRE skips reclassification due to reclassification restriction rule. This value is empty if the reclassification is skipped due to a payload or global flag. Data type: String</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.inputIndices</td>
<td>Returns the indices of the corresponding input CI. For top-level items, it is a list of integers. For related or lookup CIs, it is list of JSON objects. Data type: Array of Numbers</td>
</tr>
<tr>
<td>&lt;String&gt;.items.maskedAttributes</td>
<td>List of attributes whose update by a non-authoritative data source was skipped as defined by the Reconciliation Rules. Data type: Array</td>
</tr>
</tbody>
</table>
| <String>.items.operation | Operation that took place. Possible values:  
  - INSERT: New CI was inserted into the database.  
  - INSERT_AS_INCOMPLETE: Item was saved in cmdb_ire_incomplete_payloads table.  
  - INSERT_AS_PARTIAL: Item was saved in cmdb_ire_partial_payloads table.  
  - NO_CHANGE: No CI changes were made.  
  - UPDATE: Existing CI was updated.  
  - UPDATE_WITH_DOWNGRADE: CI was updated and the class changed to a more generic class (ancestor class).  
  - UPDATE_WITH_SWITCH: CI was updated and the class changed to a class that is not ancestor or descendent.  
  - UPDATE_WITH_UPGRADE: CI was updated and the class changed to a more specialized class (descendent class). Data type: String |
| <String>.items.relatedItems | List of JSON objects that provide information about the processed related items. Data type: Array |

*relatedItems*:

```javascript
[]
```
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;className&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;errorCount&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;inputIndices&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;mergePayloadIds&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;operation&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sysId&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;warningCount&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;warnings&quot;: [Array]</td>
</tr>
</tbody>
</table>

<String>.items.relatedItems.className
Data type: String

<String>.items.relatedItems.errorCount
Data type: Number

<String>.items.relatedItems.errors
List of errors that occurred during processing of the related item.
Data type: Array

"errors": [
|
| "error": "String", |
| "message": "String"
|

<String>.items.relatedItems.errors.error
Type of error encountered while processing the related item.
Data type: String

<String>.items.relatedItems.errors.message
The error message encountered while processing the related item.
Data type: String

<String>.items.relatedItems.inputIndices
Index of the corresponding input item. For top-level items, it is a list of integers. For related or lookup items, it is list of JSON objects.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Data type: Array of Numbers or Array of Objects | "inputIndices": [  
| |  
| |   "mainIndex": Number,  
| |   "subIndex": Number  
<p>| | ] |
| &lt;String&gt;.items.relatedItems.inputIndices.mainIndex | Index value from the request body items array that corresponds to the CI parent of the related item. |
| Data type: Number | |
| &lt;String&gt;.items.relatedItems.inputIndices.subIndex | Index value from the request body items.lookup array that corresponds to the related item. |
| Data type: Number | |
| &lt;String&gt;.items.relatedItems.mergedPayloadIds | List of sys_ids of the partial payloads that were merged into the CI during processing. |
| Data type: Array | |
| &lt;String&gt;.items.relatedItems.operation | Operation that took place. |
| Possible values: | |
| INSERT: New related CI was inserted into the database. | |
| INSERT_AS_INCOMPLETE: Item was saved in cmdb Ire_incomplete_payloads table. | |
| INSERT_AS_PARTIAL: Item was saved in cmdb Ire_partial_payloads table. | |
| NO_CHANGE: No related CI changes were made. | |
| UPDATE: Existing related CI was updated. | |
| UPDATE_WITH_DOWNGRADE: Related CI was updated and the class changed to a more generic class (ancestor class). | |</p>
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• UPDATE_WITH_SWITCH: Related CI was updated and the class changed to a class that is not ancestor or descendent.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE_WITH_UPGRADE: Related CI was updated and the class changed to a more specialized class (descendent class).</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

<String>.items.relatedItems.sysId
Data type: String

<String>.items.relatedItems.warningCount
Data type: Number

<String>.items.relatedItems.warnings
Array of objects that describes a warning encountered while processing the related items.
Data type: Array

```
"warnings": [
{
  "error": "String",
  "message": "String"
}
]
```

<String>.items.relatedSysIds
List of the sys_id values of the CIs used during lookup-based identification of related items.
Data type: String

<String>.items.sysId
Sys_id of the CI that was updated or created.
Data type: String

<String>.relations
List of JSON objects that provides information about the processed relations.
Data type: Array

```
"relations": [
{
```
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **className**: "String",
- **errorCount**: Number,
- **errors**: [Array],
- **inputIndices**: [Array],
- **operation**: "String",
- **sysId**: "String"

### <String>.relations.className

- Synonym: class_name of this dependent relationship CI.
- Only supported value:
  - cmdb_rel_ci: CI Relationship table.
- Data type: String

### <String>.relations.errorCount

- Description: Number of errors encountered when processing the dependent relationship CI.
- Data type: Number

### <String>.relations.errors

- Description: Array of objects that describe errors that were encountered while processing this dependent relationship CI.
- Data type: Array

```json
"errors": [
  {
    "error": "String",
    "message": "String"
  }
]
```

### <String>.relations.inputIndices

- Description: Indexes for the dependent relationship CI objects in the request body relations array that correspond to this dependent relationship CI.
- Data type: Array

### <String>.relations.operation

- Description: Type of operation performed.
- Possible values:
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• INSERT: The dependent relationship CI was inserted into the target table as a new record.</td>
</tr>
<tr>
<td></td>
<td>• INSERT_AS_INCOMPLETE: The dependent relationship CI had errors and was inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.</td>
</tr>
<tr>
<td></td>
<td>• INSERT_AS_PARTIAL: The dependent relationship CI had errors and was inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.</td>
</tr>
<tr>
<td></td>
<td>• NO_CHANGE: No operation was performed.</td>
</tr>
<tr>
<td></td>
<td>• UPDATE: An existing dependent relationship CI in the target table was updated.</td>
</tr>
</tbody>
</table>

Data type: String

- `<String>.relations.sysId` – Sys_id of the dependent relationship CI.
  - Data type: String

- `<String>.summary` – List of JSON properties that provide statistics on how many items were inserted, updated, and such, per class.
  - Data type: Array

- `<String>.summary.<class_name>` – Statistics for a specific class.
  - Data type: Object

```json
<class_name>: {
  "additionalInsertedItemCount": Number,
  "errorCount": Number,
  "incompleteItemCount": Number,
  "insertedItemCount": Number,
  "partialItemCount": Number,
  "skippedItemCount": Number,
  "unchangedItemCount": Number,
  "updatedItemCount": Number,
  "warningCount": Number
}
```

- `<String>.summary.<class_name>.additionalInsertedItemCount` – Number of items inserted due to processing of partial payloads.

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.summary.&lt;class_name&gt;.errorCount</td>
<td>Number of errors encountered when processing items. Data type: Number</td>
</tr>
<tr>
<td>&lt;String&gt;.summary.&lt;class_name&gt;.incompleteItemCount</td>
<td>Number of items inserted in the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table. Data type: Number</td>
</tr>
<tr>
<td>&lt;String&gt;.summary.&lt;class_name&gt;.insertedItemCount</td>
<td>Number of items created. Data type: Number</td>
</tr>
<tr>
<td>&lt;String&gt;.summary.&lt;class_name&gt;.partialItemCount</td>
<td>Number of items saved in the Partial Payload table [cmdb_ire_partial_payloads]. Data type: Number</td>
</tr>
<tr>
<td>&lt;String&gt;.summary.&lt;class_name&gt;.skippedItemCount</td>
<td>Number of items that were not modified. Data type: Number</td>
</tr>
<tr>
<td>&lt;String&gt;.summary.&lt;class_name&gt;.updatedItemCount</td>
<td>Number of items updated. Data type: Number</td>
</tr>
<tr>
<td>&lt;String&gt;.summary.&lt;class_name&gt;.warningCount</td>
<td>Number of items that generated a warning when processed. Data type: Number</td>
</tr>
</tbody>
</table>

```javascript
var payload = {
  "items": [
    {
      "className": "cmdb_ci_win_server",
      "values": {
        "chassis_type": "Desktop",
        "os": "Windows 2012 R2 Datacenter",
        "name": "Windows2012Server1",
        "serial_number": "0000-0011-1690-8730-8636-5722-52",
        "cpu_count": "1"
      }
    }
  ]
};
```
"lookup": [
{
"values": {
"valid": "true",
"serial_number": "0000-0011-1690-8730-8636-5722-52",
"serial_number_type": "bios"
},
"className": "cmdb_serial_number"
},
{
"values": {
"valid": "true",
"serial_number": "3311-9736-4988-9744-1749-4183-41",
"serial_number_type": "chassis"
},
"className": "cmdb_serial_number"
}
],
"internal_id": "16777219",
"sys_object_source_info": {
"source_feed": "SN Discovery Feed 1",
"source_name": "ServiceNow",
"source_native_key": "16777219",
"source_recency_timestamp": "2019-10-18 08:31:23"
}
],
{
"className": "cmdb_ci_spkg",
"values": {
"name": "Windows 2012 R2 Datacenter",
"key": "Microsoft Windows Server 2012 R2 Datacenter:::_NULL"
}
},
"related": [
{
"internal_id": "Microsoft Windows Server 2012 R2 Datacenter:::_NULL|16777219",
"values": {
"name": "Windows 2012 R2 Datacenter-SAMLABVM52"
},
"className": "cmdb_software_instance",
"sys_object_source_info": {
"source_feed": "SN Discovery Feed 1",
"source_name": "ServiceNow",
"source_native_key": "Microsoft Windows Server 2012 R2 Datacenter:::_NULL|16777219"
}
}
"internal_id": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL|16777229",
  "values": {
    "name": "Windows 2012 R2 Datacenter-SAMLABVM52"
  },
"className": "cmdb_software_instance",
"sys_object_source_info": {
  "source_feed": "SN Discovery Feed 1",
  "source_name": "ServiceNow",
  "source_native_key": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL|16777229"
}
],
"settings": {
  "skipReclassificationRestrictionRules": "false",
  "updateWithoutDowngrade": "true",
  "updateWithoutUpgrade": "true",
  "updateWithoutSwitch": "true"
},
"className": "cmdb_ci_app_server_tomcat",
"values": {
  "running_process_key_parameters": "/opt/OV/nonOV/tomcat/b/temp org.apache.catalina.startup.Bootstrap start",
  "install_directory": "/opt/OV/nonOV/tomcat/b",
  "name": "Tomcat@hpom9:3443",
  "server_port": "8006",
  "sys_class_name": "cmdb_ci_app_server_tomcat"
},
"internal_id": "tomcat_id"
],
"relations": [
  {
    "parent_id": "tomcat_id",
    "child_id": "16777219",
    "type": "Runs on::Runs"
  }
],
"referenceItems": [
  {
    "referenceField": "installed_on",
    "referenced": "16777219",
    "internal_id": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL|16777229"
  }
],
"className": "cmdb_software_instance",
"sys_object_source_info": {
  "source_feed": "SN Discovery Feed 1",
  "source_name": "ServiceNow",
  "source_native_key": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL|16777229"
}
]
var input = JSON.stringify(payload);
var output = sn_cmdb.IdentificationEngine.identifyCIEnhanced('ServiceNow', input, {});
gs.info(JSON.stringify(JSON.parse(output), null, 2));

Output:

{
  "items": [
  {
    "className": "cmdb_ci_win_server",
    "operation": "INSERT",
    "relatedSysIds": [null, null],
    "relatedItems": [
      {
        "errors": [],
        "operation": "INSERT",
        "className": "cmdb_serial_number",
        "errorCount": 0,
        "markers": [],
        "inputIndices": [null],
        "mergedPayloadIds": [],
        "warningCount": 0
      },
      {
        "errors": [],
        "operation": "INSERT",
        "className": "cmdb_serial_number",
        "errorCount": 0,
        "markers": [],
        "inputIndices": [null]
      }
    ]
  }
]
"subIndex": 1
},
"mergedPayloadIds": [],
"warningCount": 0
],
"identificationAttempts": [
{
"attributes": [],
"info": "sys_object_source NO_MATCH",
"identifierName": "",
"attemptResult": "NO_MATCH",
"hybridEntryCiAttributes": []
},
{
"attributes": [
  "serial_number",
  "serial_number_type"
],
"identifierName": "Hardware Rule",
"attemptResult": "NO_MATCH",
"searchOnTable": "cmdb_serial_number",
"hybridEntryCiAttributes": []
},
{
"attributes": [
  "serial_number"
],
"identifierName": "Hardware Rule",
"attemptResult": "NO_MATCH",
"searchOnTable": "cmdb_ci_hardware",
"hybridEntryCiAttributes": []
},
{
"attributes": [
  "name"
],
"identifierName": "Hardware Rule",
"attemptResult": "NO_MATCH",
"searchOnTable": "cmdb_ci_hardware",
"hybridEntryCiAttributes": []
}],
"subIndex": 1
}
"attributes": [ 
    "mac_address",
    "name"
],
"identifierName": "Hardware Rule",
"attemptResult": "SKIPPED",
"searchOnTable": "cmdb_ci_network_adapter",
"hybridEntryCiAttributes": []
},
"errorCount": 0,
"markers": [],
"inputIndices": [ 0 ],
"mergedPayloadIds": [],
"warningCount": 0},
{ 
"className": "cmdb_ci_spkg",
"operation": "INSERT",
"relatedSysIds": [ null ],
"relatedItems": [ 
    { 
        "errors": [],
        "operation": "INSERT",
        "className": "cmdb_software_instance",
        "errorCount": 0,
        "markers": [],
        "inputIndices": [ 
            { 
                "mainIndex": 1,
                "subIndex": 0
            }
        ],
        "mergedPayloadIds": [],
        "warningCount": 0
    }
]
"error": "MISSING_MATCHING_ATTRIBUTES",
"message": "In payload missing minimum set of input values for criterion (matching) attributes from identify rule for table [cmdb_software_instance]. Add these input values in payload item 
'"className": "cmdb_software_instance", 
"values": {}, 
"internal_id": ":f7273ccce010056cd4bb46eb4db5d", 
"sys_object_source_info": { 
"source_feed": "SN Discovery Feed 1", 
"source_name": "ServiceNow", 
"source_native_key": "Microsoft Windows Server 2012 R2 Datacenter:::NULL|16777229" }, 
"settings": {}, 
"sys_ire_info": { 
"ire_received_time": "2020-05-10 17:57:48" 
}'}" 

"operation": "INSERT_AS_PARTIAL",
"className": "cmdb_software_instance",
"errorCount": 0,
"sysId": "Unknown",
"markers": [],
"inputIndices": [
  {
    "mainIndex": 1,
    "subIndex": 1
  }
],
"mergedPayloadIds": [],
"warningCount": 1
],
"identificationAttempts": [
  {
    "attributes": [],
    "info": "sys_object_source SKIPPED",
    "identifierName": ",",
    "attemptResult": "SKIPPED",
    "hybridEntryCiAttributes": []
  },
  {
    "attributes": [
      "key"
    ],
    "identifierName": "Software",
    "attemptResult": "NO_MATCH",
    "searchOnTable": "cmdb_ci_spkg",
    "hybridEntryCiAttributes": []
  }
]
**IdentificationEngine - runIdentificationAudit(GlideRecord now_GR)**

Runs an identification audit against the specified configuration item (CI) to detect duplicates.

If duplicates are found, duplication tasks are created. Only use this method on CI types with independent identification rules.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>CI on which to run the audit to detect duplicates. The CI must have independent identification rules.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to check a record in the Linux Servers [cmdb_ci_linux_server] table for duplicates.

```javascript
var sysId = '<cbdb_ci_sys_id>';  
var gr = new GlideRecord('cmdb_ci_linux_server');  
gr.get(sysId);  
SNC.IdentificationEngineScriptableApi.runIdentificationAudit(gr);
```

**IdentificationEngineScriptableApi - Global**

The IdentificationEngineScriptableApi uses the Identification and Reconciliation framework to minimize the creation of duplicate configuration items (CIs) and to reconcile CI attributes by only accepting information from authorized data sources when updating the Configuration Management Database (CMDB).

**IdentificationEngineScriptableApi - createOrUpdateCI(String source, String input)**

Inserts or updates configuration items and non-Configuration Management Database (CMDB) configuration items (classes not extending from cmdb_ci) in the CMDB based on identification and reconciliation rules. Use this API instead of updating the CMDB directly.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>String</td>
<td>Required. A JSON formatted string of configuration items to add or update.</td>
</tr>
<tr>
<td>input.items</td>
<td>Array</td>
<td>Array of objects that define the items to add or update.</td>
</tr>
<tr>
<td>input.items.className</td>
<td>String</td>
<td>Required. Class/table name, sys_class_name, of the configuration item (CI) to create or update. This value can be any CMDB class/table, such as cmdb_ci_linux_server or cmdb_ci_win_server.</td>
</tr>
<tr>
<td>input.items.display_values</td>
<td>Object</td>
<td>Reference fields to create or update for this related item as name-value pairs,</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.internal_id</td>
<td>String</td>
<td>Unique item identifier for the associated payload. This can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td>input.items.lookup</td>
<td>Array</td>
<td>Identifies the top-level item containing the lookup (lookup-based identification). These records are used to identify the configuration item based on a lookup table that has a reference back to cmdb_ci. For example:</td>
</tr>
</tbody>
</table>

```json
"lookup": {
  "className": "String",
  "model_id": "String",
  "location": "String"
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.lookup.className</td>
<td>String</td>
<td>Required. Class/table name, <code>sys_class_name</code>, of the configuration item (CI) to create or update. This value can be any CMDB class/table, such as <code>cmdb_serial_number</code> or <code>cmdb_ci_network_adapter</code>.</td>
</tr>
<tr>
<td>input.items.lookup.internal_id</td>
<td>String</td>
<td>Unique item identifier for the associated payload. This can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info</td>
<td>Object</td>
<td>Unique CI identifier for a specific source. <strong>sys_object_source_info</strong>: <code>{source_feed: String, source_name: String, source_native_key: String, source_recency_timestamp: String}</code></td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_feed</td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide the name of the feed sending this item. The data source generates this feed name. It can be any value, but must be unique within the payload.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.lookup.sys_object_source_info.source_name</td>
<td>String</td>
<td>Data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the [cmdb_ci] table.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_native_key</td>
<td>String</td>
<td>Unique key/id for the item from the source. The data source generates this key. It can be any string that is unique to the item.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date and time that the item was scanned. Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>input.items.lookup.values</td>
<td>Object</td>
<td>Fields to create or update for this related item as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced sys_id. If you want to use the display value instead of the sys_id for reference fields, pass this information in a display_values object instead of in the values object. Field names and types depend on the fields selected by the user, such as:</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;values&quot;:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;host_name&quot;:</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>&quot;ip_address&quot;:</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>&quot;name&quot;:</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>&quot;os_name&quot;:</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_class_name&quot;:</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>input.items.related</td>
<td>Array</td>
<td>Reference to the top-level item, in the Related Entry [cmdb_related_entry],</td>
</tr>
<tr>
<td></td>
<td></td>
<td>define what type of records can be included. These records are used to add</td>
</tr>
<tr>
<td></td>
<td></td>
<td>items based on a related table that has a reference to the CI being</td>
</tr>
<tr>
<td></td>
<td></td>
<td>identified. The related table may or may not extend cmdb_ci. These records</td>
</tr>
<tr>
<td></td>
<td></td>
<td>are not used to identify the configuration item.</td>
</tr>
<tr>
<td>&quot;related&quot;:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;className&quot;:</td>
<td>String</td>
<td>Required. Class/table name, sys_class_name, of the configuration item (CI)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>to create or update.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.

Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.related.internal_id</td>
<td>String</td>
<td>Unique item identifier for the associated payload. This can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info</td>
<td>Object</td>
<td>Unique CI identifier for a specific source.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info.source_feed</td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide the feed sending this item. The data source generates this feed name. It can be any string that uniquely identifies the source feed.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info.source_name</td>
<td>String</td>
<td>Data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info.source_native_key</td>
<td>String</td>
<td>Unique key/id for the item from the source.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.related.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date and time that the item was scanned. Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>input.items.related.values</td>
<td>Object</td>
<td>Fields to create or update for this related item as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced sys_id. If you want to use the display value instead of the sys_id for reference fields, pass this information in a <code>display_values</code> object instead of a <code>values</code> object. Field names and types depend on the fields selected by the user, such as:</td>
</tr>
<tr>
<td>input.items.settings</td>
<td>Object</td>
<td>Parameters that define the types of updates that are permitted.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.settings.skipReclassificationRestrictionRules</td>
<td>Boolean</td>
<td>Flag that indicates whether IRE should not run the Reclassification Restriction rule that matches the class for the payload item. Valid values:  • true: Skip running the rule.  • false: Run the rule. Default: false</td>
</tr>
<tr>
<td>input.items.settings.updateWithoutDowngrade</td>
<td>Boolean</td>
<td>Flag that indicates whether update and downgrade are both permitted for this item. Valid values:  • true: Update the item but downgrade is not permitted.  • false: Both item update and downgrade are permitted. Default: false</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| `input.items.settings.updateWithoutUpgrade` | Boolean | Flag that indicates whether update and upgrade are both permitted. Valid values:
|                                           |         | • true: Update the item but upgrade is not permitted.                      |
|                                           |         | • false: Both item update and upgrade are permitted.                       |
|                                           |         | Default: false                                                             |
| `input.items.settings.updateWithoutSwitch` | Boolean | Flag that indicates whether the item can be updated and the class switched. Valid values:
|                                           |         | • true: Update the item but class switching is not permitted.              |
|                                           |         | • false: Both item update and class switching are permitted.               |
|                                           |         | Default: false                                                             |
| `input.items.sys_object_source_info`      | Object  | Unique CI identifier for a specific source.                                |
| `input.items.sys_object_source_info.source_feed` | String  | If the source can have multiple feeds, use this field.                     |
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.sys_object_source_info.source_name</td>
<td>String</td>
<td>to provide the name of the feed sending this item. The data source generates this feed name. It can be any string that uniquely identifies the source feed.</td>
</tr>
<tr>
<td>input.items.sys_object_source_info.source_native_key</td>
<td>String</td>
<td>Data source information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td>input.items.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date and time that the item was scanned. Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>input.items.values</td>
<td>Object</td>
<td>Fields to create or update for this related item as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced sys_id. If you want to use the display value instead of the sys_id for reference fields, pass this information in a <code>display_values</code> object instead of in the <code>values</code> object.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Field names and types depend on the fields selected by the user, such as:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;values&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;host_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;ip_address&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;os_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;sys_class_name&quot;: &quot;String&quot; }</td>
</tr>
<tr>
<td>input.referenceltems</td>
<td>Array</td>
<td>Array of objects that define references between items in the input payload.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;referenceItems&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;referenced&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;referencedBy&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;referenceField&quot;: &quot;String&quot; }</td>
</tr>
<tr>
<td>input.referenceltems.referenced</td>
<td>String</td>
<td>The internal_id defined for the item being referenced by another item.</td>
</tr>
<tr>
<td>input.referenceltems.referencedBy</td>
<td>String</td>
<td>The internal_id defined for the item that references another item.</td>
</tr>
<tr>
<td>input.referenceltems.referenceField</td>
<td>String</td>
<td>Name of the reference field in the class/table for the referencedBy item.</td>
</tr>
<tr>
<td>input.relations</td>
<td>Array</td>
<td>Array of objects that specify relationships between items in the input payload.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>An object in this array</td>
<td></td>
<td>can use either of two formats.</td>
</tr>
<tr>
<td>• The object can define</td>
<td></td>
<td>a relationship between two top-level items using <code>parent</code> and <code>child</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>name-value pairs, with values representing item indexes from the payload.</td>
</tr>
</tbody>
</table>
|                           |      | `relations`: `{
|                           |      |   "child": Number,
|                           |      |   "parent": Number,
|                           |      |   "sys_rel_source_info": Object,
|                           |      |   "type": "String"
|                           |      | }`                                                      |
| • The object can define a |      | relationship between any two items, including top-level, related, or lookup |
|                           |      | items, using `parent_id` and `child_id` key-value pairs, with values      |
|                           |      | representing internal_id values defined for those items.                  |
| `relations`: `{
|                           |      |   "child_id": "String",
|                           |      |   "parent_id": "String",
|                           |      |   "sys_rel_source_info": Object,
|                           |      |   "type": "String"
|                           |      | }`
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.relations.child</td>
<td>Number</td>
<td>Integer index of the CI object that is the child in the relationship (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td>input.relations.child_id</td>
<td>String</td>
<td>The internal_id of the child item in the relationship (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td>input.relations.parent</td>
<td>Number</td>
<td>Integer index of the parent item in the relationship (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td>input.relations.parent_id</td>
<td>String</td>
<td>The internal_id of the parent item in the relationship (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td>input.relations.sys_rel_source_info</td>
<td>Object</td>
<td>Discovery source information for the relationship. For non-dependency relationships, this information is saved in the Relationship Sources [sys_rel_source] table (not persisted for identifyCIEnhanced() or identifyCI() methods.).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;sys_rel_source_info&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;source_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;source_feed&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td>input.relations.sys_rel_source_info.source_name</td>
<td>String</td>
<td>Discovery source name.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.relations.sys_rel_source_info.source_feed</td>
<td>String</td>
<td>Any string that is a sub-discovery/scan within the discovery source. Default: 'UNKNOWN' is stored in the source_feed column when creating a record in the sys_rel_source table.</td>
</tr>
<tr>
<td>input.relations.type</td>
<td>String</td>
<td>Type of relationship that exists between the parent and child items. Must be a name field value from the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td>source</td>
<td>String</td>
<td>Identifies the source of the CI information. Must be one of the choice values defined for the discovery_source field of the cmdb_ci table.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;</td>
<td>JSON formatted string that is a list of results for the configuration items in the input string. Each result string is in the format 'items: [{}], relations: [{}]', where each item within items and relations lists contains name-value pairs. Data type: String</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.additionalCommittedItems</code></td>
<td>No values are currently returned.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations</code></td>
<td>No values are currently returned.</td>
</tr>
</tbody>
</table>
| `<String>.items` | List of objects that describe the created or updated CIs.  
Data type: Array  
```
"items": {
    "additionalRelatedItems": [Array],
    "className": "String",
    "duplicateIndices": [Array],
    "errorCount": Number,
    "errors": [Array],
    "identificationAttempts": [Array],
    "identifierEntrySysId": "String",
    "info": [Array],
    "inputIndices": [Array],
    "maskedAttributes": [Array],
    "operation": "String",
    "relatedItems": [Array],
    "relatedSysIds": [Array],
    "sysId": "String"
}
```
| `<String>.items.additionalRelatedItems` | List of JSON objects that provides information about additional lookup and related items that were processed but not provided as part of the input payload. These items are from partial payloads.  
This information is not currently returned.  
Data type: Array |
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;additionalRelatedItems&quot;: [</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;className&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;inputIndices&quot;: [Array],</td>
<td></td>
</tr>
<tr>
<td>&quot;operation&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sysId&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>]</td>
<td></td>
</tr>
</tbody>
</table>

### <String>.items.additionalRelatedItems.className

Class/table name (sys_class_name) of the CI that was created or updated.

Data type: String

### <String>.items.additionalRelatedItems.inputIndices

Index of the corresponding input item. For top-level items, it is a list of integers. For related or lookup items, it is list of JSON objects.

Data type: Array of Numbers or Array of objects

### <String>.items.additionalRelatedItems.inputIndices.mainIndex

Index value from the request body items array that corresponds to the CI parent of the additional related item.

Data type: Number

### <String>.items.additionalRelatedItems.inputIndices.subIndex

Index value from the request body items.lookup array that corresponds to the additional related item.

Data type: Number

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| <String>.items.additionalRelatedItems.operation                     | Type of operation. Possible values:  
|                                                                      | - INSERT: New CI was inserted into the database.  
|                                                                      | - NO_CHANGE: No CI changes were made.  
|                                                                      | - UPDATE: Existing CI was updated.  
|                                                                      | Data type: String |
| <String>.items.additionalRelatedItems.sysId                         | Sys_id of the additional related items. Data type: String |
| <String>.items.className                                             | Class/table name (sys_class_name) of the CI that was created or updated. Data type: String |
| <String>.items.duplicateIndices                                      | List of indexes of CIs that are duplicates of the current item. Data type: Array |
| <String>.items.errorCount                                           | Number of errors. Data type: Number |
| <String>.items.errors                                                | Array of objects in which each object describes an error encountered while processing this CI. Data type: Array |

```json
"errors": [
  
]
```
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.errors.error</code></td>
<td>Type of error encountered while processing the CI. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.errors.message</code></td>
<td>Error message associated with the error. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.identificationAttempts</code></td>
<td>List of attempts that were made to identify the CIs. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"identificationAttempts": [
  {
    "attemptResult": "String",
    "attributes": [Array],
    "hybridEntryCiAttributes": [Array],
    "identifierName": "String",
    "searchOnTable": "String"
  }
]
```

<table>
<thead>
<tr>
<th><code>&lt;String&gt;.items.identificationAttempts.attemptResult</code></th>
<th>Results of the attempt to identify the CI. Possible values:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• MATCHED: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.</td>
<td></td>
</tr>
<tr>
<td>• MULTI_MATCH: Identification failed with an error. Duplicate CIs were found in the identifier rule table when</td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>matching against the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• NO_MATCH: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• SKIPPED: Identification not attempted. The attributes required for this identifier rule table search were not provided, so the rule was not applied.</td>
</tr>
</tbody>
</table>

Data type: String

```json
<String>.items.identificationAttempts.attributes
```

List of CI identifier entry attributes that were used during the identification process.

Data type: Array

Attribute names and types depend on the request body data and the identifier in use, such as:

```json
"attributes": {
  "serial_number": "String",
  "serial_number_type": "String"
}
```

```json
<String>.items.identificationAttempts.hybridEntryCiAttributes
```

No values are currently returned.

`<String>.items.identificationAttempts.identifierName`

Identifier rule used for this CI identification attempt.

Data type: String

`<String>.items.identificationAttempts.searchOnTable`

Name of the table searched during the identification process.
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.identifierEntrySysId</code></td>
<td>Sys_id for the identifier rule used to identify the CI. Located in the Identifier Entry [cmdb_identifier_entry] table.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.info</code></td>
<td>List of objects that contain additional information about the processing of the item. Data type: Array</td>
</tr>
<tr>
<td><code>&quot;info&quot;: [ { &quot;code&quot;: &quot;String&quot;, &quot;message&quot;: &quot;String&quot;, &quot;ruleSysId&quot;: &quot;String&quot; } ]</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.info.code</code></td>
<td>Reclassification type that was skipped. Possible values: • SKIPPED_CLASS_SWITCH • SKIPPED_CLASS_DOWNGRADE • SKIPPED_CLASS_UPGRADE Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.info.message</code></td>
<td>Message that provides additional insights into the reason for skipping the reclassification. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.info.ruleSysId</code></td>
<td>Sys_id of the reclassification restriction rule that was matched. Applicable only when IRE skips reclassification.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.inputIndices</td>
<td>Index values for CIs from the request body <code>items</code> array that correspond to this CI. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.items.maskedAttributes</td>
<td>List of attributes whose update by a non-authoritative data source was skipped as defined by the Reconciliation Rules. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems</td>
<td>List of JSON objects that provides information about processed related items. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"relatedItems": [ 
  
  { 
    "className": "String",
    "errors": [Array],
    "errorCount": Number,
    "inputIndices": [Array],
    "operation": "String",
    "sysId": "String"
  } 
]
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.relatedItems.className</td>
<td>Class/table name (sys_class_name) of the related item. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.errors</td>
<td>List of errors that occurred during processing.</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data type</strong>: Array</td>
<td></td>
</tr>
<tr>
<td>&quot;errors&quot;: [</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;error&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;message&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td><strong>Number of errors detected during processing.</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Data type</strong>: Number</td>
<td></td>
</tr>
<tr>
<td><strong>Indexes of the corresponding related items.</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Data type</strong>: Array or Numbers</td>
<td></td>
</tr>
<tr>
<td>&quot;inputIndices&quot;: [</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;mainIndex&quot;: Number,</td>
<td></td>
</tr>
<tr>
<td>&quot;subIndex&quot;: Number</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td><strong>Integer value from the request body items array that corresponds to the CI parent of the related item.</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Data type</strong>: Number</td>
<td></td>
</tr>
<tr>
<td><strong>Integer value from the request body items.lookup array that corresponds to the related item.</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Data type</strong>: Number</td>
<td></td>
</tr>
<tr>
<td><strong>Type of operation.</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Possible values:</strong></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| • INSERT: New CI was inserted into the database.  
• NO_CHANGE: No CI changes were made.  
• UPDATE: Existing CI was updated. |

Data type: String

| <String>.items.relatedSysIds | List of the sys_id values for related items (table lookup items) from the request body items.lookup array.  
Notable values:  
• null: No sys_id was identified for this related item. |

Data type: Array

| <String>.items.sys_id | Sys_id of the CI that was updated or created. |

Data type: String

| <String>.relations | List of JSON objects that provides information about processed relations. |

Data type: Array

```json
"relations":[
  {
    "className": "String",
    "errorCount": Number,
    "inputIndices": [Array],
    "operation": "String",
    "sysId": "String"
  }
]
```

| <String>.relations.className | Sys_class_name of this dependent relationship CI. |
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only supported value:</td>
<td>• cmdb_rel_ci: CI Relationship table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.relations.errorCount</td>
<td>Number of errors. Data type: Number</td>
</tr>
<tr>
<td>&lt;String&gt;.relations.inputIndices</td>
<td>Indexes of the corresponding input relations. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.relations.operation</td>
<td>Type of operation performed. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• INSERT</td>
</tr>
<tr>
<td></td>
<td>• UPDATE</td>
</tr>
<tr>
<td></td>
<td>• NO_CHANGE</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.relations.sysId</td>
<td>Sys_id of the dependent relationship CI. Data type: String</td>
</tr>
</tbody>
</table>

The following shows how to reclassify a configuration item.

```javascript
var payload = {
  "items": [
    {
      "className": "cmdb_ci_win_server",
      "values": {
        "short_description": "Linux server description",
        "name": "Linux Server 1"
      }
    }
  ]
};
```
var input = new JSON().encode(payload);
var output = SNC.IdentificationEngineScriptableApi.createOrUpdateCI('ServiceNow', input);
gs.print(output);

Output:

```json
{
  "items": [
    {
      "className": "cmdb_ci_linux_server",
      "operation": "NO_CHANGE",
      "sysId": "440577800f321010150efc91ff767e94",
      "identifierEntrySysId": "556eb250c3400200d8d4bea192d3ae92",
      "identificationAttempts": [
        {
          "attributes": [],
          "info": "sys_object_source SKIPPED",
          "identifierName": "",
          "attemptResult": "SKIPPED",
          "hybridEntryCiAttributes": []
        },
        {
          "attributes": [
            "serial_number",
            "serial_number_type"
          ],
          "identifierName": "Hardware Rule",
          "attemptResult": "SKIPPED",
          "searchOnTable": "cmdb_serial_number",
          "hybridEntryCiAttributes": []
        },
        {
          "attributes": [
            "serial_number"
          ],
          "identifierName": "Hardware Rule",
          "attemptResult": "SKIPPED",
          "searchOnTable": "cmdb_ci_hardware",
          "hybridEntryCiAttributes": []
        }
      ]
    }
  ]
}
```
"identifierName": "Hardware Rule",
"attemptResult": "MATCHED",
"searchOnTable": "cmdb_ci_hardware",
"hybridEntryCiAttributes": []
},
"info": [
{
"message": "CI Reclassification not allowed from class: [cmdb_ci_linux_server] to [cmdb_ci_win_server] by a reclassification restriction rule",
"code": "SKIPPED_CLASS_SWITCH",
"ruleSysId": "b3d4b3800f321010150efc91ff767eab"
}
],
"errorCount": 0,
"warningCount": 0,
"markers": [],
"inputIndices": [0],
"mergedPayloadIds": []
"additionalCommittedItems": [],
"relations": [],
"additionalCommittedRelations": []
} 

var payload = {
"items": [
{
"className": "cmdb_ci_win_server",
"values": {
"chassis_type": "Desktop",
"os": "Windows 2012 R2 Datacenter",
"name": "Windows2012Server1",
"serial_number": "0000-0011-1690-8730-8636-5722-52",
"cpu_count": "1"
},
"lookup": [
{
"values": {
"valid": "true",
"
"serial_number": "0000-0011-1690-8730-8636-5722-52",
"serial_number_type": "bios"
},
"className": "cmdb_serial_number"
},
{
"values": {
"valid": "true",
"serial_number": "3311-9736-4988-9744-1749-4183-41",
"serial_number_type": "chassis"
},
"className": "cmdb_serial_number"
}
],
"internal_id": "16777219",
"sys_object_source_info": {
"source_feed": "SN Discovery Feed 1",
"source_name": "ServiceNow",
"source_native_key": "16777219",
"source_recency_timestamp": "2019-10-18 08:31:23"
},
{
"className": "cmdb_ci_spkg",
"values": {
"name": "Windows 2012 R2 Datacenter",
"key": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL"
}
},
"related": [
{
"internal_id": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL|16777219",
"values": {
"name": "Windows 2012 R2 Datacenter-SAMLABVM52"
},
"className": "cmdb_software_instance",
"sys_object_source_info": {
"source_feed": "SN Discovery Feed 1",
"source_name": "ServiceNow",
"source_native_key": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL|16777219"
}
]
)}}
}
"running_process_key_parameters": "/opt/OV/nonOV/tomcat/b/temp org.apache.catalina.startup.Bootstrap start",
"install_directory": "/opt/OV/nonOV/tomcat/b",
"name": "Tomcat@hpom9:3443",
"server_port": "8006",
"sys_class_name": "cmdb_ci_app_server_tomcat"
},

"internal_id": "tomcat_id"
],

"relations": [
{
  "parent_id": "tomcat_id",
  "child_id": "16777219",
  "type": "Runs on::Runs"
}
],

"referenceItems": [
{
  "referenceField": "installed_on",
  "referenced": "16777219",
  "referencedBy": "Microsoft Windows Server 2012 R2 Datacenter:::_NULL|16777219"
}]
];

var jsonUntil = new JSON();
var input = jsonUntil.encode(payload);
var output = SNC.IdentificationEngineScriptableApi.createOrUpdateCI('ServiceNow', input);
gs.print(output);

Output:

{  
  "items": [
  {
    "className": "cmdb_ci_win_server",
    "operation": "INSERT",
    "sysId": "d56ab6eadbd51002f67dfe5e96194e",
    "relatedSysIds": [
      "dd6af62adb190102f67dfe5e96197f",
      "996af62adb190102f67dfe5e961980"
    ],
    "relatedItems": [
    
```
"errors": [],
"operation": "INSERT",
"errorCount": 0,
"mergedPayloadIds": [],
"warningCount": 0,
"sysId": "dd6af62adb1910102f67dfea5e96197f",
"markers": [],
"inputIndices": [
  {
    "mainIndex": 0,
    "subIndex": 0
  }
],
"className": "cmdb_serial_number"
},
{
  "errors": [],
  "operation": "INSERT",
  "errorCount": 0,
  "mergedPayloadIds": [],
  "warningCount": 0,
  "sysId": "996af62adb1910102f67dfea5e961980",
  "markers": [],
  "inputIndices": [
    {
      "mainIndex": 0,
      "subIndex": 1
    }
  ],
  "className": "cmdb_serial_number"
}
],
"identifierEntrySysId": "Unknown",
"identificationAttempts": [
  {
    "info": "sys_object_source NO_MATCH",
    "identifierName": "",
    "attemptResult": "NO_MATCH",
    "attributes": [],
    "hybridEntryCiAttributes": []
  },
  {
    "identifierName": "Hardware Rule",
    "attemptResult": "NO_MATCH",
    
  }
"attributes": [
  "serial_number",
  "serial_number_type"
],
"searchOnTable": "cmdb_serial_number",
"hybridEntryCiAttributes": []
},
{
  "identifierName": "Hardware Rule",
  "attemptResult": "NO_MATCH",
  "attributes": [
    "serial_number"
  ],
  "searchOnTable": "cmdb_ci_hardware",
  "hybridEntryCiAttributes": []
},
{
  "identifierName": "Hardware Rule",
  "attemptResult": "NO_MATCH",
  "attributes": [
    "name"
  ],
  "searchOnTable": "cmdb_ci_hardware",
  "hybridEntryCiAttributes": []
},
{
  "identifierName": "Hardware Rule",
  "attemptResult": "SKIPPED",
  "attributes": [
    "mac_address",
    "name"
  ],
  "searchOnTable": "cmdb_ci_network_adapter",
  "hybridEntryCiAttributes": []
}
],
"errorCount": 0,
"mergedPayloadIds": [],
"warningCount": 0,
"markers": [],
"inputIndices": [0]
{  
  "className": "cmdb_ci_spkg",
  "operation": "INSERT",
  "sysId": "116af62adb1910102f67dfea5e961981",
  "relatedSysIds": [  
    "6d6af62adb1910102f67dfea5e961984"
  ],
  "relatedItems": [  
    {  
      "errors": [],
      "operation": "INSERT",
      "errorCount": 0,
      "mergedPayloadIds": [],
      "warningCount": 0,
      "sysId": "6d6af62adb1910102f67dfea5e961984",
      "markers": [],
      "inputIndices": [  
        {  
          "mainIndex": 1,
          "subIndex": 0
        }
      ],
      "className": "cmdb_software_instance"
    }
  ],
  "identifierEntrySysId": "Unknown",
  "identificationAttempts": [  
    {  
      "info": "sys_object_source SKIPPED",
      "identifierName": ",",
      "attemptResult": "SKIPPED",
      "attributes": [],
      "hybridEntryCiAttributes": []
    },
    {  
      "identifierName": "Software",
      "attemptResult": "NO_MATCH",
      "attributes": [  
        {  
          "key": 
        }
      ],
      "searchOnTable": "cmdb_ci_spkg",
      "hybridEntryCiAttributes": []
    }
  ]
}
"errorCount": 0,
"mergedPayloadIds": [],
"warningCount": 0,
"markers": [],
"inputIndices": [1],
},
{
  "className": "cmdb_ci_app_server_tomcat",
  "operation": "INSERT",
  "sysId": "e96af62adb1910102f67dfea5e961985",
  "identifierEntrySysId": "Unknown",
  "identificationAttempts": [
  {,
    "info": "sys_object_source SKIPPED",
    "identifierName": ",",
    "attemptResult": "SKIPPED",
    "attributes": [],
    "hybridEntryCiAttributes": []
  }
  ],
  "errorCount": 0,
  "mergedPayloadIds": [],
  "warningCount": 0,
  "markers": [],
  "inputIndices": [2],
  }
},
"additionalCommittedItems": [],
"relations": [,
  {,
    "className": "cmdb_rel_ci",
    "operation": "INSERT",
    "sysId": "e96af62adb1910102f67dfea5e961989",
    "identifierEntrySysId": "Unknown",
    "errorCount": 0,
    "mergedPayloadIds": [],
    "warningCount": 0,
    "markers": [],
    "inputIndices": [0]
Identify a dependent CI.

```javascript
var payload = {
    items: [
        {className: 'cmdb_ci_web_server',
            values: {name: 'apache linux den 200',
                running_process_command: 'xyz',
                running_process_key_parameters: 'abc',
                tcp_port: '3452'}},
        {className: 'cmdb_ci_linux_server',
            values: {name: 'lnux100', ram: '2048'}},
        relations: [{parent: 0, child: 1, type: 'Runs on::Runs'}]
    ];

var jsonUntil = new JSON();
var input = jsonUntil.encode(payload);
var output = SNC.IdentificationEngineScriptableApi.createOrUpdateCI('ServiceWatch', input);
gs.print(output);
```

Output:

```json
{
    "items": [
        {
            "className": "cmdb_ci_web_server",
            "operation": "INSERT",
            "sysId": "b9bb766adb1910102f67dfe5e961962",
            "identifierEntrySysId": "Unknown",
            "identificationAttempts": [
                {
                    "info": "sys_object_source SKIPPED",
                    "identifierName": "",
                    "attemptResult": "SKIPPED",
                    "attributes": [],
                    "hybridEntryCiAttributes": []
                },
                {
                    "info": "sys_object_source SKIPPED",
                    "identifierName": "",
```

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
"attemptResult": "SKIPPED",
"attributes": [],
"hybridEntryCiAttributes": []
],
"errorCount": 0,
"mergedPayloadIds": [],
"warningCount": 0,
"markers": [],
"inputIndices": [0]
],
{
"className": "cmdb_ci_linux_server",
"operation": "INSERT",
"sysId": "a5bb766adb1910102f67dfea5e96195b",
"identifierEntrySysId": "Unknown",
"identificationAttempts": [
{
"info": "sys_object_source SKIPPED",
"identifierName": "",
"attemptResult": "SKIPPED",
"attributes": [],
"hybridEntryCiAttributes": []
},
{
"identifierName": "Hardware Rule",
"attemptResult": "SKIPPED",
"attributes": [
   "serial_number",
   "serial_number_type"
],
"searchOnTable": "cmdb_serial_number",
"hybridEntryCiAttributes": []
},
{
"identifierName": "Hardware Rule",
"attemptResult": "SKIPPED",
"attributes": [
   "serial_number"
],
"searchOnTable": "cmdb_ci_hardware",
"hybridEntryCiAttributes": []
}]}
Identify an independent CI with lookup-based identification.

```javascript
var payload = {items: [
    {className:'cmdb_ci_netgear',
    values: {name:'ny8500-nbxs08',
    ports:'1200'},
    lookup: [{className:'cmdb_serial_number',
    values:{serial_number:'1234ABCD',
    serial_number_type:'uuid',absent:'false',valid:'true'}},
    {className:'cmdb_serial_number',
    values:{serial_number:'3456EFGH',
    serial_number_type:'system',absent:'false',valid:'true'}}]
  ];

var jsonUntil = new JSON();
var input = jsonUntil.encode(payload);
var output = SNC.IdentificationEngineScriptableApi.createOrUpdateCI('ServiceNow', input);
gs.print(output);
```

Output:

```javascript
{
    "items": [
        {
            "className": "cmdb_ci_netgear",
            "operation": "INSERT",
            "sysId": "787c7e6adb1910102f67dfeaa5e96196e",
            "relatedSysIds": [
                "f47c7e6adb1910102f67dfeaa5e961977",
                "3c7c7e6adb1910102f67dfeaa5e961977"
            ],
            "relatedItems": [
                {
                    "errors": [],
                    "operation": "INSERT",
                    "errorCount": 0,
                    "mergedPayloadIds": [],
                    "warningCount": 0,
                    "sysId": "f47c7e6adb1910102f67dfeaa5e961977",
                    "markers": [],
                    "inputIndices": [
```
{  
  "mainIndex": 0,  
  "subIndex": 0  
  
},  
"className": "cmdb_serial_number"  
},  
{  
  "errors": [],  
  "operation": "INSERT",  
  "errorCount": 0,  
  "mergedPayloadIds": [],  
  "warningCount": 0,  
  "sysId": "3c7c7e6adb1910102f67dfea5e961977",  
  "markers": [],  
  "inputIndices": [  
    {  
      "mainIndex": 0,  
      "subIndex": 1  
    }  
  ],  
  "className": "cmdb_serial_number"  
},  
"identifierEntrySysId": "Unknown",  
"identificationAttempts": [  
  {  
    "info": "sys_object_source SKIPPED",  
    "identifierName": ",",  
    "attemptResult": "SKIPPED",  
    "attributes": [],  
    "hybridEntryCiAttributes": []  
  },  
  {  
    "identifierName": "Hardware Rule",  
    "attemptResult": "NO_MATCH",  
    "attributes": [  
      "serial_number",  
      "serial_number_type"  
    ],  
    "searchOnTable": "cmdb_serial_number",  
    "hybridEntryCiAttributes": []  
  },  
  {  
    "identifierName": "Hardware Rule",  
    "attemptResult": "NO_MATCH",  
    "attributes": [  
      "serial_number",  
      "serial_number_type"  
    ],  
    "searchOnTable": "cmdb_serial_number",  
    "hybridEntryCiAttributes": []  
  }  
]
"identifierName": "Hardware Rule",
"attemptResult": "SKIPPED",
"attributes": [
  "serial_number"
],
"searchOnTable": "cmdb_ci_hardware",
"hybridEntryCiAttributes": []
},
{
  "identifierName": "Hardware Rule",
  "attemptResult": "NO_MATCH",
  "attributes": ["name"],
  "searchOnTable": "cmdb_ci_hardware",
  "hybridEntryCiAttributes": []
},
{
  "identifierName": "Hardware Rule",
  "attemptResult": "SKIPPED",
  "attributes": ["mac_address", "name"],
  "searchOnTable": "cmdb_ci_network_adapter",
  "hybridEntryCiAttributes": []
}
"errorCount": 0,
"mergedPayloadIds": [],
"warningCount": 0,
"markers": [],
"inputIndices": [0]
]}
],
"additionalCommittedItems": [],
"relations": [],
"additionalCommittedRelations": []
}
Scoped equivalent

To use the `createOrUpdateCI(String source, String input)` method in a scoped application, use the corresponding scoped IdentificationEngine method: `IdentificationEngine - createOrUpdateCI(String source, String input)`.

IdentificationEngineScriptableApi - `createOrUpdateCIEnhanced(String source, String input, Object options)`

Inserts or updates configuration items and non-Configuration Management Database (CMDB) configuration items (classes not extending from cmdb_ci) in the CMDB based on identification and reconciliation rules. Use this API instead of updating the CMDB directly.

In addition to providing the functionality of the `createOrUpdateCI()` method, this method also supports:

- Handling partial payloads
- Handling partial commits
- Removing duplicate items within a payload
- Generating output summaries

For additional information on IRE and more detailed explanations of the data used by this method, see Identification and Reconciliation (IRE).

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| input  | String    | Required. A JSON formatted string of configuration items to add or update.
| input.items | Array | Array of objects that define the items to add or update. |

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.className</td>
<td>String</td>
<td>Required. Class/table name, sys_class_name, of the configuration item (CI) to create or update. This value can be any CMDB class/table, such as cmdb_ci_linux_server or cmdb_ci_win_server.</td>
</tr>
<tr>
<td>input.items.display_values</td>
<td>Object</td>
<td>Reference fields to create or update for this related item as name-value pairs, where the name is the field name and the value is the referenced display value. If you want to use the sys_id instead of the display value for reference fields, pass the information in the input.items.lookup.values parameter instead of in this parameter. Reference field names depend on the fields selected by the user, such as:</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>input.items.internal_id</code></td>
<td>String</td>
<td>Unique item identifier for the associated payload. This can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td><code>input.items.lookup</code></td>
<td>Array</td>
<td>Identifies the top-level item containing the lookup (lookup-based identification). These records are used to identify the configuration item based on a lookup table that has a reference back to cmdb_ci. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>input.items.lookup.className</code></td>
<td>String</td>
<td>Required. Class/table name, sys_class_name, of the configuration item (CI) to create or update. This value can be any CMDB class, such as cmdb_ci_network_adapter.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>---------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>input.items.lookup.internal_id</code></td>
<td>String</td>
<td>Unique lookup identifier for the associated payload. This can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td><code>input.items.lookup.sys_object_source_info</code></td>
<td>Object</td>
<td>Defines a unique CI identifier for a specific data source. Different sources may have different name-value pairs.</td>
</tr>
<tr>
<td><code>input.items.lookup.sys_object_source_info.source_feed</code></td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide the name of the feed sending this item. The data source generates this feed name. It can be any string that uniquely identifies the source.</td>
</tr>
<tr>
<td><code>input.items.lookup.sys_object_source_info.source_name</code></td>
<td>String</td>
<td>Data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td><code>input.items.lookup.sys_object_source_info.source_native_key</code></td>
<td>String</td>
<td>Unique key-id for the item from the source.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.lookup.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date/time that the item was scanned. Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>input.items.lookup.values</td>
<td>Object</td>
<td>Field information for the CI as name-value pairs, where the name is the field name. When updating reference fields, the value must be the referenced sys_id. Field names and types depend on the fields selected, such as:</td>
</tr>
<tr>
<td>input.items.related</td>
<td>Array</td>
<td>Reference to the top-level item that contains the related list. Rules in the Related Entry [cmdb_related_entry] define what type of records can be in this array. These records are used to add items based on a related table that has a reference to the CI being identified.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.related.className</td>
<td>String</td>
<td>Required. Class/table name, sys_class_name, of the configuration item (CI) to create or update. This value can be any CMDB class/table, such as cmdb_software_instance or cmdb_key_value.</td>
</tr>
<tr>
<td>input.items.related.internal_id</td>
<td>String</td>
<td>Unique identifier for this related item in this payload. Can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info</td>
<td>Object</td>
<td>Object that makes up a unique CI identifier for a specified data source. Different sources may have different name-value pairs for the same CI.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>input.items.related.sys_object_source_info.source_feed</code></td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide the name of the feed sending this item. The data source generates this feed name. It can be any string that uniquely identifies the source feed.</td>
</tr>
<tr>
<td><code>input.items.related.sys_object_source_info.source_name</code></td>
<td>String</td>
<td>Identifies the data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td><code>input.items.related.sys_object_source_info.source_native_key</code></td>
<td>String</td>
<td>Unique key/ID from the source for the related item. The data source generates this key. It can be any string that is unique to the item.</td>
</tr>
<tr>
<td><code>input.items.related.sys_object_source_info.source_recency_timestamp</code></td>
<td>String</td>
<td>UTC date and time that the item was scanned. Format: <code>YYYY-MM-DD hh:mm:ss</code></td>
</tr>
<tr>
<td><code>input.items.related.values</code></td>
<td>Object</td>
<td>Fields to create or update for this related item as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced sys_id. If you want to use the <code>sys_id</code> field, you must provide the sys_id of the related CI.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>display value instead of the sys_id for reference fields, pass this information in a display_values object instead of in the values object. Field names and types depend on the fields selected as:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>values:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;host_name&quot;: &quot;String&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;ip_address&quot;: &quot;String&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;name&quot;: &quot;String&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;os_name&quot;: &quot;String&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;sys_class_name&quot;: &quot;String&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>input.items.settings</td>
<td>Object</td>
<td>Parameters that define the types of updates that are permitted.</td>
</tr>
<tr>
<td>&quot;settings&quot;:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;skipReclassificationRestrictionRules&quot;: Boolean,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;updateWithoutDowngrade&quot;: Boolean,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;updateWithoutSwitch&quot;: Boolean,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;updateWithoutUpgrade&quot;: Boolean</td>
<td></td>
<td></td>
</tr>
<tr>
<td>input.items.settings.skipReclassificationRestrictionRules</td>
<td>Boolean</td>
<td>Flag that IRE should not run the Reclassification Restriction rule that is defined for the payload item.</td>
</tr>
<tr>
<td>Valid values:</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| input.items.settings.updateWithoutDowngrade | Boolean | Flag that indicates whether update and downgrade are both permitted for this item. Valid values:  
- true: Update the item but downgrade is not permitted.  
- false: Both item update and downgrade are permitted.  
Default: false |
| input.items.settings.updateWithoutSwitch | Boolean | Flag that indicates whether the item can be updated and the class switched. Valid values:  
- true: Update the item but class switching is not permitted.  
- false: Both item update and class switching are permitted.  
Default: false |
| input.items.settings.updateWithoutUpgrade | Boolean | Flag that indicates whether update and upgrade are both permitted for this item. Valid values:  
- true: Skip running the rule.  
- false: Run the rule.  
Default: false |
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.sys_object_source_info</td>
<td>Object</td>
<td>Unique CI identifier for a specific source.</td>
</tr>
</tbody>
</table>
| input.items.sys_object_source_info.source_feed            | String  | If the source can have multiple feeds, use this field to provide the feed name. 
|                                                           |         | The data source generates this feed name. It can be any string that uniquely identifies the source. |
| input.items.sys_object_source_info.source_name            | String  | Data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item (cmdb_ci) table. |
| input.items.sys_object_source_info.source_native_key      | String  | Unique key/id for the item from the source.                                |

- true: Update the item but upgrade is not permitted.
- false: Both item update and upgrade are permitted.

Default: false
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date and time that the item was scanned. Format: YYYY-MM-DD hh:mm:ss.</td>
</tr>
<tr>
<td>input.items.values</td>
<td>Object</td>
<td>Fields to create or update for this related item as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced sys_id. If you want to use the display value instead of the sys_id for reference fields, pass this information in a <code>display_values</code> object instead of in the values object. Field names and types depend on the fields selected as:</td>
</tr>
<tr>
<td>input.referenceltems</td>
<td>Array</td>
<td>Array of objects that define references between items in the input payload.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.referencedItems.referenced</td>
<td>String</td>
<td>The <code>internal_id</code> defined for the item being referenced by another item.</td>
</tr>
<tr>
<td>input.referencedItems.referencedBy</td>
<td>String</td>
<td>The <code>internal_id</code> defined for the item that references another item.</td>
</tr>
<tr>
<td>input.referencedItems.referenceField</td>
<td>String</td>
<td>Name of the reference field in the class/table for the <code>referencedBy</code> item.</td>
</tr>
</tbody>
</table>
| input.relations | Array | Array of objects that specify relationships between items in the input payload. An object in this array can use either of two formats.  
- The object can define a relationship between two top-level items (only) using `parent` and `child` name-value pairs, with values representing item indexes from the payload. |

```json
"relations": [
  {
    "child": Number,
    "parent": Number,
    "sys_rel_source_info": {Object}
  }
]
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.relations.child</td>
<td>Number</td>
<td>Integer index of the CI object in the items array that represents the child in the relationship (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td>input.relations.child_id</td>
<td>String</td>
<td>The <code>internal_id</code> of the child item in the relationship (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td>input.relations.parent</td>
<td>Number</td>
<td>Integer index of the parent item in the items array (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>input.relations.parent_id</td>
<td>String</td>
<td>The <em>internal_id</em> of the parent item in the relation (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td>input.relations.sys_rel_source_info</td>
<td>Object</td>
<td>Discovery source information for the relationship. For non-dependency relationships, this information is saved in the Relation Sources [sys_rel_source] table (not persisted for identifyCIEnhanced() or identifyCI() methods.).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>input.relations.sys_rel_source_info.source_name</td>
<td>String</td>
<td>Discovery source name. Default: Discovery source passed in the API method parameter.</td>
</tr>
<tr>
<td>input.relations.sys_rel_source_info.source_feed</td>
<td>String</td>
<td>Any string that is a sub-discovery/scan within the discovery source. Default: 'UNKNOWN' is stored in the source_feed column when creating a record in the sys_rel_source table.</td>
</tr>
<tr>
<td>input.relations.type</td>
<td>String</td>
<td>Type of relationship that exists between the parent and child items. Should be a name field value from</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional, but {} or null must be passed-in. Options to enable or disable features.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>options.deduplicate_payloads</strong>: Boolean Flag that indicates whether duplicate items are merged or considered errors.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- true: Merge duplicate items.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- false: Consider duplicate items as errors.</td>
</tr>
</tbody>
</table>

**Note:**
- If `partial_commits` is set to false, `partial_payloads` are enabled as they are essential for partial payloads functionality. If `partial_payloads` is set to false, `partial_commits` are enabled.
- `options`: {
  - **deduplicate_payloads**: Boolean
  - **generate_summary**: Boolean
  - **partial_commits**: Boolean
  - **partial_payloads**: Boolean
}

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| options.generate_summary  | Boolean| Flag that indicates whether the returned results contain summary information. For the details of the returned summary, see `<String>` in the return results table. Valid values:     
|                           |       | • true: Include summary information.                                      |
|                           |       | • false: Do not include summary information.                                |
|                           |       | Default: false                                                              |
| options.partial_commits   | Boolean| Flag that indicates whether partial commit support is enabled. For additional information on partial commits, see Enhanced IRE features. Valid values:       
|                           |       | • true: Partial commit enabled.                                              |
|                           |       | • false: Partial commit disabled.                                             |
|                           |       | Default: true                                                               |
| options.partial_payloads  | Boolean| Flag that indicates whether partial payload support is enabled. For additional information on partial payloads, see Create an IRE data source rule. Valid values:       
|                           |       | • true: Partial payload enabled.                                             |
|                           |       | • false: Partial payload disabled.                                           |
|                           |       | Default: true                                                               |
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Valid value:  
- true: Partial payload enabled.  
- false: Partial payload disabled.  
Default: true | | |
| source | String | Data source of the CI information. This value must be one of the values defined for the discovery_source field of the Configuration Item [cmdb_ci] table. |

## Returns

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| <String> | JSON formatted string that is a list of results for the configuration items in the input string.  
Data type:  
```json
{
  "additionalCommittedItems": [Array],  
  "additionalCommittedRelations": [Array],  
  "hasError": "Boolean",  
  "hasWarning": "Boolean",  
  "items": [Array],  
  "relations": [Array],  
  "summary": Object
}
```
| <String>.additionalCommittedItems | List of CIs that were committed during the IRE processing of the current payload. |
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>not present in the current input payload.</td>
<td></td>
</tr>
</tbody>
</table>

Data type:

```
{  
  class_name: "String",  
  errorCount: Number,  
  operation: "String",  
  identificationAttempts: [Array],  
  inputIndices: [Array],  
  markers: [Array],  
  mergedPayloads: [Array],  
  sysId: "String"  
}
```

**<String>.additionalCommittedItems.className**  
Sys_class_name of this additional CI.  
Data type: String

**<String>.additionalCommittedItems.errorCount**  
Number of errors encountered while processing this additional CI.  
Data type: Number

**<String>.additionalCommittedItems.errors**  
Array of objects that describes errors encountered while processing this additional CI.  
Data type: Array

```
{  
  error: "String",  
  message: "String"  
}
```

**<String>.additionalCommittedItems.errors.error**  
Type of error encountered while processing the additional CI.  
Data type: String
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.additionalCommittedItems.errors.message</code></td>
<td>Error message encountered while processing the additional CI. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedItems.identificationAttempts</code></td>
<td>Array of objects in which each object describes an attempt made to identify this additional CI. Data type: Array</td>
</tr>
</tbody>
</table>
| `<String>.additionalCommittedItems.identificationAttempts.attemptResult` | Outcome of this additional CI identification attempt. Possible values:  
- MATCHED: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.  
- MULTI_MATCH: Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.  
- NO_MATCH: Identification failed. No CI was found in the identifier rule table which matched the specified attributes. |
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• SKIPPED: Identification not attempted. The attributes required for this identifier rule table search were not provided, so the rule was not applied.</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.identificationAttempts.attributes</td>
<td>Array of CI identifier entry attributes used during this additional CI identification attempt. Data type: Array. Attribute names and types depend on the request body data and the identifier in use, such as:</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.identificationAttempts.hybridEntryCiAttributes</td>
<td>Array of CI identifier entry attributes used during this additional CI identification attempt. Data type: Array. Attribute names and types depend on the request body data and the identifier in use, such as:</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.identificationAttempts.identifierName</td>
<td>Identifier rule used for this additional CI identification attempt. Data type: String.</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.identificationAttempts.searchOnTable</td>
<td>Name of the table searched for this additional CI identification attempt. Data type: String.</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.additionalCommittedItems.identifierEntrySysId</code></td>
<td>Sys_id for the identifier rule used to identify this additional CI. Notable values: • Unknown: Identification of this additional CI failed. See errors for details.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedItems.inputIndices</code></td>
<td>Array of index values for CIs from the request body items array that correspond to this additional CI. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedItems.markers</code></td>
<td>Array of marker values for internal use. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedItems.mergedPayloads</code></td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this additional CI. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedItems.operation</code></td>
<td>Operation performed for this additional CI. Data type: String Possible values:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Returns (continued)

### Parameter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• DELETE</td>
<td>An existing CI is removed from the target table.</td>
</tr>
<tr>
<td>• INSERT</td>
<td>The additional CI is inserted into the target table as a new record.</td>
</tr>
<tr>
<td>• NO_CHANGE</td>
<td>No operation is performed for the additional CI.</td>
</tr>
<tr>
<td>• UPDATE</td>
<td>An existing CI in the target table is updated.</td>
</tr>
<tr>
<td>• UPDATE_WITH_DOWNGRADE</td>
<td>An existing CI in the target table is updated and its class is changed to a more generic class (ancestor class).</td>
</tr>
<tr>
<td>• UPDATE_WITH_SWITCH</td>
<td>An existing CI in the target table is updated and its class is changed to another class which is not an ancestor or descendent class.</td>
</tr>
<tr>
<td>• UPDATE_WITH_UPGRADE</td>
<td>An existing CI in the target table is updated and its class is changed to a more specialized class (descendent class).</td>
</tr>
</tbody>
</table>

*<String>.additionalCommittedItems.sysId*<br>

**Sys_id** found for this additional CI through identification.  
Data type: String  
Notable values:  
• Unknown: Identification of this additional CI failed. See errors for details.

*<String>.additionalCommittedItems.warnings*<br>

Array of objects that describe a warning encountered while processing this additional CI.  
Data type: Array
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.warnings.error</td>
<td>Type of warning encountered while processing this additional CI. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedItems.warnings.message</td>
<td>Warning message encountered while processing this additional CI. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedRelations</td>
<td>Array of objects that describe a dependent relationship CI that was not included in the request body relations list to insert or update. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.additionalCommittedRelations.className</td>
<td>The sys_class_name of this additional dependent relationship CI. Data type: String Only supported value: • cmdb_rel_ci: The CI Relationship table</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations.errorCount</code></td>
<td>Number of errors encountered while processing this additional dependent relationship CI. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations.errors</code></td>
<td>Array of objects that describe errors encountered while processing this additional dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td><code>{ &quot;errors&quot;: [ { &quot;error&quot;: &quot;String&quot;, &quot;message&quot;: &quot;String&quot; } ] }</code></td>
<td>Type of error encountered while processing the additional dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations.errors.error</code></td>
<td>The error type encountered while processing this additional dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations.errors.message</code></td>
<td>The error message encountered while processing this additional dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations.inputIndices</code></td>
<td>Index values for dependent relationship CI objects in the request body relations array that correspond to this additional dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td><code>{ &quot;inputIndices&quot;: [ Number ] }</code></td>
<td>Marker values for internal use. Data type: Marker</td>
</tr>
<tr>
<td><code>&lt;String&gt;.additionalCommittedRelations.markers</code></td>
<td>Marker values for internal use. Data type: Marker</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.additionalCommittedRelations.mergedPayloadIds</td>
<td>Sys_id values for partial payloads from the [cmdb_ire_partial_payloads] table that were merged during processing of this additional dependent relationship CI.</td>
</tr>
</tbody>
</table>
| <String>.additionalCommittedRelations.operation | Operation that was performed for the additional dependent relationship CI. Possible values:  
  - INSERT: The dependent relationship CI is inserted into the target table as a new record.  
  - INSERT_AS_INCOMPLETE: The dependent relationship CI had errors and is inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.  
  - INSERT_AS_PARTIAL: The dependent relationship CI had errors and is inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.  

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• NO_CHANGE: No operation is performed for the dependent relationship CI.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE: An existing dependent relationship CI in the target table is updated.</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.hasError</td>
<td>Flag that indicates whether any item or relation has errors.</td>
</tr>
<tr>
<td>&lt;String&gt;.hasWarning</td>
<td>Flag that indicates whether any item or relation has warnings.</td>
</tr>
<tr>
<td>&lt;String&gt;.items</td>
<td>Array of objects that describe the created or updated CIs.</td>
</tr>
</tbody>
</table>

```json
"items": [
  {
    "additionalRelatedItems": [Array],
    "className": "String",
    "duplicateIndices": [Array],
    "errorCount": Number,
    "errors": [Array],
    "identificationAttempts": [Array],
    "identifierEntrySysId": "String",
    "incompleteSysIds": "String",
    "info": [Array],
    "inputIndices": [Array],
    "maskedAttributes": [Array],
    "operation": "String",
    "partialSysIds": "String",
    "relatedItems": [Array],
    "relatedSysIds": [Array],
    "sysId": "String"
  }
]```
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.additionalRelatedItems</code></td>
<td>List of JSON objects that provide information about additional lookup and related items that were processed but not provided as part of the input payload. These items are from partial payloads. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.additionalRelatedItems.className</code></td>
<td>Class/table name (sys_class_name) of the CI that was created or updated. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.additionalRelatedItems.inputIndices</code></td>
<td>Index values for CIs from the request body items array that correspond to this related item. Data type: Array of Numbers</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.additionalRelatedItems.mergedPayloadIds</code></td>
<td>List of sys_ids of the partial payloads that were merged into the related item. Located in the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.additionalRelatedItems.operation</code></td>
<td>Type of operation. Possible values:</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• INSERT: New CI was inserted into the database.</td>
<td></td>
</tr>
<tr>
<td>• NO_CHANGE: No CI changes were made.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE: Existing CI was updated.</td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.additionalRelatedItems.sysId</td>
<td>Sys_id of the CI that was updated or created.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.className</td>
<td>Class/table name (sys_class_name) of the CI that was created or updated.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.duplicateIndices</td>
<td>List of indexes of CIs that are duplicates of the current item.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.errorCount</td>
<td>Number of errors encountered while processing the CI.</td>
</tr>
<tr>
<td>Data type: Number</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.errors</td>
<td>Array of objects in which each object describes an error encountered while processing this CI.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td><em>errors</em></td>
<td></td>
</tr>
<tr>
<td><em>error</em></td>
<td></td>
</tr>
<tr>
<td><em>message</em></td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.items.errors.error</td>
<td>Type of error encountered while processing the CI.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>&lt;String&gt;.items.errors.message</td>
<td>Error message associated with the error. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.identificationAttempts</td>
<td>List of attempts that were made to identify CIs. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"identificationAttempts": [
  {
    "attemptResult": "String",
    "attributes": [Array],
    "hybridEntryCiAttributes": [Array],
    "identifierName": "String",
    "searchOnTable": "String"
  }
]
```

<table>
<thead>
<tr>
<th>&lt;String&gt;.items.identificationAttempts.attemptResult</th>
<th>Results of the attempt to identify the CI. Possible values:</th>
</tr>
</thead>
<tbody>
<tr>
<td>MATCHED</td>
<td>Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.</td>
</tr>
<tr>
<td>MULTI_MATCH</td>
<td>Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.</td>
</tr>
<tr>
<td>NO_MATCH</td>
<td>Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
</tr>
<tr>
<td>SKIPPED</td>
<td>Identification not attempted. The attributes could not be used to identify the CI.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.attributes</td>
<td>List of CI identifier entry attributes that were used during the identification process. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.hybridEntryCiAttributes</td>
<td>List of CI identifier entry attributes that were used during the identification process. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.identifierName</td>
<td>Identifier rule used for this CI identification attempt. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.searchOnTable</td>
<td>Name of the table searched during the identification process. Data type: String</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.identifierEntrySysId</td>
<td>Sys_id for the identifier rule used to identify the CI. Located in the Identifier Entry [cmdb_identifier_entry] table. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.incompleteSysIds</td>
<td>If the item was saved as an incomplete payload, this parameter contains the sys_id of the record in the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.info</td>
<td>List of objects that contains additional information about the processing of the item. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;info&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
</tbody>
</table>
|   |     "code": "String",
|   |     "message": "String",
|   |     "ruleSysId": "String"
|   |   }
<p>|   | ] |
| &lt;String&gt;.items.info.code | Reclassification type that was skipped. Possible values: SKIPPED_CLASS_SWITCH, SKIPPED_CLASS_DOWNGRADE, SKIPPED_CLASS_UPGRADE Data type: String |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.info.message</code></td>
<td>Message that provides additional insights into the reason for skipping the reclassification. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.info.ruleSysId</code></td>
<td>Sys_id of the reclassification restriction rule that was matched. Applicable only when IRE skips reclassification due to reclassification restriction rule. This value is empty if the reclassification is skipped due to a payload or global flag. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.inputIndices</code></td>
<td>Indexes of the corresponding input CI. For top-level items, it is a list of integers. For related or lookup CIs, it is a list of JSON objects. Data type: Array of Numbers</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.maskedAttributes</code></td>
<td>List of attributes whose update by a non-authoritative data source was skipped as defined by the Reconciliation Rules. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.operation</code></td>
<td>Operation that took place. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• INSERT: New CI was inserted into the database.</td>
</tr>
<tr>
<td></td>
<td>• INSERT_AS_INCOMPLETE: Item was saved in <code>cmdb_ire_incomplete_payloads</code> table.</td>
</tr>
<tr>
<td></td>
<td>• INSERT_AS_PARTIAL: Item was saved in <code>cmdb_ire_partial_payloads</code> table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>• NO_CHANGE: No CI changes were made.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE: Existing CI was updated.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE_WITH_DOWNGRADE: CI was updated and the class changed to a more generic class (ancestor class).</td>
<td></td>
</tr>
<tr>
<td>• UPDATE_WITH_SWITCH: CI was updated and the class changed to a class that is not ancestor or descendent.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE_WITH_UPDRADE: CI was updated and the class changed to a more specialized class (descendent class).</td>
<td></td>
</tr>
</tbody>
</table>

**Data type:** String

`<String>.items.partialSysIds`

If the item had errors and was saved as a partial payload, this parameter contains the sys_id of the partial payload record.

**Data type:** String

`<String>.items.relatedItems`

List of JSON objects that describes a related CI (table lookup CI) from the request body `items.lookup` array.

**Data type:** Array

```json
"relatedItems": [
  {
    "className": "String",
    "errors": [Array],
    "errorCount": Number,
    "incompleteSysIds": [Array],
    "inputIndices": [Array],
    "mergePayloadIds": [Array],
    "operation": "String"
  }
]
```
Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.relatedItems.className</td>
<td>Class/table name (sys_class_name) of the related item. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.errors</td>
<td>List of errors that occurred during processing of the related item. Data type: Array</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.errors.error</td>
<td>Type of error encountered while processing the related item. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.errors.message</td>
<td>Error message associated with the error. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.errorCount</td>
<td>Number of errors detected while processing the related items. Data type: Number</td>
</tr>
<tr>
<td>&lt;String&gt;.items.relatedItems.incompleteSysIds</td>
<td>If the related item was saved as an incomplete payload, this value is the sys_id of the record in the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| `<String>.items.relatedItems.inputIndices` | Index of the corresponding input item. For top-level items, it is a list of integers. For related or lookup items, it is a list of JSON objects.  
Data type: Array of Numbers or Array of Objects |
| `<String>.items.relatedItems.inputIndices.mainIndex` | Index value from the request body items array that corresponds to the CI parent of the related item.  
Data type: Number |
| `<String>.items.relatedItems.inputIndices.subIndex` | Index value from the request body items.lookup array that corresponds to the related item.  
Data type: Number |
| `<String>.items.relatedItems.mergedPayloadIds` | List of sys_ids of the partial payloads that were merged into the CI during processing.  
Data type: Array |
| `<String>.items.relatedItems.operation` | Operation that took place. Possible values:  
- INSERT: New related CI was inserted into the database.  
- INSERT_AS_INCOMPLETE: Item was saved in cmdb_ire_incomplete_payloads table. |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• INSERT</td>
<td>Item was saved in cmdb_ire_partial_payloads table.</td>
</tr>
<tr>
<td>• NO_CHANGE</td>
<td>No related CI changes were made.</td>
</tr>
<tr>
<td>• UPDATE</td>
<td>Existing related CI was updated.</td>
</tr>
<tr>
<td>• UPDATE_WITH_DOWNGRADE</td>
<td>Related CI was updated and the class changed to a more generic class (ancestor class).</td>
</tr>
<tr>
<td>• UPDATE_WITH_SWITCH</td>
<td>Related CI was updated and the class changed to a class that is not ancestor or descendent.</td>
</tr>
<tr>
<td>• UPDATE_WITH_UPDRADE</td>
<td>Related CI was updated and the class changed to a more specialized class (descendent class).</td>
</tr>
</tbody>
</table>

Data type: String

```<String>.items.relatedItems.partialSysIds```

If the related item had errors and was saved as a partial payload, this contains a list of the sys_ids of the associated records in the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.

Data type: Array

```<String>.items.relatedItems.sysId```

The sys_id of the related item.

Data type: String

```<String>.items.relatedItems.warningCount```

Number of warnings encountered when processing the related items.

Data type: Number
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.warnings</code></td>
<td>Array of objects that describes a warning encountered while processing the related items. Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.warnings.error</code></td>
<td>Type of warning encountered while processing the related item. Data type: String</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedItems.warnings.message</code></td>
<td>Message associated with the warning. Data type: String</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.relatedSysIds</code></td>
<td>List of the CIs used during lookup-based identification of related items. Data type: String</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.sys_id</code></td>
<td>Sys_id of the CI that was updated or created. Data type: String</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations</code></td>
<td>List of JSON objects that describe a dependent relationship CI from the request body relations array. Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.relations.className</code></td>
<td>Sys_class_name of this dependent relationship CI. Only supported value: • cmdb_rel_ci: CI Relationship table.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations.errorCount</code></td>
<td>Number of errors encountered when processing the dependent relationship CI.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations.errors</code></td>
<td>Array of objects that describe errors that were encountered while processing this dependent relationship CI.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations.errors.error</code></td>
<td>Type of error encountered while processing the dependent relationship CI.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations.errors.message</code></td>
<td>Error message encountered while processing this dependent relationship CI.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations.incompleteSysIds</code></td>
<td>If the relation was saved as an incomplete payload, this value is the sys_id of the record in the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.</td>
</tr>
<tr>
<td><code>&lt;String&gt;.relations.inputIndices</code></td>
<td>Indexes for the dependent relationship CI objects in the request body relations array that correspond to this dependent relationship CI.</td>
</tr>
</tbody>
</table>
| `<String>.relations.operation`          | Type of operation performed. Possible values:  
  - INSERT: The dependent relationship CI was inserted into the target table as a new record.  
  - INSERT_AS_INCOMPLETE: The dependent relationship CI had errors and was inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.  
  - INSERT_AS_PARTIAL: The dependent relationship CI had errors and was inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.  

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• NO_CHANGE</td>
<td>No operation was performed.</td>
</tr>
<tr>
<td>• UPDATE</td>
<td>An existing dependent relationship CI in the target table was updated.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.relations.partialSysIds</td>
<td>If the relation had errors and was saved as a partial payload, this value is the sys_id of the record in the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.relations.sysId</td>
<td>Sys_id of the dependent relationship CI.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.summary</td>
<td>List of JSON properties that provide statistics on how many items were inserted, updated, etc., per class.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>&lt;String&gt;.summary.&lt;class_name&gt;</td>
<td>Statistics for a specific class.</td>
</tr>
<tr>
<td>Data type: Object</td>
<td></td>
</tr>
</tbody>
</table>

```json
<class_name>: {
    "additionalInsertedItemCount": Number,
    "errorCount": Number,
    "incompleteItemCount": Number,
    "insertedItemCount": Number,
    "partialItemCount": Number,
    "skippedItemCount": Number,
    "unchangedItemCount": Number,
    "updatedItemCount": Number,
}
```
## Returns (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.additionalInsertedItemCount</code></td>
<td>Number of items inserted due to processing of partial payloads. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.errorCount</code></td>
<td>Number of errors encountered when processing items. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.incompleteltemCount</code></td>
<td>Number of items inserted in the CMDB IRE Incomplete Payloads <code>[cmdb_ire_incomplete_payloads]</code> table. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.insertedItemCount</code></td>
<td>Number of items created. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.partialItemCount</code></td>
<td>Number of items saved in the Partial Payload table <code>[cmdb_ire_partial_payloads]</code>. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.skippedItemCount</code></td>
<td>Number of items that were skipped. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.unchangedItemCount</code></td>
<td>Number of items that had entries but were not modified. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.updatedItemCount</code></td>
<td>Number of items updated. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;String&gt;.summary.&lt;class_name&gt;.warningCount</code></td>
<td>Number of items that generated a warning when processed. Data type: Number</td>
</tr>
</tbody>
</table>

Identify a dependent CI.
```javascript
var payload = {
  "items": [
    {
      "className": "cmdb_ci_win_server",
      "values": {
        "chassis_type": "Desktop",
        "os": "Windows 2012 R2 Datacenter",
        "name": "Windows2012Server1",
        "serial_number": "0000-0011-1690-8730-8636-5722-52",
        "cpu_count": "1"
      },
      "lookup": [
        {
          "values": {
            "valid": "true",
            "serial_number": "0000-0011-1690-8730-8636-5722-52",
            "serial_number_type": "bios"
          },
          "className": "cmdb_serial_number"
        },
        {
          "values": {
            "valid": "true",
            "serial_number": "3311-9736-4988-9744-1749-4183-41",
            "serial_number_type": "chassis"
          },
          "className": "cmdb_serial_number"
        }
      ],
      "internal_id": "16777219",
      "sys_object_source_info": {
        "source_feed": "SN Discovery Feed 1",
        "source_name": "ServiceNow",
        "source_native_key": "16777219",
        "source_recency_timestamp": "2019-10-18 08:31:23"
      }
    },
    {
      "className": "cmdb_ci_spkg",
      "values": {
        "name": "Windows 2012 R2 Datacenter",
        "key": "Microsoft Windows Server 2012 R2 Datacenter:::NULL"
      }
    }
  ]
};
```
"related": [
{
"internal_id": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL|16777219",
"values": {
  "name": "Windows 2012 R2 Datacenter-SAMLABVM52"
},
"className": "cmdb_software_instance",
"sys_object_source_info": {
  "source_feed": "SN Discovery Feed 1",
  "source_name": "ServiceNow",
  "source_native_key": "Microsoft Windows Server 2012 R2 Datacenter_:::_NULL|16777219"
}
},
{
"className": "cmdb_ci_app_server_tomcat",
"values": {
  "running_process_key_parameters": "/opt/OV/nonOV/tomcat/b/temp org.apache.catalina.startup.Bootstrap start",
  "install_directory": "/opt/OV/nonOV/tomcat/b",
  "name": "Tomcat@hpom9:3443",
  "server_port": "8006",
  "sys_class_name": "cmdb_ci_app_server_tomcat"
}
},
"internal_id": "tomcat_id"
}],
"relations": [
{
  "parent_id": "tomcat_id",
  "child_id": "16777219",
  "type": "Runs on::Runs"
}
],
"referenceItems": [
{
  "referenceField": "installed_on",
  "referenced": "16777219"
}]
}
var input = new JSON().encode(payload);
var output = SNC.IdentificationEngineScriptableApi.createOrUpdateCIEnhanced('ServiceNow', input, {});

Output:

```json
{
  "items": [
    {
      "className": "cmdb_ci_win_server",
      "operation": "NO_CHANGE",
      "sysId": "65d873d2b3a0001028f6eae2c6a8dc2a",
      "relatedSysIds": [
        "a1d873d2b3a0001028f6eae2c6a8dc32",
        "a1d873d2b3a0001028f6eae2c6a8dc33"
      ],
      "relatedItems": [
        {
          "className": "cmdb_serial_number",
          "sysId": "a1d873d2b3a0001028f6eae2c6a8dc32",
          "markers": [],
          "inputIndices": [
            {
              "mainIndex": 0,
              "subIndex": 0
            }
          ],
        },
        {
          "className": "cmdb_serial_number",
          "sysId": "a1d873d2b3a0001028f6eae2c6a8dc33",
          "markers": [],
          "inputIndices": [
            {
              "mainIndex": 0,
              "subIndex": 1
            }
          ],
        }
      ],
      "additionalRelatedItems": [],
      "identificationAttempts": []
    }
  ]
}
```
null
"identifierEntrySysId": "Unknown",
"identificationAttempts": [
{
  "attributes": [
    "install_directory",
    "running_process_key_parameters",
    "sys_class_name"
  ],
  "identifierName": "Tomcat",
  "attemptResult": "NO_MATCH",
  "searchOnTable": "cmdb_ci_app_server_tomcat",
  "hybridEntryCiAttributes": []
},
{
  "attributes": [
    "cl_port",
    "sys_class_name"
  ],
  "identifierName": "Application Rule",
  "attemptResult": "SKIPPED",
  "searchOnTable": "cmdb_ci_appl",
  "hybridEntryCiAttributes": []
},
{
  "attributes": [
    "running_process_command",
    "running_process_key_parameters",
    "sys_class_name"
  ],
  "identifierName": "Application Rule",
  "attemptResult": "NO_MATCH",
  "searchOnTable": "cmdb_ci_appl",
  "hybridEntryCiAttributes": []
}],
"errorCount": 0,
"inputIndices": [2]
],
"additionalCommittedItems": [],
"relations": [
{
  "className": "cmdb_rel_ci",
  "classType": "cmdb_ci_app_server_tomcat"
}]
}
IdentificationEngineScriptableApi - identifyCI(String jsonString)

Determines the operation (insert/update) to perform with the specified payload without committing the operation in the database.

This works just like createOrUpdateCI(), but does not commit the result.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| jsonString | String| A JSON formatted string of configuration items to be added or updated. Each input string is in the format 'items: [{}], relations:[{}]', where each item within the items and relations lists contains name-value pairs. The possible name-value pairs within the items list are:

  - className - Sys_class_name of the CI to be created or updated.
  - values:[] - Field information for the CI as name-value pairs, where the name is the field name.
  - lookup:[{}] - List of records with each item having name-value pairs like the items list.

The possible name-value pairs within the relations list are:

  - parent - Index of the parent item in the dependency relation
  - child - Index of the child item in the dependency relation
  - type - Relationship type. This is one of the name field values from the cmdb_rel_type table. |
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String | A JSON formatted string that is a list of results. Each result string is in the format 'items: [{}], relations:[{}]', where each item within the items and relations lists contains name-value pairs. The possible name-value pairs within the items list are:  
  • className - Sys_class_name for the CI that was updated or created.  
  • operation - Operation to perform, which is one of the following:  
    ◦ DELETE  
    ◦ INSERT  
    ◦ NO_CHANGE  
    ◦ UPDATE  
    ◦ UPDATE_WITH_DOWNGRADE  
    ◦ UPDATE_WITH_SWITCH  
    ◦ UPDATE_WITH_UPGRADE  
  • sysId - Sys_id of the CI that was updated or created.  
  • relatedSysIds - List of sys_id values of CIs used during lookup based identification.  
  • identifierEntrySysId - Sys_id of identifier entry used during matching.  
  • errors - List of errors in the format of (error, message string)  
  • duplicateIndices - List of indexes of items that are duplicates of the current item.  
  • identificationAttempts - List of attempts in the format of (attributes, identifierName, attemptResult, searchOnTable) where  
    ◦ attributes - Attributes of identifier entry used during identification  
    ◦ identifierName - CI identifier to which this identifier entry belongs  
    ◦ attemptResult - One of SKIPPED, NO_MATCH, MATCHED, MULTI_MATCH  
    ◦ searchOnTable - Table searched during the identification process.  
  The possible name-value pairs within the relations list are: |
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• className</td>
<td>Relationship CI's class name and is always cmdb_rel_ci</td>
</tr>
<tr>
<td>• operation</td>
<td>Type of operation: INSERT, UPDATE, NO_CHANGE</td>
</tr>
<tr>
<td>• sysId</td>
<td>Sys_id of the relationship CI inserted or updated</td>
</tr>
</tbody>
</table>

Identify an independent CI with lookup-based identification.

```javascript
var payload = {items: [
  {className:'cmdb_ci_netgear',
   values: {name:'ny8500-nbxs08',
            ports:'1200'},
   lookup: [{className:'cmdb_serial_number',
             values:{serial_number:'1234ABCD',
                     serial_number_type:'uuid',absent:'false',valid:'true'},
             {className:'cmdb_serial_number',
              values:{serial_number:'3456EFGH',
                     serial_number_type:'system',absent:'false',valid:'true'}}]},
  ];

var jsonUntil = new JSON();
var input = jsonUntil.encode(payload);
var output = SNC.IdentificationEngineScriptableApi.identifyCI(input);
gs.print(output);
```

Output:

```
{
  "items": [
    {
      "className": "cmdb_ci_netgear",
      "operation": "INSERT",
      "sysId": "787c7e6adb1910102f67dfe5e96196e",
      "relatedSysIds": [
        "f47c7e6adb1910102f67dfe5e96196e",
        "3c7c7e6adb1910102f67dfe5e961977"
      ],
      "relatedItems": [
        {
          "errors": [],
          "operation": "INSERT",
          "errorCount": 0,
          "mergedPayloadIds": []
        }
      ]
    }
  ]
}
```
"warningCount": 0,
"sysId": "f47c7e6adb1910102f67dfea5e961977",
"markers": [],
"inputIndices": [
{
   "mainIndex": 0,
   "subIndex": 0
}
],
"className": "cmdb_serial_number"
],
{
"errors": [],
"operation": "INSERT",
"errorCount": 0,
"mergedPayloadIds": [],
"warningCount": 0,
"sysId": "3c7c7e6adb1910102f67dfea5e961977",
"markers": [],
"inputIndices": [
{
   "mainIndex": 0,
   "subIndex": 1
}
],
"className": "cmdb_serial_number"
]
,"identifierEntrySysId": "Unknown",
"identificationAttempts": [
{
   "info": "sys_object_source SKIPPED",
   "identifierName": "",
   "attemptResult": "SKIPPED",
   "attributes": [],
   "hybridEntryCiAttributes": []
},
{
   "identifierName": "Hardware Rule",
   "attemptResult": "NO_MATCH",
   "attributes": [
      "serial_number",
      "serial_number_type"
   ]
}
"searchOnTable": "cmdb_serial_number",
"hybridEntryCiAttributes": []
},
{
"identifierName": "Hardware Rule",
"attemptResult": "SKIPPED",
"attributes": [
  "serial_number"
],
"searchOnTable": "cmdb_ci_hardware",
"hybridEntryCiAttributes": []
},
{
"identifierName": "Hardware Rule",
"attemptResult": "NO_MATCH",
"attributes": [
  "name"
],
"searchOnTable": "cmdb_ci_hardware",
"hybridEntryCiAttributes": []
},
{
"identifierName": "Hardware Rule",
"attemptResult": "SKIPPED",
"attributes": [
  "mac_address",
  "name"
],
"searchOnTable": "cmdb_ci_network_adapter",
"hybridEntryCiAttributes": []
}
],
"errorCount": 0,
"mergedPayloadIds": [],
"warningCount": 0,
"markers": [],
"inputIndices": [0]
},
"additionalCommittedItems": [],
"relations": []
"additionalCommittedRelations": []
}

Scoped equivalent

To use the `identifyCI(String jsonString)` method in a scoped application, use the corresponding scoped IdentificationEngine method: `IdentificationEngine - identifyCI(String jsonString)`.

`IdentificationEngineScriptableApi - identifyCIEnhanced(String source, String input, Object options)`

Determines the Configuration Management Database (CMDB) operation (insert/update) to perform with the specified payload (request body), without committing the operations in the database.

Use this method to simulate submission of a payload.

This method is similar to the `IdentificationEngineScriptableApi - identifyCI(String jsonString)` method, however it also supports the following functionality:

- Partial payloads
  - In case of an item having a warning or error, indicates if an item operation is `INSERT_AS_PARTIAL` or `INSERT_INCOMPLETE`.
  - Returns the sys_ids of partial payloads that were merged with existing partial payloads.
- Supports payload deduplicate feature.
- Generates a summary.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>String</td>
<td>Required. A JSON formatted string of configuration items to add or update.</td>
</tr>
<tr>
<td>input.items</td>
<td>Array</td>
<td>Array of objects that define the items to add or update.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.className</td>
<td>String</td>
<td>Required. Class/table name, sys_class_name, of the configuration item (CI) to create or update. This value can be any CMDB class/table, such as cmdb_ci_linux_server or cmdb_ci_win_server.</td>
</tr>
<tr>
<td>input.items.display_values</td>
<td>Object</td>
<td>Reference fields to create or update for this related item as name-value pairs, where the name is the field name and the value is the referenced display value. If you want to use the sys_id instead of the display value for reference fields, pass the information in the input.items.lookup.values parameter instead of in this parameter. Reference field names depend on the fields.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>--------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>input.items.internal_id</td>
<td>String</td>
<td>Unique item identifier for the associated payload. This can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td>input.items.lookup.className</td>
<td>String</td>
<td>Required. Class/table name, sys_class_name, of the configuration item (CI) to create or update. This value can be any CMDB class/table, such as cmdb_ci.</td>
</tr>
</tbody>
</table>

For example:

```json
"lookup": [
    {
        "className": "String",
        "internal_id": "String",
        "sys_object_source_info": {
            Object
        },
        "values": {
            Object
        }
    }
]
```

input.items.lookup.className

| String | Required. Class/table name, sys_class_name, of the configuration item (CI) to create or update. This value can be any CMDB class/table, such as cmdb_ci. |

For example:

```json
"lookup": [
    {
        "className": "String",
        "internal_id": "String",
        "sys_object_source_info": {
            Object
        },
        "values": {
            Object
        }
    }
]
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.lookup.internal_id</td>
<td>String</td>
<td>Unique lookup item identifier for the associated payload. This can be any value, but must be unique within the payload.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info</td>
<td>Object</td>
<td>Defines a unique CI identifier for a specific data source. Different sources may have different name-value pairs.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_feed</td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide the name of the feed sending this item. The data source generates this feed name. It can be any string that uniquely identifies the source feed.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_name</td>
<td>String</td>
<td>Data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.lookup.sys_object_source_info.source_native_key</td>
<td>String</td>
<td>Unique key-id for the item from the source. The data source generates this key. It can be any string that is unique to the item.</td>
</tr>
<tr>
<td>input.items.lookup.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date/time that the item was scanned. Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>input.items.lookup.values</td>
<td>Object</td>
<td>Field information for the CI as name-value pairs, where the name is the field name. When updating reference fields, the value must be the referenced sys_id. Field names and types depend on the fields selected, such as:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>input.items.related</td>
<td>Array</td>
<td>Reference to the top-level item that contains the related list. Rules in the Related Entry [cmdb_related_entry] define what type of records can be included, and rules in the items base tables define which external keys can be referenced.</td>
</tr>
</tbody>
</table>
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.related.className</td>
<td>String</td>
<td>Required. Class/table name, sys_class_name, of the configuration item (CI) to create or update. This value can be any CMDB class/table, such as cmdb_software_instance or cmdb_key_value.</td>
</tr>
<tr>
<td>input.items.related.internal_id</td>
<td>String</td>
<td>Unique identifier for this related item in this payload. You can use any value, but it must be unique within the payload.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info</td>
<td>Object</td>
<td>Object that makes up a unique CI identifier for a specified data source. Different sources may have different name-value pairs for the same CI.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info.source_feed</td>
<td>String</td>
<td>If the source can have multiple feeds, use this field to provide the name of the feed sending this item. The data source generates this feed name. It can be any string that uniquely identifies the source feed.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info.source_name</td>
<td>String</td>
<td>Identifies the data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info.source_native_key</td>
<td>String</td>
<td>Unique key/ID from the source for the related item. The data source generates this key. It can be any string that is unique to the item.</td>
</tr>
<tr>
<td>input.items.related.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date and time that the item was scanned. Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>input.items.related.values</td>
<td>Object</td>
<td>Fields to create or update for this related item as name/value pairs, where the name is the field name.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>field, the value must be the referenced sys_id. If you want to use the display value instead of the sys_id for reference fields, pass this information in a <code>display_values</code> object instead of in the <code>values</code> object. Field names and types depend on the fields selected as:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;values&quot;: { &quot;host_name&quot;: &quot;String&quot;, &quot;ip_address&quot;: &quot;String&quot;, &quot;name&quot;: &quot;String&quot;, &quot;os_name&quot;: &quot;String&quot;, &quot;sys_class_name&quot;: &quot;String&quot; }</td>
<td></td>
<td></td>
</tr>
<tr>
<td>input.items.settings</td>
<td>Object</td>
<td>Parameters that define the types of updates that are permitted.</td>
</tr>
<tr>
<td>&quot;settings&quot;: { &quot;skipReclassificationRestrictionRules&quot;: Boolean, &quot;updateWithoutDowngrade&quot;: Boolean, &quot;updateWithoutSwitch&quot;: Boolean, &quot;updateWithoutUpgrade&quot;: Boolean }</td>
<td></td>
<td></td>
</tr>
<tr>
<td>input.items.settings.skipReclassificationRestrictionRules</td>
<td>Boolean</td>
<td>Flag that indicates whether IRE should not run the Reclassification Restriction</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Parameters (continued)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Name</strong></td>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>input.items.settings.updateWithoutDowngrade</td>
<td>Boolean</td>
<td>Flag that indicates whether update and downgrade are both permitted for the item. Valid values: • true: Update the item but downgrade is not permitted. • false: Both item update and downgrade are permitted. Default: false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>input.items.settings.updateWithoutSwitch</td>
<td>Boolean</td>
<td>Flag that indicates whether the item can be updated and the class switched. Valid values: • true: Update the item but class switching is not permitted. • false: Both item update and class switching are permitted. Default: false</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>input.items.settings.updateWithoutUpgrade</td>
<td>Boolean</td>
<td>Flag that indicates whether update and upgrade are both permitted for this item. Valid values: • true: Update the item but upgrade is not permitted. • false: Both item update and upgrade are permitted. Default: false</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>input.items.sys_object_source_info</strong></td>
<td><strong>Object</strong></td>
<td>Unique CI identifier for a specific source.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>source_feed</strong>: String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the source can have multiple feeds, use this field to provide the name of the feed sending this item. The data source generates this feed name. It can be any string that uniquely identifies the source feed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>source_name</strong>: String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data source of the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>source_native_key</strong>: String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unique key/id for the item from the source.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>source_recency_timestamp</strong>: String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The recency timestamp of this source.</td>
</tr>
</tbody>
</table>

- true: Update the item but upgrade is not permitted.
- false: Both item update and upgrade are permitted.

Default: false
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.items.sys_object_source_info.source_recency_timestamp</td>
<td>String</td>
<td>UTC date and time that the item was scanned. Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
</tbody>
</table>
| input.items.values            | Object | Fields to create or update for this related item as name/value pairs, where the name is the field name. For reference fields, the value must be the referenced sys_id. If you want to use the display value instead of the sys_id for reference fields, pass this information in a `display_values` object instead of in the `values` object. Field names and types depend on the fields selected by the user, such as: |}

```json
  "values": {
    "host_name": "String",
    "ip_address": "String",
    "name": "String",
    "os_name": "String",
    "sys_class_name": "String"
  }
```
| input.referenceltems         | Array  | Array of objects that define references between items in the input payload. |
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>input.referenceltems.referenced</code></td>
<td>String</td>
<td>The internal_id defined for the item being referenced by another item.</td>
</tr>
<tr>
<td><code>input.referenceltems.referencedBy</code></td>
<td>String</td>
<td>The internal_id defined for the item that references another item.</td>
</tr>
<tr>
<td><code>input.referenceltems.referenceField</code></td>
<td>String</td>
<td>Name of the reference field in the class/table for the referencedBy item.</td>
</tr>
<tr>
<td><code>input.relations</code></td>
<td>Array</td>
<td>Array of objects that specify relationships between items in the input payload. An object in this array can use either:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- The object can define a relationship between two top-level items using <code>parent</code> and <code>child</code> name-value pairs, with values representing item indexes.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.relations.child</td>
<td>Number</td>
<td>Integer index of the CI object in the items array that represents the child in the relationship (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td>input.relations.child_id</td>
<td>String</td>
<td>The internal_id of the child item in the relationship (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td>input.relations.parent</td>
<td>Number</td>
<td>Integer index of the parent item in the items array (items, items.related, or items.lookup).</td>
</tr>
</tbody>
</table>

- The object can define a relationship between any two items, using top-level, related, and lookup items, with parent_id and child_id key/value pairs representing the values for these items.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input.relations.parent_id</td>
<td>String</td>
<td>The <strong>internal_id</strong> of the parent item in the relation (items, items.related, or items.lookup).</td>
</tr>
<tr>
<td>input.relations.sys_rel_source_info</td>
<td>Object</td>
<td>Discovery source information for the relationship. For non-dependency relationships, this information is saved in the Relationship Sources [sys_rel_source] table (not persisted for identifyCIEnhanced() or identifyCI() methods.).</td>
</tr>
<tr>
<td>input.relations.sys_rel_source_info.source_name</td>
<td>String</td>
<td>Discovery source name. Default: Discovery source passed in the API method parameter.</td>
</tr>
<tr>
<td>input.relations.sys_rel_source_info.source_feed</td>
<td>String</td>
<td>Any string that is a sub-discovery/scan within the discovery source. Default: 'UNKNOWN' is stored in the source_feed column when creating a record in sys_rel_source table.</td>
</tr>
<tr>
<td>input.relations.type</td>
<td>String</td>
<td>Type of relationship that exists between the parent and child items. Must be a name field value from...</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>the CI Relationship Type</td>
<td></td>
<td>[cmdb_rel_type] table.</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional, but {} or null must be passed in. Optional object options to enable or disable features.</td>
</tr>
<tr>
<td>options.deduplicate_payloads</td>
<td>Boolean</td>
<td>Flag that indicates whether duplicate items are merged or considered errors.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Merge duplicate items.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Consider duplicate items as errors.</td>
</tr>
</tbody>
</table>

**Note:**
- If `partial_commits` is set to true, both `partial_commits` and `deduplicate_payloads` are enabled, even if they are set to false, as those features are essential for partial payloads functionality.

```json
"options": {
  "deduplicate_payloads": "Boolean",
  "generate_summary": "Boolean",
  "partial_commits": "Boolean",
  "partial_payloads": "Boolean"
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options.generate_summary</td>
<td>Boolean</td>
<td>Flag that indicates whether the returned results contain summary information. For the details of the returned summary, see <code>&lt;String&gt;</code> in the return results. Valid values: • true: Include summary information. • false: Do not include summary information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>options.partial_commits</td>
<td>Boolean</td>
<td>Flag that indicates whether partial commit support is enabled. For additional information on partial commits, see Enhanced IRE features. Valid values: • true: Partial commit enabled. • false: Partial commit disabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td>options.partial_payloads</td>
<td>Boolean</td>
<td>Flag that indicates whether partial payload support is enabled. For additional information on partial payloads, see Enhanced IRE features and Create an IRE data source rule.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: true</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Valid values:
- true: Partial payload enabled.
- false: Partial payload disabled.

Default: true

source

String

Data source of the CI information. This value must be one of the values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;</td>
<td>JSON formatted string that is a list of results for the configuration items in the input string. Data type: String</td>
</tr>
</tbody>
</table>

```
{
  "additionalCommittedItems": [Array],
  "additionalCommittedRelations": [Array],
  "hasError": "Boolean",
  "hasWarning": "Boolean",
  "items": [Array],
  "relations": [Array],
  "summary": [Object]
}
```

| <String>.additionalCommittedItems | Committed items were committed during the IRE processing of the current payload, but were not present in the current input payload. Data type: Array |
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
"additionalCommittedItems": [
{
  "className": "String",
  "errorCode": Number,
  "operation": "String",
  "identificationAttempts": [Array],
  "inputIndices": [Array],
  "markers": [Array],
  "mergedPayloads": [Array],
  "sysId": "String"
}
]
```

**<String>.additionalCommittedRelations**

Array of objects that describe a dependent relationship CI that was not included in the request body relations list to insert or update.

Data type: Array

```
"additionalCommittedRelations": [
{
  "className": "String",
  "inputIndices": [Array],
  "markers": [Array],
  "mergedPayloadIds": [Array],
  "operation": "String"
}
]
```

**<String>.hasError**

Flag that indicates whether any item or relation has errors.

Data type: Boolean

**<String>.hasWarning**

Flag that indicates whether any item or relation has warnings.

Data type: Boolean

**<String>.items**

List of objects that describe the created or updated CIs.

Data type: Array

```
"items": [
{
  "additionalRelatedItems": [Array],
```
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```json
"className": "String",
"duplicateIndices": [Array],
"errorCount": Number,
"errors": [Array],
"identificationAttempts": [Array],
"identifierEntrySysId": "String",
"info": [Array],
"inputIndices": [Array],
"maskedAttributes": [Array],
"operation": "String",
"relatedSysIds": [Array],
"sysId": "String"
```

### `<String>.items.additionalRelatedItems`

List of JSON objects that provide information about additional lookup and related items that were processed but not provided as part of the input payload. These items are from partial payloads.

Data type: Array

```json
"additionalRelatedItems": [ 
  
  
  "className": "String",
  "inputIndices": [Array],
  "mergedPayloadIds": [Array],
  "operation": "String",
  "sysId": "String"

  
]`
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.additionalRelatedItems.mergedPayloadIds</td>
<td>List of sys_ids of the partial payloads that were merged into the related item. Located in the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table. Data type: Array</td>
</tr>
</tbody>
</table>
| <String>.items.additionalRelatedItems.operation | Possible values:  
- INSERT: New CI was inserted into the database.  
- NO_CHANGE: No CI changes were made.  
- UPDATE: Existing CI was updated.  
Data type: String |
| <String>.items.additionalRelatedItems.sysId | Sys_id of the CI that was updated or created. Data type: String |
| <String>.items.className | Class/table name (sys_class_name) of the CI that was created or updated. Data type: String |
| <String>.items.duplicateIndices | List of indexes of CIs that are duplicates of the current item. Data type: Array |
| <String>.items.errorCount | Number of errors encountered while processing the item. Data type: Number |
| <String>.items.errors | Array of objects in which each object describes an error encountered while processing this CI. Data type: Array |

```
"errors": [
  {
    "error": "String",
    "message": "String"
  }
]
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>&lt;String&gt;.items.errors.message</td>
<td>The error message encountered while processing the CI. Data type: String</td>
</tr>
<tr>
<td>&lt;String&gt;.items.identificationAttempts</td>
<td>List of attempts that were made to identify the CIs. Data type: Array</td>
</tr>
</tbody>
</table>

```
"identificationAttempts": [
  {
    "attemptResult": "String",
    "attributes": [Array],
    "hybridEntryCiAttributes": [Array],
    "identifierName": "String",
    "searchOnTable": "String"
  }
]
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.attemptResult</td>
<td>Results of the attempt to identify the CI. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• MATCHED: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• MULTI_MATCH: Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• NO_MATCH: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• SKIPPED: Identification not attempted. The attributes required for this identifier rule table search were not provided, so the rule was not applied.</td>
</tr>
</tbody>
</table>

Data type: String |

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.items.identificationAttempts.attributes</td>
<td>List of CI identifier entry attributes that were used during the identification process. Data type: Array</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Attribute names and types depend on the request body data and the identifier in use, such as:</td>
</tr>
</tbody>
</table>
|      | ```
|      | "attributes": {
|      | "serial_number": "String",
|      | "serial_number_type": "String",
|      | }
|      | ``` |
|      | Identifier rule used for this CI identification attempt. |
|      | Data type: String |
|      | Name of the table searched during the identification process. |
|      | Data type: String |
|      | Sys_id for the identifier rule used to identify the CI. Located in the Identifier Entry [cmdb_identifier_entry] table. |
|      | Data type: String |
|      | List of objects that contains additional information about the processing of the item. |
|      | Data type: Array |
|      | ```
|      | "info": [
|      | |
|      | "code": "String",
|      | "message": "String",
|      | "ruleSysId": "String"
|      | ]
|      | ``` |
|      | Reclassification type that was skipped. |
|      | Possible values: |
|      | • SKIPPED_CLASS_SWITCH |
|      | • SKIPPED_CLASS_DOWNGRADE |
|      | • SKIPPED_CLASS_UPGRADE |
|      | Data type: String |
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;String&gt;.items.info.message</code></td>
<td>Message that provides additional insights into the reason for skipping the reclassification. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.info.ruleSysId</code></td>
<td>Sys_id of the reclassification restriction rule that was matched. Applicable only when IRE skips reclassification due to reclassification restriction rule. This value is empty if the reclassification is skipped due to a payload or global flag. Data type: String</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.inputIndices</code></td>
<td>Indexes of the corresponding input CI. For top-level items, it is a list of integers. For related or lookup CIs, it is list of JSON objects. Data type: Array of Numbers</td>
</tr>
<tr>
<td><code>&lt;String&gt;.items.maskedAttributes</code></td>
<td>List of attributes whose update by a non-authoritative data source was skipped as defined by the Reconciliation Rules. Data type: Array</td>
</tr>
</tbody>
</table>
| `<String>.items.operation`     | Operation that took place. Possible values:  
  - INSERT: New CI was inserted into the database.  
  - INSERT_AS_INCOMPLETE: Item was saved in cmdb_ire_incomplete_payloads table.  
  - INSERT_AS_PARTIAL: Item was saved in cmdb_ire_partial_payloads table.  
  - NO_CHANGE: No CI changes were made.  
  - UPDATE: Existing CI was updated.  
  - UPDATE_WITH_DOWNGRADE: CI was updated and the class changed to a more generic class (ancestor class).  |
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| • UPDATE_WITH_SWITCH: CI was updated and the class changed to a class that is not ancestor or descendent.  
• UPDATE_WITH_UPGRADE: CI was updated and the class changed to a more specialized class (descendent class). |  

Data type: String

<String>.items.relatedItems

JSON objects that provide information about the processed related items.

Data type: Array

```
"relatedItems": [
  {
    "className": "String",
    "errorCount": Number,
    "errors": [Array],
    "inputIndices": [Array],
    "mergePayloadIds": [Array],
    "operation": "String",
    "sysId": "String",
    "warningCount": Number,
    "warnings": [Array]
  }
]
```

<String>.items.relatedItems.className

Class/table name (sys_class_name) of the related item.

Data type: String

<String>.items.relatedItems.errorCount

Number of errors detected while processing the related items.

Data type: Number

<String>.items.relatedItems.errors

List of errors that occurred during processing of the related item.

Data type: Array

```
"errors": [
  {
    "error": "String",
  }
]
```
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| | "message": "String"
| | }
| | ]

### <String>.items.relatedErrors

Type of error encountered while processing the related item. Data type: String

### <String>.items.relatedErrors.message

The error message encountered while processing the related item. Data type: String

### <String>.items.relatedErrors.inputIndices

Index of the corresponding input item. For top-level items, it is a list of integers. For related or lookup items, it is list of JSON objects. Data type: Array of Numbers or Array of Objects

```
"inputIndices": [
  {
    "mainIndex": Number,
    "subIndex": Number
  }
]
```

### <String>.items.relatedErrors.mainIndex

Index value from the request body items array that corresponds to the CI parent of the related item. Data type: Number

### <String>.items.relatedErrors.subIndex

Index value from the request body items.lookup array that corresponds to the related item. Data type: Number

### <String>.items.relatedErrors.mergedPayloadIds

List of sys_ids of the partial payloads that were merged into the CI during processing. Data type: Array

### <String>.items.relatedErrors.operation

Operation that took place. Possible values:
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• INSERT: New related CI was inserted into the database.</td>
<td></td>
</tr>
<tr>
<td>• INSERT_AS_INCOMPLETE: Item was saved in cmdb_ire_incomplete_payloads table.</td>
<td></td>
</tr>
<tr>
<td>• INSERT_AS_PARTIAL: Item was saved in cmdb_ire_partial_payloads table.</td>
<td></td>
</tr>
<tr>
<td>• NO_CHANGE: No related CI changes were made.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE: Existing related CI was updated.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE_WITH_DOWNGRADE: Related CI was updated and the class changed to a more generic class (ancestor class).</td>
<td></td>
</tr>
<tr>
<td>• UPDATE_WITH_SWITCH: Related CI was updated and the class changed to a class that is not ancestor or descendent.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE_WITH_UPGRADE: Related CI was updated and the class changed to a more specialized class (descendent class).</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

| <String>.items.relatedItems.sysId | The sys_id of the related item. |
| Data type: String |

| <String>.items.relatedItems.warningsCount | Number of warnings encountered when processing the related items. |
| Data type: Number |

| <String>.items.relatedItems.warnings | Array of objects that describes a warning encountered while processing the related items. |
| Data type: Array |

```json
"warnings": [
  {
    "error": "String",
    "message": "String"
  }
]
```
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| `<String>.items.relatedSysIds` | List of the sys_id values of the CIs used during lookup-based identification of related items.  
Data type: String |
| `<String>.items.sys_id` | Sys_id of the CI that was updated or created.  
Data type: String |
| `<String>.relations` | List of JSON objects that provides information about the processed relations.  
Data type: Array |
|    | ![](image) |
| `<String>.relations.className` | Sys_class_name of this dependent relationship CI.  
Only supported value:  
• `cmdb_rel_ci`: CI Relationship table.  
Data type: String |
| `<String>.relations.errorCount` | Number of errors encountered when processing the dependent relationship CI.  
Data type: Number |
| `<String>.relations.errors` | Array of objects that describe errors that were encountered while processing this dependent relationship CI.  
Data type: Array |
<p>|    | <img src="image" alt="" /> |</p>
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;.relations.inputIndices</td>
<td>Indexes for the dependent relationship CI objects in the request body relations array that correspond to this dependent relationship CI. Data type: Array</td>
</tr>
</tbody>
</table>
| <String>.relations.operation | Type of operation performed. Possible values:  
  - INSERT: The dependent relationship CI was inserted into the target table as a new record.  
  - INSERT_AS_INCOMPLETE: The dependent relationship CI had errors and was inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.  
  - INSERT_AS_PARTIAL: The dependent relationship CI had errors and was inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.  
  - NO_CHANGE: No operation was performed.  
  - UPDATE: An existing dependent relationship CI in the target table was updated. Data type: String |
| <String>.relations.sysId   | Sys_id of the dependent relationship CI. Data type: String                                                                                                                                                  |
| <String>.summary           | List of JSON properties that provide statistics on how many items were inserted, updated, and such, per class. Data type: Array                                                                                 |
| <String>.summary.<class_name> | Statistics for a specific class. Data type: Object                                                                                                                                                        |
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;errorCount&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;incompleteItemCount&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;insertedItemCount&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;partialItemCount&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;skippedItemCount&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;unchangedItemCount&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;updatedItemCount&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;warningCount&quot;: Number</td>
</tr>
</tbody>
</table>

- `<String>.summary.<class_name>.additionalInsertedItemCount`: Number of items inserted due to processing of partial payloads.
  - Data type: Number
- `<String>.summary.<class_name>.errorCount`: Number of errors encountered when processing items.
  - Data type: Number
- `<String>.summary.<class_name>.incompleteItemCount`: Number of items inserted in the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.
  - Data type: Number
- `<String>.summary.<class_name>.insertedItemCount`: Number of items created.
  - Data type: Number
- `<String>.summary.<class_name>.partialItemCount`: Number of items saved in the Partial Payload table [cmdb_ire_partial_payloads].
  - Data type: Number
- `<String>.summary.<class_name>.skippedItemCount`: Number of items that were skipped.
  - Data type: Number
- `<String>.summary.<class_name>.unchangedItemCount`: Number of items that had entries but were not modified.
  - Data type: Number
- `<String>.summary.<class_name>.updatedItemCount`: Number of items updated.
  - Data type: Number
- `<String>.summary.<class_name>.warningCount`: Number of items that generated a warning when processed.
  - Data type: Number
```javascript
var payload = {
  "items": [
    {
      "className": "cmdb_ci_win_server",
      "values": {
        "chassis_type": "Desktop",
        "os": "Windows 2012 R2 Datacenter",
        "name": "Windows2012Server1",
        "serial_number": "0000-0011-1690-8730-8636-5722-52",
        "cpu_count": "1"
      },
      "lookup": [
        {
          "values": {
            "valid": "true",
            "serial_number": "0000-0011-1690-8730-8636-5722-52",
            "serial_number_type": "bios"
          },
          "className": "cmdb_serial_number"
        },
        {
          "values": {
            "valid": "true",
            "serial_number": "3311-9736-4988-9744-1749-4183-41",
            "serial_number_type": "chassis"
          },
          "className": "cmdb_serial_number"
        }
      ],
      "internal_id": "16777219",
      "sys_object_source_info": {
        "source_feed": "SN Discovery Feed 1",
        "source_name": "ServiceNow",
        "source_native_key": "16777219",
        "source_recency_timestamp": "2019-10-18 08:31:23"
      }
    },
    {
      "className": "cmdb_ci_spkg",
      "values": {
        "name": "Windows 2012 R2 Datacenter",
        "key": "Microsoft Windows Server 2012 R2 Datacenter:::NULL"
      }
    }
  ]
};
```
"related": [  
  {  
    "internal_id": "Microsoft Windows Server 2012 R2 Datacenter::_:::_NULL|16777219",
    "values": {  
      "name": "Windows 2012 R2 Datacenter-SAMLABVM52"
    },  
    "className": "cmdb_software_instance",
    "sys_object_source_info": {  
      "source_feed": "SN Discovery Feed 1",
      "source_name": "ServiceNow",
      "source_native_key": "Microsoft Windows Server 2012 R2 Datacenter::_:::_NULL|16777219"
    }
  },  
  {  
    "internal_id": "Microsoft Windows Server 2012 R2 Datacenter::_:::_NULL|16777229",
    "values": {},  
    "className": "cmdb_software_instance",
    "sys_object_source_info": {  
      "source_feed": "SN Discovery Feed 1",
      "source_name": "ServiceNow",
      "source_native_key": "Microsoft Windows Server 2012 R2 Datacenter::_:::_NULL|16777229"
    }
  }
],  
"settings": {  
  "skipReclassificationRestrictionRules": "true",
  "updateWithoutDowngrade": "true",
  "updateWithoutUpgrade": "true",
  "updateWithoutSwitch": "true"
},  
{  
  "className": "cmdb_ci_app_server_tomcat",
  "values": {  
    "running_process_key_parameters": "/opt/OV/nonOV/tomcat/b/temp org.apache.catalina.startup.Bootstrap start",
    "install_directory": "/opt/OV/nonOV/tomcat/b",
    "name": "Tomcat@hpom9:3443",
    "server_port": "8006",
    "sys_class_name": "cmdb_ci_app_server_tomcat"
  }
},  
{  
    "internal_id": "tomcat_id"
}]}
"relations": [
{
    "parent_id": "tomcat_id",
    "child_id": "16777219",
    "type": "Runs on::Runs"
}],

"referenceItems": [
{
    "referenceField": "installed_on",
    "referenced": "16777219",
    "referencedBy": "Microsoft Windows Server 2012 R2 Datacenter::_::_NULL|16777219"
}]},

var input = new JSON().encode(payload);
var output = SNC.IdentificationEngineScriptableApi.identifyCIEnhanced('ServiceNow', input, {});

Output:

{
  "items": [
  {
    "className": "cmdb_ci_win_server",
    "operation": "INSERT",
    "relatedSysIds": [
      null,
      null
    ],
    "relatedItems": [
      {
        "errors": [],
        "operation": "INSERT",
        "className": "cmdb_serial_number",
        "errorCount": 0,
        "markers": [],
        "inputIndices": [
          {
            "mainIndex": 0,
            "subIndex": 0
          }
        ],
        "mergedPayloadIds": [],
        "warningCount": 0
      }
    ]
  }
]
"errors": [],
"operation": "INSERT",
"className": "cmdb_serial_number",
"errorCount": 0,
"markers": [],
"inputIndices": [
  {
    "mainIndex": 0,
    "subIndex": 1
  }
],
"mergedPayloadIds": [],
"warningCount": 0
],
"identificationAttempts": [

  {
    "attributes": [],
    "info": "sys_object_source NO_MATCH",
    "identifierName": "",
    "attemptResult": "NO_MATCH",
    "hybridEntryCiAttributes": []
  },

  {
    "attributes": [
      "serial_number",
      "serial_number_type"
    ],
    "identifierName": "Hardware Rule",
    "attemptResult": "NO_MATCH",
    "searchOnTable": "cmdb_serial_number",
    "hybridEntryCiAttributes": []
  },

  {
    "attributes": ["serial_number"
    ],
    "identifierName": "Hardware Rule",
    "attemptResult": "NO_MATCH",
    "searchOnTable": "cmdb_ci_hardware",
    "hybridEntryCiAttributes": []
  }
]
"subIndex": 0,
},
"mergedPayloadIds": [],
"warningCount": 0
},
{
"errors": [],
"warnings": [
{
"error": "MISSING_MATCHING_ATTRIBUTES",
"message": "In payload missing minimum set of input values for criterion (matching) attributes from identify rule for table [cmdb_software_instance]. Add these input values in payload item '{"className":"cmdb_software_instance","values":{},"internal_id":"f7273cccec3010105cd4bb4e6eb4db5d","sys_object_source_info":{"source_feed":"SN Discovery Feed 1","source_name":"ServiceNow","source_native_key":"Microsoft Windows Server 2012 R2 Datacenter_:::_NULL|16777229","settings":{},"sys_ire_info":{"ire_received_time":"2020-05-10 17:57:48"}}'",
"operation": "INSERT_AS_PARTIAL",
"className": "cmdb_software_instance",
"errorCount": 0,
"sysId": "Unknown",
"markers": [],
"inputIndices": [
{
"mainIndex": 1,
"subIndex": 1
}
],
"mergedPayloadIds": [],
"warningCount": 1
},
"identificationAttempts": [
{
"attributes": [],
"info": "sys_object_source SKIPPED",
"identifierName": "",
"attemptResult": "SKIPPED",
"hybridEntryCiAttributes": []
{  
  "attributes": [  
    {  
      "key":  
      }  
  ],  
  "identifierName": "Software",  
  "attemptResult": "NO_MATCH",  
  "searchOnTable": "cmdb_ci_spkg",  
  "hybridEntryCiAttributes": []  
},  
  "errorCount": 0,  
  "markers": [],  
  "inputIndices": [  
    1  
  ],  
  "mergedPayloadIds": [],  
  "warningCount": 0  
},  
  {  
    "className": "cmdb_ci_app_server_tomcat",  
    "operation": "INSERT",  
    "identificationAttempts": [  
      {  
        "attributes": [],  
        "info": "sys_object_source SKIPPED",  
        "identifierName": ",",  
        "attemptResult": "SKIPPED",  
        "hybridEntryCiAttributes": []  
      }  
    ],  
    "errorCount": 0,  
    "markers": [],  
    "inputIndices": [  
      2  
    ],  
    "mergedPayloadIds": [],  
    "warningCount": 0  
  }  
]  
"additionalCommittedItems": [],  
"relations": [  
  {  
    "className": "cmdb_rel_ci",  


IdentificationEngineScriptableApi - runIdentificationAudit(GlideRecord now_GR)

Runs an identification audit against the specified configuration item (CI) to detect duplicates.

If duplicates are found, duplication tasks are created. Only use this method on CI types with independent identification rules.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>CI on which to run the audit to detect duplicates. The CI must have independent identification rules.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Scoped equivalent

To use the `runIdentificationAudit(GlideRecord now_GR)` method in a scoped application, use the corresponding scoped IdentificationEngine method:

IdentificationEngine - runIdentificationAudit(GlideRecord now_GR).

IDResult - Global

A container class for the result of a CI Identifier.

Use with any server-side discovery script.
IDResult - IDResult(Boolean explore, String sysID)

Creates an instance of the IDResult class.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>explore</td>
<td>Boolean</td>
<td>Set to true if the CI should be explored; otherwise, false.</td>
</tr>
<tr>
<td>sysID</td>
<td>String</td>
<td>An existing CI's sys_id, or null if no existing CI could be found.</td>
</tr>
</tbody>
</table>

IEventSender - Global

The IEventSender API provides methods to send events from a MID Server to a ServiceNow instance and to query the status of the event queue.

Before calling any methods within this class, you must call the SNEventSenderProvider.getEventSender() method to instantiate an IEvent object.

Use the Event API to add/update fields within an event. Use the SNEventSenderProvider API to instantiate an event sender object. Then use the IEventSender API to send the event to a ServiceNow instance.

You must activate the Event Management (com.glideapp.itom.snac) plugin before attempting to access this API. The Event Management plugin requires a separate subscription and must be activated by ServiceNow personnel. This plugin includes demo data and activates related plugins if they are not already active.

For additional information on event management, see Event Management.

IEventSender - getQueueSize()

Returns the size of the event queue.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Size of the event queue.</td>
</tr>
<tr>
<td></td>
<td>If an event queue is not currently implemented, returns &quot;0&quot;.</td>
</tr>
</tbody>
</table>
var event = new Event();
event.setField("source", "SAP Solman");
event.setField("severity", "3");
event.setField("time_of_event", "2019-18-05 13:12:05");
event.setField("type", "SAP object");
// parameter added to additional info section
event.setField("user", "admin");
var eventSender = SNEventSenderProvider.getEventSender();
var sendStatus = eventSender.sendEvent(event); // send event
var queueSize = eventSender.getQueueSize(); // check the size of the event queue

IEventSender - isFullQueue()
Checks if the event queue has exceeded its maximum size.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates whether the event queue has exceeded its maximum size. When this happens, no additional event messages can be queued until the queue decreases in size. Valid values:  
- true: Event queue is full.  
- false: Event queue is not full and can add more event entries. |

var eventSender = SNEventSenderProvider.getEventSender();
if (!eventSender.isFullQueue()) // If queue is not full, send event
    eventSender.sendEvent(event);

IEventSender - sendEvent(Object event)
Places the specified event in the event queue.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
<td>IEventSender</td>
<td>Event object to place in the events queue.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the event was successfully queued</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Event was queued</td>
</tr>
<tr>
<td></td>
<td>• false: Error</td>
</tr>
</tbody>
</table>

```javascript
var eventSender = SNEventSenderProvider.getEventSender();
eventSender.sendEvent(event); // send event
```

## Image - Scoped, Global

Creates an Image object representing an image and its layout insert in a PDF. Enables defining attributes such as scale, alignment, and border color.

This API is part of the ServiceNow PDF Generation Utilities plugin (com.snc.apppdfgenerator) and is provided within the sn_pdfgeneratorutils namespace. The plugin is activated by default.

This API is a component used with the Document API to generate a PDF.

You can add an image to a PDF using one of the following methods:

- **Cell** – `addImage(Image image)` – Adds an image to a table cell
- **Table** – `addImageCell(Image image)` – Adds a cell that contains an image to a table.
- **Document** – `addImage(Image image)` - Adds an image to a page

## Image - Image(String attachmentSysId)

Instantiates a new Image object. Used to verify if an image attachment exists and is available for modification.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attachmentSysId</td>
<td>String</td>
<td>Sys_id of an image in the Attachments [sys_attachment] table.</td>
</tr>
</tbody>
</table>

The following example shows how to create a `Image` object.

```javascript
var image = new sn_pdfgeneratorutils.Image("<sys_id>");
```

**Image – scaleAbsolute(Number width, Number height)**

Scales an image to absolute width and height sizes. This setting does not preserve the width-height ratio of the image and might result in undesired stretching if settings are not precise.

To scale to an absolute size that preserves width-height ratio of an image, use the `scaleToFit()` method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Image width in points.</td>
</tr>
<tr>
<td>height</td>
<td>Number</td>
<td>Image height in points.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add an image to a PDF with absolute width and height settings.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var scaleAbsPic = new sn_pdfgeneratorutils.Image("<img_sys_id>");
scaleAbsPic.scaleAbsolute(25,50);
document.addImage(scaleAbsPic);
document.saveAsAttachment("incident", "<record_sys_id>", "docWithImg.pdf");
```
**Image – scaleToFit(Number width, Number height)**

Scales an image to an absolute size while preserving the width-height ratio.

Resulting output varies by image aspect ratio. If the width and height parameter values do not match the image aspect ratio, one value renders smaller in output than the value given.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Maximum image width in points.</td>
</tr>
<tr>
<td>height</td>
<td>Number</td>
<td>Maximum image height in points.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to insert an image scaled to fit using the `Cell.addImage()` method.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// add a table
var table = new sn_pdfgeneratorutils.Table([1,3],false);

// text for the left column
var text = "sample image";

// add a table cell for the image in the right column
var imgCell = new sn_pdfgeneratorutils.Cell(1, 1);

// add an image and set it scale-to-fit
var scaleToFitPic = new sn_pdfgeneratorutils.Image("<img_sys_id>");
scaleToFitPic.scaleToFit(90,175);

// add the image to the cell
imgCell.addImage(scaleToFitPic);

// add the right cell
table.addTextCell(text);

table.addCell(imgCell);
```
Image – setAutoScale(Boolean value)

Enables scaling width and height to a page or cell while retaining dimensions.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| value  | Boolean    | Flag that indicates whether to automatically scale an image. Valid values:  
|        |            | • true: Automatically scales the image  
|        |            | • false: Image does not scale  
|        |            | Default: false |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add an image to a PDF with automatic scaling. The image is added using the Table – addImageCell() method.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// Add a table
var table = new sn_pdfgeneratorutils.Table([1,2],false);

// Text in left column
var text = "sample image";
```
// Image in right column
var autoScaledPic = new sn_pdfgeneratorutils.Image("<image_sys_id>");
autoScaledPic.setAutoScale(true);

table.addTextCell(text);
table.addImageCell(autoScaledPic);

document.addTable(table);
document.saveAsAttachment("incident", ":<record_sys_id>", "imgAutoScale.pdf");

Image – setColoredBorder(Color color, Number width)
Sets a border on a PDF in the specified color.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>Color</td>
<td>Image border color.</td>
</tr>
<tr>
<td>width</td>
<td>Number</td>
<td>Width of the border in points.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a five-point red-colored border on an image.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var borderColor = new sn_pdfgeneratorutils.Color([1.0,0.0,0.0]);

// declare image using sys attachment
var image = new sn_pdfgeneratorutils.Image("<imgAttachment_sys_id>");

image.setColoredBorder(borderColor, 5);
document.addImage(image);
document.saveAsAttachment("incident", ":<record_sys_id>", "docWithBorderedImage.pdf");
```
Image – setHorizontalAlignment(String alignment)
Sets the horizontal alignment of the image.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alignment</td>
<td>String</td>
<td>Positions image alignment on a page or block element. Valid values: • Center • Left • Right Default: Left</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add a centered image on a PDF page.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare image using sys attachment
var image = new sn_pdfgeneratorutils.Image("<imgAttachment_sys_id>");

String alignment = "Center";
image.setHorizontalAlignment(alignment);
document.addImage(image);

// save pdf as attachment to target record in the Incident table
document.saveAsAttachment("incident", ",<sys_id>, "docWithImageCentered.pdf");
```

Image – setNoBorder()
Sets an image to have no border.
The following example shows how to add an image to a document without a border.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare image using sys attachment
var image = new sn_pdfgeneratorutils.Image("<imgAttachment_sys_id>");
image.setNoBorder();
document.addImage(image);
document.saveAsAttachment("incident", "<sys_id>", "docWithImgNoBorder.pdf");
```

### Interaction - Scoped

Use the Interaction API to define the behavior for interaction records.

⚠️ **Note:** This method has been deprecated.

This class requires the Interaction Logging, Routing, and Queueing plugin (com.glide.interaction).

To use this class in a scoped API, use the `sn_interaction` namespace identifier.

**Scoped Interaction - create(Object options)**

Create an interaction.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Field values for an interaction record. The channel field with the channel sys_id and channel metadata are required. If a queue is not included in the parameter, the system returns Interaction b2c0a3af202a1300964f959e0488de75 has no queue specified... running queue matching rules.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Interaction</td>
</tr>
</tbody>
</table>

```javascript
var interactionObj = sn_interaction.Interaction.create(
  {
    channel: '28a59893873103002ae97e2526cb0b5d',
    channel_metadata: {},
    opened_for: '46d44a23a9fe19810012d100cca80666',
    queue: 'f3a50867b3030302186a72256a8dcb7'
  });
```

### Scoped Interaction - getInteraction(GlideRecord interaction)

Get an interaction record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interaction</td>
<td>GlideRecord</td>
<td>Interaction record from the interaction table [interaction] that is retrieved from the system.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Interaction</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('interaction');
now_GR.get('608a21bd096a9300964ffbd57ba7dd8d');
var interaction = sn_interaction.Interaction.getInteraction(now_GR);
```
Scoped Interaction - transferToAgent(String sysID)

Transfer an interaction record to an agent using the sys_id for the agent.

When an interaction is transferred from one agent to another, the interaction needs to be accepted or rejected using GlideRecord APIs. Use the GlideRecord API to change the state of the interaction and update. For more information, see the GlideRecord - update method.

ℹ️ Note: This method has been deprecated.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>SysID</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('interaction');
GR.get('24b927ef202a1300964f959e0488de2d');
var interaction = sn_interaction.Interaction.getInteraction(GR);
interaction.transferToAgent('6816f79cc0a8016401c5a33be04be441');
```

Scoped Interaction - transferToQueue(String sysID)

Transfer an interaction record to an interaction queue.

Transferring an interaction from one queue to another closes the original interaction and creates an interaction in the new queue.

ℹ️ Note: This method has been deprecated.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>SysID</td>
</tr>
</tbody>
</table>
### Scoped Interaction - accept()

Accept a new interaction.

Accepting a transfer only works for pending transfers. Find any pending transfers in the Interaction Agent Transfer [interaction_agent_transfer] table.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the transfer is accepted.</td>
</tr>
</tbody>
</table>

```javascript
var now_GR = new GlideRecord('interaction');
GR.get('24b927ef202a1300964f959e0488de2d');
var interaction = sn_interaction.Interaction.getInteraction(GR);
interaction.transferToQueue('86fee1933b101300088d832b44efc474');
```

### IPAddressFixup - Global

After a device has been successfully discovered, ensures that no other device has the same IP address. If any duplicates are found, the IP address field is cleared.

Use with any server-side Discovery script to validate IP addresses.

#### IPAddressFixup - dedupe(String tableName, String ip)

Removes duplicates of the specified IP address in the specified table.

```javascript
var now_GR = new GlideRecord('interaction');
GR.get('b2c0a3af202a1300964f959e0488de75');
var interaction = sn_interaction.Interaction.getInteraction(GR).accept();
```
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>The table to check for duplicates</td>
</tr>
<tr>
<td>ip</td>
<td>String</td>
<td>The IP address to check for</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### IPAddressFixup - dedupeAll()

Removes all duplicate IP addresses from the tables.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### IPAddressFixup - fix()

Removes all duplicate IP addresses and ensures that the parent ip_address record is set to one of the NIC's IP addresses.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**IPAddressFixup - getParentIP()**

Returns the parent IP address for the current device.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**IPAddressFixup - syncIP()**

Ensures that the parent ip_address record is set to one of the NIC's IP addresses, or leaves it alone if there were no NICs.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**IPAddressFixup - setParentIP(String ip)**

Sets the IP address field for the current CI.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip</td>
<td>String</td>
<td>The IP address for the current CI.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**IPService - Global**

Encapsulates an IP Service.

Use this class during the discovery scanning phase.

**IPService - creates**

The table where this service creates entries.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>creates</td>
<td>String</td>
<td>The table where this service creates entries.</td>
</tr>
</tbody>
</table>

**IPService - description**

Description of the IPService.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>String</td>
<td>Description of the IPService.</td>
</tr>
</tbody>
</table>

**IPService - getFromArrayList(Array list)**

Returns an array of IPService instances specified by a Java ArrayList of sys_ids.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>list</td>
<td>Array</td>
<td>List of sys_ids.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>IPService instances</td>
</tr>
</tbody>
</table>

### IPService - IPService(Object source)

Creates an instance of the IPService class.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>Object</td>
<td>Either a GlideRecord instance or a sys_id string</td>
</tr>
</tbody>
</table>

### IPService - port

The TCP or UDP port used by the service.

#### Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>port</td>
<td>String</td>
<td>The TCP or UDP port used by the service.</td>
</tr>
</tbody>
</table>

### IPService - protocol

The protocol used by the service ("UDP", "TCP", or "TCP/UDP").

#### Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>protocol</td>
<td>String</td>
<td>The protocol used by the service (&quot;UDP&quot;, &quot;TCP&quot;, or &quot;TCP/UDP&quot;).</td>
</tr>
</tbody>
</table>
IPService - name
A short name or handle for the IPService.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the IPService</td>
</tr>
</tbody>
</table>

IPService - serviceName
A long, descriptive English name for the IPService.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceName</td>
<td>String</td>
<td>A long, descriptive English name for the IPService.</td>
</tr>
</tbody>
</table>

IPService - sysID
The sys_id of this record.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysID</td>
<td>String</td>
<td>The sys_id of this record.</td>
</tr>
</tbody>
</table>

JavascriptProbe - Global
Provides methods for using the JavascriptProbe on the MID server.
Use these methods in server scripts to send a probe to the MID server from JavaScript.

JavascriptProbe - addParameter(String name, String value)
Adds a parameter with the specified value to the probe.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The parameter name</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>The parameter's value</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var jspr = new JavascriptProbe('msserv');
jspr.setName('jsprobe100');
jspr.setSource('jspr_source');
jspr.addParameter('support_feat', 'yes');
```

JavascriptProbe - create()

Creates the probe for the current MID server, puts it on the output queue, and sets its state to "ready".

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var jspr = new JavascriptProbe('msserv');
jspr.setName('jsprobe100');
jspr.setSource('jspr_source');
jspr.setJavascript('jspr_script');
jspr.create();
```

JavascriptProbe - JavascriptProbe(String midServer)

Creates an instance of the JavascriptProbe.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>midServer</td>
<td>String</td>
<td>The MID server name</td>
</tr>
</tbody>
</table>

**JavascriptProbe - setJavascript(String script)**

Adds a script to the probe.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>script</td>
<td>String</td>
<td>The script to add to the probe</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var jspr = new JavascriptProbe('msserv');
jspr.setName('jsprobe100');
jspr.setSource('jspr_source');
jspr.setJavascript('jspr_script');
```

**JavascriptProbe - setName(String name)**

Sets the name of the JavascriptProbe

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name to give the JavascriptProbe.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var jspr = new JavascriptProbe('msserv');
jspr.setName('jsprobe100');

**JavascriptProbe - setSource(String ip)**

Sets the source of the probe.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip</td>
<td>String</td>
<td>The IP address of the target probe.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var jspr = new JavascriptProbe('msserv');
jspr.setName('jsprobe100');
jspr.setSource('100.100.100.1');

**J2js - Global**

2JS script include allows you to convert Java objects to JavaScript objects.

If the given value is a Java object that can be converted to an equivalent JavaScript object, that conversion is performed and the result is returned. Otherwise the original Java object is returned.

The J2js class is available to server-side scripts.

The specific conversion performed in the order they are checked.

- Java String -> JavaScript string
- Java Boolean -> JavaScript boolean
- Java Integer -> JavaScript number
- Java Long -> JavaScript number
- Java Double -> JavaScript number
- Java Byte -> JavaScript number
- Java Float -> JavaScript number
- Java Short -> JavaScript number
• Java Character -> JavaScript number
• Java array -> JavaScript Array with order preserved
• Java List -> JavaScript Array with order preserved
• Java Map -> JavaScript Object with the key/value pairs translated into
  property/value pairs
• Java Set -> JavaScript Array in arbitrary order

Conversions are performed recursively on the elements of arrays, lists, or
collections. For example, given a Java ArrayList of ArrayLists of Strings, this will
return a JavaScript Array of Arrays of strings.

J2js - j2js(Object javaObject)

Converts a java object from system code to a JavaScript object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>javaObject</td>
<td>Object</td>
<td>A Java object from system code such as a Packages call.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>A JavaScript object if the parameter can be converted, otherwise it returns</td>
</tr>
<tr>
<td></td>
<td>the Java object.</td>
</tr>
</tbody>
</table>

```
var tu = new TableUtils("cmdb_ci_win_server");
var classes = tu.getHierarchy();
//getHierarchy returns a Java ArrayList, which is not exactly like a JavaScript Array
//for example you cannot get length
gs.print("classes = " + classes);
gs.print("classes.length = " + classes.length);

//convert to a JavaScript Array
gs.include("j2js");
var jsClasses = j2js(classes);
gs.print("jsClasses = " + jsClasses);
gs.print("jsClasses.length = " + jsClasses.length);
```
Output:

```
classes = [cmdb_ci_win_server, cmdb_ci_server, cmdb_ci_computer, cmdb_ci_hardware, cmdb_ci]
classes.length = undefined
jsClasses = cmdb_ci_win_server, cmdb_ci_server, cmdb_ci_computer, cmdb_ci_hardware, cmdb_ci
jsClasses.length = 5
```

### JSON - Global

Provides methods to create JSON objects from a string, and to turn JSON objects into strings.

The JSON API has dynamic and static methods. You access the dynamic methods by creating a JSON object. To use the dynamic methods in a scoped application, add the global prefix when calling the constructor. You access the static methods by using the static JSON object.

The JavaScript ES5 native JSON object is used instead of the JSON static methods. If your script needs the old behavior, use the `encode()` and `decode()` methods.

This example creates a JSON object.

```javascript
// obj is a JSON formatted object
var parser = new JSON(); // encode() and decode() are dynamic methods.
var str = parser.encode(obj);
```

The `encode()` and `decode()` methods are deprecated. Use the JavaScript JSON object instead.

This example creates a JSON object in a scoped script.

```javascript
// obj is a JSON formatted object
var parser = new global.JSON();
var str = parser.encode(obj);
```

This example uses the static methods.

```javascript
// obj is a JSON formatted object
var str = JSON.stringify(obj);
// str is a string containing content to be turned into a JSON formatted object
var obj = JSON.parse(str);
```

### JSON - decode(String str)

Creates an object or primitive type from a JSON formatted string.

⚠️ **Note:** This class is deprecated. Use the JavaScript JSON object instead.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>str</td>
<td>String</td>
<td>A JSON formatted string.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object created from the specified string.</td>
</tr>
</tbody>
</table>

```
var str = '{"name":"George","lastname":"Washington"}';
var parser = new JSON();
var obj = parser.decode(str);
gs.info('The first name is ' + obj.name);
```

Output: The first name is George

To use in a scoped script.

```
var str = '{"name":"George","lastname":"Washington"}';
var parser = new global.JSON();
var obj = parser.decode(str);
gs.info('The first name is ' + obj.name);
```

Output: The first name is George

**JSON - encode(Object jsonObject)**

Creates a string from a JSON object.

⚠️ Note: This class is deprecated. Use the JavaScript JSON object instead.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jsonObject</td>
<td>Object</td>
<td>The JSON object to be turned into a string.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A JSON formatted string.</td>
</tr>
</tbody>
</table>
```javascript
var obj = {"name":"George","lastname":"Washington"};
var parser = new JSON();
var str = parser.encode(obj);
gs.info('The object ' + str);
```

Output: The object {"lastname":"Washington","name":"George"}

To use in a scoped script.

```javascript
var obj = {"name":"George","lastname":"Washington"};
var parser = new global.JSON();
var str = parser.encode(obj);
gs.info('The object ' + str);
```

**JSON - JSON()**

Creates an instance of the JSON class.

This class is deprecated. Use the JavaScript JSON object instead.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**JSON - parse(String str)**

Creates an object or primitive type from a JSON formatted string.

The JavaScript ES5 native JSON object is used instead of the JSON static methods. If your script needs the old behavior, use the `encode()` and `decode()` methods.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>str</td>
<td>String</td>
<td>A JSON formatted string.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object created from the specified string.</td>
</tr>
</tbody>
</table>
var str = '{"name":"George","lastname":"Washington"}';
var obj = JSON.parse(str);
gs.info('The first name is ' + obj.name);

Output: The first name is George

**JSON - stringify(Object jsonObject)**

Creates a string from a JSON object.

The `JSON.stringify()` method can only convert numbers, strings, and Java
native objects to strings. It cannot convert user-defined objects to strings, unless
those objects provide a `toJSON()` method. The call to `current.sys_id()` returns
a GlideElement object which does not have a `toJSON()` method, so the return
value for `stringify` is empty: `"{}`. The JavaScript ES5 native JSON object is used
instead of the JSON static methods. If your script needs the old behavior, use the
`encode()` and `decode()` methods.

`JSON.stringify()` converts a value to JSON notation using the following
guidelines:

- If the value has a `toJSON()` method, it is responsible for defining the data that is
  serialized.
- Boolean, number, and string objects are converted to the corresponding
  primitive values during stringification; in accordance with the traditional
  conversion semantics.
- If a function, undefined, or a symbol is encountered during conversion,
  it is either omitted (when it is found in an object) or censored to null
  (when it is found in an array). `JSON.stringify()` also returns undefined
  when passing in "pure" values, such as `JSON.stringify(function(){})` or
  `JSON.stringify(undefined)`.
- All symbol-keyed properties are ignored, even when using a `replacer()`
  function.
- Instances of Date implement the `toJSON()` function by returning a string (the
  same as `date.toISOString()`), thus they are treated as strings.
- The numbers Infinity and NaN, as well as the value null, are all considered null.
- For all other object instances, only their enumerable properties are serialized.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>jsonObject</td>
</tr>
</tbody>
</table>
## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>JSON formatted string.</td>
</tr>
</tbody>
</table>

```javascript
var obj = {"name":"George","lastname":"Washington"};
var str = JSON.stringify(obj);
gs.info('The object ' + str);
```

Output: The object `{"name":"George","lastname":"Washington"}`

It is also possible to define a `replacer()` function and use that in the `stringify()` call. This function leverages the `GlideElement.toString()` method to provide a string representation of the GlideElement object.

```javascript
function replacer(name, val) {
    // convert GlideElement to string
    if ( val && val.constructor === GlideElement ) {
        return val.toString();
    } else {
        return val; // return as is
    }
}

var s = global.JSON.stringify(obj, replacer);
gs.info("json="+s);
```

### JSONParser - Global

Provides a JSON parser that does not use the JavaScript `eval()` function. The methods of JSONParser do not validate the JSON string. This class cannot be used in a scoped application. Use the JavaScript JSON object instead.

⚠️ **Note:** This class is deprecated. Use the JavaScript JSON object instead.

### JSONParser - JSONParser()

Creates a JSONParser object. This class cannot be used in a scoped application. Use the JavaScript JSON object instead. This class is deprecated. Use the JavaScript JSON object instead.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## JSON - Scoped

Provides scoped methods to create JSON objects from a string, and to turn JSON objects into strings.

For scoped applications, the JSON API uses static methods that call the JavaScript ES5 native JSON object.

### Scoped JSON - parse(String str)

Creates an object or primitive type from a JSON formatted string.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>str</td>
<td>String</td>
<td>A JSON formatted string.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An object created from the specified string.</td>
</tr>
</tbody>
</table>

```javascript
var str = '{"name":"George","lastname":"Washington"}';
var obj = JSON.parse(str);
gs.info('The first name is ' + obj.name);
```

Output:

```
The first name is George
```

### Scoped JSON - stringify(Object jsonObject)

Creates a string from a JSON object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jsonObject</td>
<td>Object</td>
<td>The JSON object to be turned into a string.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A JSON formatted string.</td>
</tr>
</tbody>
</table>

```javascript
var obj = {"name": "George", "lastname": "Washington"};
var str = JSON.stringify(obj);
gs.info('The object' + str);
```

Output:

The object {"name": "George", "lastname": "Washington"}

**JSONStreamingAPI - Scoped**

Build a large streaming JSON payload to use in a REST or SOAP request to send bulk data to a third-party API. You can also create the payload as a JSON string for a non-streaming option.

Use these methods in the Flow Designer script step with the `sn_ih` namespace identifier. For example, you can use this API to create a JSON payload in the Flow Designer Script step and pass the returned value to the REST step to send the request to a third-party service. For more information, see the Flow Designer Script step.

You can only use this API within the Flow Designer environment.

To use this class, you must call the `build()` method in the `JSONStreamingBuilder` class to return a `JSONStreamingAPI` object. See `JSONStreamingBuilder - Scoped`.

**API call order**

Generate JSON payloads using these APIs in the following order:

1. `JSONStreamingBuilder`: Creates a builder object
   - Use these methods in the following order to create a builder object:
     1. `JSONStreamingBuilder()`: Instantiates the `JSONStreamingBuilder` object.
     2. `withAttachment()`: Optional. Creates the JSON object as a streaming attachment and stores it in the Streaming Attachments.
[streaming_attachment] table. If you do not call this method, the API creates the payload as a JSON string.

3. `expiresAt()`: Optional. Sets a time when the attachment expires. Must also call the `withAttachment()` method.

4. `build()`: Returns a JSONStreamingAPI object.

**JSONStreamingAPI: Builds the JSON payload**

Use these methods in the following order to create the JSON payload:

1. `startObject()`: Creates the parent JSON object.
2. Methods to generate the JSON key-value pairs, such as `writeFieldName()`, `writeString()`, and `writeNumberField()`.
3. `endObject()`: Closes the parent JSON object.
4. `getJSONString()` or `getAttachmentId()`: Returns the JSON string or attachment ID that you created.
5. `close()`: Closes the JSONStreamingAPI object.

**Size limits**

Payloads generated through this API cannot exceed these size limits:

- Attachments: 200 MB
- Strings: 5 MB

**Example**

This example create a JSON object and stores it in the Attachment [sys_attachment] table with a defined expiration date. You can use this option to create payloads under 5 MB.

```javascript
try {
  var ttl = new GlideDateTime("2011-01-01 12:00:00");
  var builder = new sn_ih.JSONStreamingBuilder()
    .withAttachment() // Creates the JSON object in streaming mode within an attachment.
    .expiresAt(ttl) // Sets an expiration date for the attachment.
    .build(); // Creates the JSONStreamingAPI object.

  builder.startObject() // Begins generating the JSON object.
    .writeFieldName("firstName") // Adds a "firstName" field
    .writeString("John") // Writes the value of the "firstName" field
    .writeFieldName("lastName")
    .writeString("Smith")
```

Alternatively, this example uses the API in the Script step and creates the payload as a JSON string. You can use this option to create payloads under 5 MB.

(function execute(inputs, outputs) {

  var builder = new sn_ih.JSONStreamingBuilder().build();

})
builder.startObject()
    .enablePrettyPrint()
    .writeFieldName("firstName")
    .writeString("John")
    .writeFieldName("lastName")
    .writeString("Smith")
    .writeNumberField("age","25")
    .writeFieldName("address")
    .startObject()
      .writeStringField("streetAddress", "21 2nd Street")
      .writeStringField("city", "Santa Clara")
      .writeStringField("state", "CA")
      .writeStringField("postalCode", "11111")
    .endObject()
    .writeFieldName("phoneNumber")
    .startArray()
      .startObject()
        .writeFieldName("type")
        .writeString("home")
        .writeFieldName("number")
        .writeString("212 555-1234")
      .endObject()
      .startObject()
        .writeFieldName("type")
        .writeString("fax")
        .writeFieldName("number")
        .writeString("646 555-4567")
      .endObject()
    .endArray()
  .endObject()

outputs.payload = builder.getJSONString();
}

)(inputs, outputs);

Output:

{
  "firstName" : "John",
  "lastName" : "Smith",
  "age" : 25,
  "address" : {
    "streetAddress" : "21 2nd Street",
    "city" : "Santa Clara",
    "state" : "CA",
    "postalCode" : "11111"
  }
  }
JSONStreamingAPI - close()  
Closes the JSONStreamingAPI object. Must call this method to close the stream after building a JSON object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to create a JSON object and store it in the Attachment [sys_attachment] table with a defined expiration date.

```javascript
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.JSONStreamingBuilder()
        .withAttachment() // Creates the JSON object in streaming mode within an attachment.
        .expiresAt(ttl) // Sets an expiration date for the attachment.
        .build(); // Creates the JSONStreamingAPI object.
    builder.startObject() // Begins generating the JSON object.
        .writeField("firstName") // Adds a "firstName" field
        .writeString("John") // Writes the value of the "firstName" field
        .writeField("lastName")
        .writeString("Smith")
        .writeNumberField("age", "25") // Write a number field named "age" with value "25"
```
```java
.builder.writeFieldName("address")
    .startObject() // Start a new object nested under the parent object
        .writeStringField("streetAddress", "21 2nd Street")
        .writeStringField("city", "Santa Clara")
        .writeStringField("state", "CA")
        .writeStringField("postalCode", "11111")
    .endObject()
    .writeFieldName("phoneNumber")
    .startArray() // Start an array
        .startObject() // Add the first object to the array
            .writeFieldName("type")
            .writeString("home")
            .writeFieldName("number")
            .writeString("212 555-1234")
        .endObject()
        .startObject() // Add another object to the array
            .writeFieldName("type")
            .writeString("fax")
            .writeFieldName("number")
            .writeString("646 555-4567")
        .endObject()
    .endArray()
    .endObject()
}

 gs.log(builder.getAttachmentId()); // Returns the sys_id of the attachment.
}

catch (err) {
    gs.log(err);
}

finally {
    builder.close();
}
```

**JSONStreamingAPI - disablePrettyPrint()**

Ends pretty print JSON formatting.

Before calling this method, you must first call `enablePrettyPrint()` to add JSON formatting to a specific section.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>

This example adds pretty print formatting to the `address` object.

```java
try {
    var builder = new sn_ih.JSONStreamingBuilder().build();

    builder.startObject()
        .writeFieldName("firstName")
        .writeString("John")
        .writeFieldName("lastName")
        .writeString("Smith")
        .writeNumberField("age", "25")
        .enablePrettyPrint()
        .writeFieldName("address")
        .startObject()
            .writeStringField("streetAddress", "21 2nd Street")
            .writeStringField("city", "Santa Clara")
            .writeStringField("state", "CA")
            .writeStringField("postalCode", "11111")
        .endObject()
        .disablePrettyPrint()
        .writeFieldName("phoneNumber")
        .startArray()
            .startObject()
                .writeFieldName("type")
                .writeString("home")
                .writeFieldName("number")
                .writeString("212 555-1234")
            .endObject()
            .startObject()
                .writeFieldName("type")
                .writeString("fax")
                .writeFieldName("number")
        .endArray()
    .endObject()
```
gs.log(builder.toJSONString());
}
catch (err) {
  gs.log("Exception: " + err);
}
finally {
  builder.close();
}

Output:
{
  "firstName":"John","lastName":"Smith","age":25,
  "address" : {
    "streetAddress" : "21 2nd Street",
    "city" : "Santa Clara",
    "state" : "CA",
    "postalCode" : "11111"
  },"phoneNumber":[{
    "type":"home","number":"212 555-1234"},{"type":"fax","number":"646 555-4567"}]
}

**JSONStreamingAPI - enablePrettyPrint()**

Adds pretty print formatting to a JSON object, or a section of a JSON object.

To disable pretty print formatting in a JSON object section, use the `disablePrettyPrint()` method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>
try {
    var builder = new sn_ih.JSONStreamingBuilder().build();

    builder.enablePrettyPrint()
        .startObject()
        .writeFieldName("firstName")
        .writeString("John")
        .writeFieldName("lastName")
        .writeString("Smith")
        .writeNumberField("age", "25")
        .writeFieldName("address")
        .startObject()
        .writeStringField("streetAddress", "21 2nd Street")
        .writeStringField("city", "Santa Clara")
        .writeStringField("state", "CA")
        .writeStringField("postalCode", "11111")
        .endObject()
        .writeFieldName("phoneNumber")
        .startArray()
        .startObject()
        .writeFieldName("type")
        .writeString("home")
        .writeFieldName("number")
        .writeString("212 555-1234")
        .endObject()
        .startObject()
        .writeFieldName("type")
        .writeString("fax")
        .writeFieldName("number")
        .writeString("646 555-4567")
        .endObject()
        .endArray()
        .endObject()

    gs.log(builder.getJSONString());
}

catch (err) {
    gs.log("Exception: " + err);
}

finally {
    builder.close();
}
Output:

```json
{
    "firstName" : "John",
    "lastName" : "Smith",
    "age" : 25,
    "address" : {
        "streetAddress" : "21 2nd Street",
        "city" : "Santa Clara",
        "state" : "CA",
        "postalCode" : "11111"
    },
    "phoneNumber" : [ {
        "type" : "home",
        "number" : "212 555-1234"
    }, {
        "type" : "fax",
        "number" : "646 555-4567"
    } ]
}
```

**JSONStreamingAPI - endArray()**

Closes an array within the parent JSON object.

Call the `startArray()` method first to open the array.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>

This example shows how to create a JSON object and store it in the Attachment [sys_attachment] table with a defined expiration date.

```javascript
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
} `
var builder = new sn_ih.JSONStreamingBuilder()
    .withAttachment() // Creates the JSON object in streaming mode within an attachment.
    .expiresAt(ttl) // Sets an expiration date for the attachment.
    .build(); // Creates the JSONStreamingAPI object.

    builder.startObject() // Begins generating the JSON object.
    .writeFieldName("firstName") // Adds a "firstName" field
    .writeString("John") // Writes the value of the "firstName" field
    .writeFieldName("lastName")
    .writeString("Smith")
    .writeNumberField("age","25") // Write a number field named "age" with value "25"
    .writeFieldName("address")
    .startObject() // Start a new object nested under the parent object
    .writeStringField("streetAddress", "21 2nd Street")
    .writeStringField("city", "Santa Clara")
    .writeStringField("state", "CA")
    .writeStringField("postalCode", "11111")
    .endObject()
    .writeFieldName("phoneNumber")
    .startArray() // Start an array
    .startObject() // Add the first object to the array
    .writeFieldName("type")
    .writeString("home")
    .writeFieldName("number")
    .writeString("212 555-1234")
    .endObject()
    .startObject() // Add another object to the array
    .writeFieldName("type")
    .writeString("fax")
    .writeFieldName("number")
    .writeString("646 555-4567")
    .endObject()
    .endArray()
    .endObject()

    gs.log(builder.getAttachmentId()); // Returns the sys_id of the attachment.
}

try {
}

catch (err) {
    gs.log(err);
}

finally { }
builder.close();

**JSONStreamingAPI - endObject()**

Closes an object within the parent JSON object.

Call the `startObject()` method first to open the object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>

This example shows how to create a JSON object and store it in the Attachment [sys_attachment] table with a defined expiration date.

```javascript
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.JSONStreamingBuilder()
        .withAttachment() // Creates the JSON object in streaming mode within an attachment.
        .expiresAt(ttl) // Sets an expiration date for the attachment.
        .build(); // Creates the JSONStreamingAPI object.

    builder.startObject() // Begins generating the JSON object.
        .writeFieldName("firstName") // Adds a "firstName" field
        .writeString("John") // Writes the value of the "firstName" field
    .writeFieldName("lastName")
        .writeString("Smith")
        .writeNumberField("age","25") // Write a number field named "age" with value "25"
        .writeFieldName("address")
        .startObject() // Start a new object nested under the parent object
        .writeStringField("streetAddress", "21 2nd Street")
        .writeStringField("city", "Santa Clara")
        .writeStringField("state", "CA")
        .writeStringField("postalCode", "11111")
        .endObject()
        .writeFieldName("phoneNumber")
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
```java
.startArray()                  // Start an array
.startObject()                // Add the first object to the array
    .writeFieldName("type")
    .writeString("home")
    .writeFieldName("number")
    .writeString("212 555-1234")
.endObject()
.startObject()                // Add another object to the array
    .writeFieldName("type")
    .writeString("fax")
    .writeFieldName("number")
    .writeString("646 555-4567")
.endObject()
.endArray()
.endObject()

gs.log(builder.getAttachmentId()); // Returns the sys_id of the attachment.
}

catch (err) {
    gs.log(err);
}

finally {
    builder.close();
}

JSONStreamingAPI - getAttachmentId()

Returns the sys_id of the attachment record in the Streaming Attachments [streaming_attachment] table that contains the JSON payload.

You must call the withAttachment() method in the JSONStreamingBuilder class to save the JSON payload as an attachment before calling this method. See JSONStreamingBuilder - Scoped.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This example shows how to create a JSON object and store it in the Attachment [sys_attachment] table with a defined expiration date.

```java
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.JSONStreamingBuilder()
        .withAttachment() // Creates the JSON object in streaming mode within an attachment.
        .expiresAt(ttl) // Sets an expiration date for the attachment.
        .build(); // Creates the JSONStreamingAPI object.

    builder.startObject() // Begins generating the JSON object.
        .writeFieldName("firstName") // Adds a "firstName" field
        .writeString("John") // Writes the value of the "firstName" field
        .writeFieldName("lastName")
        .writeString("Smith")
        .writeNumberField("age","25") // Write a number field named "age" with value "25"
        .writeFieldName("address")
        .startObject() // Start a new object nested under the parent object
            .writeStringField("streetAddress", "21 2nd Street")
            .writeStringField("city", "Santa Clara")
            .writeStringField("state", "CA")
            .writeStringField("postalCode", "11111")
        .endObject()
        .writeFieldName("phoneNumber")
        .startArray() // Start an array
            .startObject() // Add the first object to the array
                .writeFieldName("type")
                .writeString("home")
                .writeFieldName("number")
                .writeString("212 555-1234")
            .endObject()
            .startObject() // Add another object to the array
                .writeFieldName("type")
                .writeString("fax")
                .writeFieldName("number")
                .writeString("646 555-4567")
            .endObject()
        .endArray()
    .endObject();
}
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
gs.log(builder.getAttachmentId()); // Returns the sys_id of the attachment.
}

catch (err) {
    gs.log(err);
}

finally {
    builder.close();
}

**JSONStreamingAPI - getJSONString()**

Returns the JSON object as a string.

To return the JSON object as a string, do not call the `withAttachment()` method in the `JSONStreamingBuilder` class. See **JSONStreamingBuilder - Scoped**.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Contains the JSON object built using the <code>JSONStreamingAPI</code>.</td>
</tr>
</tbody>
</table>

```javascript
try {
    var builder = new sn_ih.JSONStreamingBuilder().build();

    builder.startObject()
        .writeFieldName("firstName")
        .writeString("John")
        .writeFieldName("lastName")
        .writeString("Smith")
        .writeNumberField("age","25")
        .writeFieldName("address")
        .startObject()
        .writeStringField("streetAddress", "21 2nd Street")
    .endObject()
    .endArray()
}
```
.writeStringField("city", "Santa Clara")
.writeStringField("state", "CA")
.writeStringField("postalCode", "11111")
.endObject()
.writeStringField("phoneNumber")
.startArray()
.startObject()
.writeStringField("type")
.writeString("home")
.writeStringField("number")
.writeString("212 555-1234")
.endObject()
.startObject()
.writeStringField("type")
.writeString("fax")
.writeStringField("number")
.writeString("646 555-4567")
.endObject()
.endArray()
.endObject()

gs.log(builder.getJSONString());
}

catch (err) {
    gs.log("Exception: " + err);
}

finally {
    builder.close();
}

Output:

{
    "firstName" : "John",
    "lastName" : "Smith",
    "age" : 25,
    "address" : {
        "streetAddress" : "21 2nd Street",
        "city" : "Santa Clara",
        "state" : "CA",
        "postalCode" : "11111"  
    },
    "phoneNumber" : [ 

JSONStreamingAPI - startArray()

Opens an array within the parent JSON object.

Include the endArray() method to close the array.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>

This example shows how to create a JSON object and store it in the Attachment [sys_attachment] table with a defined expiration date.

```javascript
try {
  var ttl = new GlideDateTime("2011-01-01 12:00:00");
  var builder = new sn_ih.JSONStreamingBuilder()
    .withAttachment() // Creates the JSON object in streaming mode within an attachment.
    .expiresAt(ttl) // Sets an expiration date for the attachment.
    .build(); // Creates the JSONStreamingAPI object.

  builder.startObject() // Begins generating the JSON object.
    .writeFieldName("firstName") // Adds a "firstName" field
    .writeString("John") // Writes the value of the "firstName" field
  .writeFieldName("lastName")
    .writeString("Smith")
  .writeNumberField("age","25") // Write a number field named "age" with value "25"
  .writeFieldName("address")
    .startObject() // Start a new object nested under the parent object
      .writeStringField("streetAddress", "21 2nd Street")
}
```
JSONStreamingAPI - startArrayField(String fieldName)

Creates an array within the parent JSON object.

Surround this method with the **startArray()** and **endArray()** methods to open and close the array.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The name of the array.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>

```java
try {
    var builder = new sn_ih.JSONStreamingBuilder().build();
    builder.startObject()
        .writeFieldName("firstName")
        .writeString("John")
        .writeFieldName("lastName")
        .writeString("Smith")
        .writeNumberField("age", "25")
        .writeFieldName("address")
        .startObject()
        .writeStringField("streetAddress", "21 2nd Street")
        .writeStringField("city", "Santa Clara")
        .writeStringField("state", "CA")
        .writeStringField("postalCode", "11111")
        .endObject()
        .startArrayField("phoneNumber")
        .startArray()
        .startObject()
            .writeFieldName("type")
            .writeString("home")
            .writeFieldName("number")
            .writeString("212 555-1234")
        .endObject()
        .startObject()
            .writeFieldName("type")
            .writeString("fax")
            .writeFieldName("number")
            .writeString("646 555-4567")
        .endObject()
    .endArray()
    .endObject()
    gs.log(builder.getJSONString());
}
```

```java
catch (err) {
    gs.log("Exception: " + err);
}
```
finally {
  builder.close();
}

JSONStreamingAPI - startObject()
Opens an object within the parent JSON object.
Requires the endObject() method to close the object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>

This example shows how to create a JSON object and store it in the Attachment [sys_attachment] table with a defined expiration date.

```javascript
try {
  var ttl = new GlideDateTime("2011-01-01 12:00:00");
  var builder = new sn_ih.JSONStreamingBuilder()
    .withAttachment() // Creates the JSON object in streaming mode within an attachment.
    .expiresAt(ttl) // Sets an expiration date for the attachment.
    .build(); // Creates the JSONStreamingAPI object.

  builder.startObject() // Begins generating the JSON object.
    .writeFieldName("firstName") // Adds a "firstName" field
    .writeString("John") // Writes the value of the "firstName" field
  .writeFieldName("lastName")
    .writeString("Smith")
    .writeNumberField("age","25") // Write a number field named "age" with value "25"
  .writeFieldName("address")
    .startObject() // Start a new object nested under the parent object
      .writeStringField("streetAddress", "21 2nd Street")
      .writeStringField("city", "Santa Clara")
      .writeStringField("state", "CA")

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
JSONStreamingAPI - writeBoolean(Boolean state)

Adds a Boolean value to the parent JSON object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>Boolean</td>
<td>The boolean value to add to the parent JSON object. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>

```java
try {
    var builder = new sn_ih.JSONStreamingBuilder().build();

    builder.startObject()
        .writeFieldName("firstName")
        .writeString("John")
        .writeFieldName("lastName")
        .writeString("Smith")
        .writeFieldName("activeUser")
        .writeBoolean(true)
        .writeNumberField("age", 25)
        .writeFieldName("address")
            .startObject()
                .writeStringField("streetAddress", "21 2nd Street")
                .writeStringField("city", "Santa Clara")
                .writeStringField("state", "CA")
                .writeStringField("postalCode", "11111")
            .endObject()
        .writeFieldName("phoneNumber")
            .startArray()
                .startObject()
                    .writeFieldName("type")
                    .writeString("home")
                    .writeFieldName("number")
                    .writeString("212 555-1234")
                .endObject()
                .startObject()
                    .writeFieldName("type")
                    .writeString("fax")
                    .writeFieldName("number")
                    .writeString("646 555-4567")
                .endObject()
            .endArray()
    .endObject()

    gs.log(builder.getJSONString());
}
```
catch (err) {
    gs.log("Exception: " + err);
}

finally {
    builder.close();
}

JSONStreamingAPI - writeBooleanField(String fieldName, Boolean value)
Adds a Boolean field and value to the parent JSON object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The name of the field to add to the parent JSON object.</td>
</tr>
<tr>
<td>value</td>
<td>Boolean</td>
<td>The boolean value to add to the parent JSON object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>

try {
    var builder = new sn_ih.JSONStreamingBuilder().build();

    builder.startObject()
        .writeFieldName("firstName")
        .writeString("John")
        .writeFieldName("lastName")
        .writeString("Smith")
        .writeBooleanField("activeUser", true)
        .writeNumberField("age", "25")
        .writeFieldName("address")
        .startObject()
        .writeStringField("streetAddress", "21 2nd Street")
        .writeStringField("city", "Santa Clara")
        .writeStringField("state", "CA")
    .closeObject();
}
gs.log(builder.getJSONString());
}

catch (err) {
    gs.log("Exception: " + err);
}

finally {
    builder.close();
}

**JSONStreamingAPI - writeFieldName(String name)**

Adds a field name to the parent JSON object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Field name to add to the parent JSON object.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>
This example shows how to create a JSON object and store it in the Attachment [sys_attachment] table with a defined expiration date.

```java
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.JSONStreamingBuilder()
        .withAttachment() // Creates the JSON object in streaming mode within an attachment.
        .expiresAt(ttl) // Sets an expiration date for the attachment.
        .build(); // Creates the JSONStreamingAPI object.

    builder.startObject() // Begins generating the JSON object.
        .writeFieldName("firstName") // Adds a "firstName" field
        .writeString("John") // Writes the value of the "firstName" field
    .writeFieldName("lastName")
        .writeString("Smith")
    .writeNumberField("age","25") // Write a number field named "age" with value "25"
    .writeFieldName("address")
        .startObject() // Start a new object nested under the parent object
            .writeStringField("streetAddress", "21 2nd Street")
            .writeStringField("city", "Santa Clara")
            .writeStringField("state", "CA")
            .writeStringField("postalCode", "11111")
        .endObject()
    .writeFieldName("phoneNumber")
        .startArray() // Start an array
            .startObject() // Add the first object to the array
                .writeFieldName("type")
                .writeString("home")
                .writeFieldName("number")
                .writeString("212 555-1234")
            .endObject()
            .startObject() // Add another object to the array
                .writeFieldName("type")
                .writeString("fax")
                .writeFieldName("number")
                .writeString("646 555-4567")
            .endObject()
        .endArray()
    .endObject()

    gs.log(builder.getAttachmentId()); // Returns the sys_id of the attachment.
}
```

```java
catch (err) {
    ...
```
try {
    var builder = new sn_ih.JSONStreamingBuilder().build();

    builder.startObject()
        .writeFieldName("firstName")
        .writeString("John")
        .writeFieldName("lastName")
        .writeString("Smith")
        .writeFieldName("activeUser")
        .writeNull()
        .writeNumberField("age", "25")
        .writeFieldName("address")
        .startObject()
            .writeStringField("streetAddress", "21 2nd Street")
            .writeStringField("city", "Santa Clara")
            .writeStringField("state", "CA")
            .writeStringField("postalCode", "11111")
        .endObject()
        .writeFieldName("phoneNumber")
        .startArray()
        .startObject()
            .writeFieldName("type")
        .endObject()
    .startObject()
        .writeFieldName("address")
    .endObject()
} finally {
    builder.close();
}

**JSONStreamingAPI - writeNull()**

Adds a null value to the parent JSON object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>
JSONStreamingAPI - writeNullField(String fieldName)

Adds a field with a null value to the parent JSON object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The name of the null field.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>

```javascript
try {
    var builder = new sn_ih.JSONStreamingBuilder().build();
    builder.startObject()
        .writeFieldName("firstName")
        .writeNullField("lastName")
    .endObject()
}
```
```java
.builder.writeString("John")
.builder.writeString("Smith")
.builder.writeStringField("streetAddress", "21 2nd Street")
.builder.writeStringField("city", "Santa Clara")
.builder.writeStringField("state", "CA")
.builder.writeStringField("postalCode", "11111")
.builder.writeObject()
.builder.writeStringField("phoneNumber")
.builder.startArray()
.builder.writeObject()
.builder.writeStringField("type")
.builder.writeString("home")
.builder.writeStringField("number")
.builder.writeString("212 555-1234")
.builder.writeObject()
.builder.writeObject()
.builder.writeStringField("type")
.builder.writeString("fax")
.builder.writeStringField("number")
.builder.writeString("646 555-4567")
.builder.writeObject()
.builder.writeObject()
.builder.writeObject()

gs.log(builder.getJSONString());
}

catch (err) {
    gs.log("Exception: " + err);
}

finally {
    builder.close();
}
```

**JSONStreamingAPI - writeNumberField(String fieldName, String encodedValue)**

Adds a number field and value to the parent JSON object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The name of the number field.</td>
</tr>
<tr>
<td>encodedValue</td>
<td>String</td>
<td>The value of the number field.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>

This example shows how to create a JSON object and store it in the Attachment [sys_attachment] table with a defined expiration date.

```java
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.JSONStreamingBuilder()
        .withAttachment() // Creates the JSON object in streaming mode within an attachment.
        .expiresAt(ttl) // Sets an expiration date for the attachment.
        .build(); // Creates the JSONStreamingAPI object.

    builder.startObject() // Begins generating the JSON object.
        .writeFieldName("firstName") // Adds a "firstName" field
        .writeString("John") // Writes the value of the "firstName" field
        .writeFieldName("lastName")
        .writeString("Smith")
        .writeNumberField("age", "25") // Write a number field named "age" with value "25"
        .writeFieldName("address")
        .startObject() // Start a new object nested under the parent object
            .writeStringField("streetAddress", "21 2nd Street")
            .writeStringField("city", "Santa Clara")
            .writeStringField("state", "CA")
            .writeStringField("postalCode", "11111")
        .endObject()
        .writeFieldName("phoneNumber")
        .startArray() // Start an array
            .startObject() // Add the first object to the array
                .writeFieldName("type")
                .writeString("home")
                .writeFieldName("number")
                .writeString("212 555-1234")
            .endObject()
        .endArray()
    }
```
```javascript
try {
    var builder = new sn_ih.JSONStreamingBuilder().build();

    builder.startObject()
        // Add another object to the array
        .writeFieldName("type")
        .writeField("fax")
        .writeField("number")
        .writeFieldName("646 555-4567")
    .endObject()
    .endArray()
    .endObject()

    gs.log(builder.getAttachmentId()); // Returns the sys_id of the attachment.
}

catch (err) {
    gs.log(err);
}

finally {
    builder.close();
}
```

**JSONStreamingAPI - writeRaw(String text)**

Adds a raw value to the parent JSON object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>Raw text to add to the parent JSON object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>
```java
 gs.log(builder.getJSONString());
 }

catch (err) {
    gs.log("Exception: " + err);
}

finally {
    builder.close();
}
```

**JSONStreamingAPI - writeString(String text)**

Adds a string value to the parent JSON object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>The string value to add to the parent JSON object.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>

This example shows how to create a JSON object and store it in the Attachment [sys_attachment] table with a defined expiration date.

```javascript
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.JSONStreamingBuilder()
        .withAttachment() // Creates the JSON object in streaming mode within an attachment.
        .expiresAt(ttl) // Sets an expiration date for the attachment.
        .build(); // Creates the JSONStreamingAPI object.

    builder.startObject() // Begins generating the JSON object.
        .writeFieldName("firstName") // Adds a "firstName" field
        .writeString("John") // Writes the value of the "firstName" field
        .writeFieldName("lastName")
        .writeString("Smith")
        .writeNumberField("age","25") // Write a number field named "age" with value "25"
        .writeFieldName("address")
        .startObject() // Start a new object nested under the parent object
            .writeStringField("streetAddress", "21 2nd Street")
            .writeStringField("city", "Santa Clara")
            .writeStringField("state", "CA")
            .writeStringField("postalCode", "11111")
        .endObject()
        .writeFieldName("phoneNumber")
        .startArray() // Start an array
            .startObject() // Add the first object to the array
                .writeFieldName("type")
                .writeString("home")
                .writeFieldName("number")
                .writeString("212 555-1234")
            .endObject()
            .startObject() // Add another object to the array
```
JSONStreamingAPI - writeStringField(String fieldName, String value)

Adds a string field and value to the parent JSON object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>The name of the field to add to the parent JSON object.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The value of the field.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingAPI</td>
<td>Streaming JSON object used to construct the payload.</td>
</tr>
</tbody>
</table>

This example shows how to create a JSON object and store it in the Attachment [sys_attachment] table with a defined expiration date.

```javascript
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.JSONStreamingBuilder()
        .withAttachment() // Creates the JSON object in streaming mode within an attachment.
        .expiresAt(ttl) // Sets an expiration date for the attachment.
        .build(); // Creates the JSONStreamingAPI object.
}
```
builder.startObject() // Begins generating the JSON object.
  .writeFieldName("firstName") // Adds a "firstName" field
  .writeString("John") // Writes the value of the "firstName" field
  .writeFieldName("lastName")
  .writeString("Smith")
  .writeNumberField("age","25") // Write a number field named "age" with value "25"
  .writeFieldName("address")
  .startObject() // Start a new object nested under the parent object
    .writeStringField("streetAddress", "21 2nd Street")
    .writeStringField("city", "Santa Clara")
    .writeStringField("state", "CA")
    .writeStringField("postalCode", "11111")
  .endObject()
  .writeFieldName("phoneNumber")
  .startArray() // Start an array
    .startObject() // Add the first object to the array
      .writeFieldName("type")
      .writeString("home")
      .writeFieldName("number")
      .writeString("212 555-1234")
    .endObject()
    .startObject() // Add another object to the array
      .writeFieldName("type")
      .writeString("fax")
      .writeFieldName("number")
      .writeString("646 555-4567")
    .endObject()
  .endArray()
  .endObject()

gs.log(builder.getAttachmentId()); // Returns the sys_id of the attachment.
}

catch (err) {
  gs.log(err);
}

finally {
  builder.close();
}
**JSONStreamingBuilder - Scoped**

Create a builder object used to build a large streaming JSON payload to use in a REST or SOAP request to send bulk data to a third-party API. You can also create the payload as a JSON string for a non-streaming option.

Use these methods in the Flow Designer script step with the `sn_ih` namespace identifier. For example, you can use this API to create a JSON payload in the Flow Designer Script step and pass the returned value to the REST step to send the request to a third-party service. For more information, see the Flow Designer Script step.

You can only use this API within the Flow Designer environment.

**API call order**

Generate JSON payloads using these APIs in the following order:

**JSONStreamingBuilder: Creates a builder object**

Use these methods in the following order to create a builder object:

1. `JSONStreamingBuilder()`: Instantiates the JSONStreamingBuilder object.
2. `withAttachment()`: Optional. Creates the JSON object as a streaming attachment and stores it in the Streaming Attachments [streaming_attachment] table. If you do not call this method, the API creates the payload as a JSON string.
3. `expiresAt()`: Optional. Sets a time when the attachment expires. Must also call the `withAttachment()` method.
4. `build()`: Returns a JSONStreamingAPI object.

**JSONStreamingAPI: Builds the JSON payload**

Use these methods in the following order to create the JSON payload:

1. `startObject()`: Creates the parent JSON object.
2. Methods to generate the JSON key-value pairs, such as `writeFieldName()`, `writeString()`, and `writeNumberField()`.
3. `endObject()`: Closes the parent JSON object.
4. `getJSONObject()` or `getAttachmentId()`: Returns the JSON string or attachment ID that you created.
5. `close()`: Closes the JSONStreamingAPI object.
Size limits

Payloads generated through this API cannot exceed these size limits:

- Attachments: 200 MB
- Strings: 5 MB

Examples

This example shows how to create a JSON object and store it in the Attachment [sys_attachment] table with a defined expiration date.

```java
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.JSONStreamingBuilder()
        .withAttachment() // Creates the JSON object in streaming mode within an attachment.
        .expiresAt(ttl) // Sets an expiration date for the attachment.
        .build(); // Creates the JSONStreamingAPI object.

    builder.startObject() // Begins generating the JSON object.
        .writeFieldName("firstName") // Adds a "firstName" field
        .writeString("John")          // Writes the value of the "firstName" field
    .writeFieldName("lastName")
    .writeString("Smith")
    .writeNumberField("age","25") // Write a number field named "age" with value "25"
    .writeFieldName("address")
        .startObject() // Start a new object nested under the parent object
            .writeStringField("streetAddress", "21 2nd Street")
            .writeStringField("city", "Santa Clara")
            .writeStringField("state", "CA")
            .writeStringField("postalCode", "11111")
        .endObject()
    .writeFieldName("phoneNumber")
        .startArray() // Start an array
            .startObject() // Add the first object to the array
                .writeFieldName("type")
                .writeString("home")
                .writeFieldName("number")
                .writeString("212 555-1234")
            .endObject()
            .startObject() // Add another object to the array
                .writeFieldName("type")
                .writeString("fax")
                .writeFieldName("number")
                .writeString("646 555-4567")
        .endArray()
}
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Alternatively, this example shows how to use the API in the Script step and create the payload as a JSON string. You can use this option to create payloads under 5 MB.

(function execute(inputs, outputs) {

    var builder = new sn_ih.JSONStreamingBuilder().build();

    builder.startObject()
        .enablePrettyPrint()
        .writeTextElement("firstName","John")
        .writeString("John")
        .writeFieldName("lastName")
        .writeString("Smith")
        .writeNumberField("age","25")
        .writeFieldName("address")
        .startObject()
            .writeStringField("streetAddress", "21 2nd Street")
            .writeStringField("city", "Santa Clara")
            .writeStringField("state", "CA")
            .writeStringField("postalCode", "11111")
        .endObject()
        .writeFieldName("phoneNumber")
        .startArray()
            .startObject()
                .writeFieldName("type")
                .writeString("home")
                .writeFieldName("number")
                .writeString("212 555-1234")

    })
Output:

{{
"firstName": "John",
"lastName": "Smith",
"age": 25,
"address": {
  "streetAddress": "21 2nd Street",
  "city": "Santa Clara",
  "state": "CA",
  "postalCode": "11111"
},
"phoneNumber": [
  {
    "type": "home",
    "number": "212 555-1234"
  }, {
    "type": "fax",
    "number": "646 555-4567"
  }
]
}}

**JSONStreamingBuilder - build()**

Returns a JSONStreamingAPI object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This example shows how to create a JSON object and store it in the Attachment [sys_attachment] table with a defined expiration date.

```javascript
try {
  var ttl = new GlideDateTime("2011-01-01 12:00:00");
  var builder = new sn_ih.JSONStreamingBuilder()
    .withAttachment() // Creates the JSON object in streaming mode within an attachment.
    .expiresAt(ttl) // Sets an expiration date for the attachment.
    .build(); // Creates the JSONStreamingAPI object.

  builder.startObject() // Begins generating the JSON object.
    .writeFieldName("firstName") // Adds a "firstName" field
    .writeString("John") // Writes the value of the "firstName" field
    .writeFieldName("lastName")
    .writeString("Smith")
    .writeNumberField("age","25") // Write a number field named "age" with value "25"
    .writeFieldName("address")
    .startObject() // Start a new object nested under the parent object
      .writeStringField("streetAddress", "21 2nd Street")
      .writeStringField("city", "Santa Clara")
      .writeStringField("state", "CA")
      .writeStringField("postalCode", "11111")
    .endObject()
    .writeFieldName("phoneNumber")
    .startArray() // Start an array
      .startObject() // Add the first object to the array
        .writeFieldName("type")
        .writeString("home")
        .writeFieldName("number")
        .writeString("212 555-1234")
      .endObject()
      .startObject() // Add another object to the array
        .writeFieldName("type")
        .writeString("fax")
        .writeFieldName("number")
        .writeString("646 555-4567")
      .endObject()
    .endArray()
}
```
```
var ttl = new GlideDateTime("2011-01-01 12:00:00");
var builder = new sn_ih.JSONStreamingBuilder()
    .withAttachment() // Creates the JSON object in streaming mode within an attachment.
    .expiresAt(ttl) // Sets an expiration date for the attachment.
    .build(); // Creates the JSONStreamingAPI object.
```
builder.startObject() // Begins generating the JSON object.
    .writeFieldName("firstName")   // Adds a "firstName" field
    .writeString("John")          // Writes the value of the "firstName" field
    .writeFieldName("lastName")
    .writeString("Smith")
    .writeNumberField("age","25") // Write a number field named "age" with value "25"
    .writeFieldName("address")   // Start a new object nested under the parent object
        .writeStringField("streetAddress", "21 2nd Street")
        .writeStringField("city", "Santa Clara")
        .writeStringField("state", "CA")
        .writeStringField("postalCode", "11111")
        .endObject()
    .writeFieldName("phoneNumber")
        .startArray() // Start an array
            .startObject()  // Add the first object to the array
                .writeFieldName("type")
                .writeString("home")
                .writeFieldName("number")
                .writeString("212 555-1234")
            .endObject()
            .startObject()  // Add another object to the array
                .writeFieldName("type")
                .writeString("fax")
                .writeFieldName("number")
                .writeString("646 555-4567")
            .endObject()
        .endArray()
    .endObject()

gs.log(builder.getAttachmentId()); // Returns the sys_id of the attachment.
}

} catch (err) {
    gs.log(err);
}

} finally {
    builder.close();
}
**JSONStreamingBuilder - JSONStreamingBuilder()**

Instantiates the JSONStreamingBuilder object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

```javascript
var builder = new sn_ih.JSONStreamingBuilder()
```

**JSONStreamingBuilder - withAttachment()**

Creates the JSON object as a streaming attachment and stores it in the Streaming Attachments [streaming_attachment] table. If you do not call this method, the API creates the payload as a JSON string.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSONStreamingBuilder</td>
<td>Builder object used to initiate the JSON payload.</td>
</tr>
</tbody>
</table>

This example shows how to build the JSON payload and save it as an attachment.

```javascript
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.JSONStreamingBuilder()
        .withAttachment() // Creates the JSON object in streaming mode within an attachment.
        .expiresAt(ttl) // Sets an expiration date for the attachment.
        .build(); // Creates the JSONStreamingAPI object.

    builder.startObject() // Begins generating the JSON object.
        .writeFieldName("firstName") // Adds a "firstName" field
        .writeString("John") // Writes the value of the "firstName" field
        .writeFieldName("lastName")
        .writeString("Smith")
        .writeNumberField("age","25") // Write a number field named "age" with value "25"
}
```
writeFieldName("address")  // Start a new object nested under the parent object
.writeStringField("streetAddress", "21 2nd Street")
.writeStringField("city", "Santa Clara")
.writeStringField("state", "CA")
.writeStringField("postalCode", "11111")
.endObject()

writeFieldName("phoneNumber")
.startArray()  // Start an array
.startObject()  // Add the first object to the array
.writeFieldName("type")
.writeString("home")
.writeStringField("number")
.writeString("212 555-1234")
.endObject()
.startObject()  // Add another object to the array
.writeFieldName("type")
.writeString("fax")
.writeStringField("number")
.writeString("646 555-4567")
.endObject()
.endArray()
.endObject()

gs.log(builder.getAttachmentId());  // Returns the sys_id of the attachment.
}

catch (err) {
    gs.log(err);
}

finally {
    builder.close();
}

This example shows how to build the JSON payload and save it as a string.

try {
    var builder = new sn_ih.JSONStreamingBuilder().build();

    builder.startObject()
      .writeFieldName("firstName")
      .writeString("John")
      .writeFieldName("lastName")

.writeString("Smith")
.writeStringField("age","25")
.writeFieldName("address")
.startObject()
.writeStringField("streetAddress", "21 2nd Street")
.writeStringField("city", "Santa Clara")
.writeStringField("state", "CA")
.writeStringField("postalCode", "11111")
.endObject()
.writeFieldName("phoneNumber")
.startArray()
.startObject()
.writeFieldName("type")
.writeString("home")
.writeFieldName("number")
.writeString("212 555-1234")
.endObject()
.startObject()
.writeFieldName("type")
.writeString("fax")
.writeFieldName("number")
.writeString("646 555-4567")
.endObject()
.endArray()
.endObject()

gs.log(builder.getJSONString());
}

catch (err) {
    gs.log("Exception: " + err);
}

finally {
    builder.close();
}

Output:
{
    "firstName" : "John",
    "lastName" : "Smith",
    "age" : 25,
    "address" : {
        "streetAddress" : "21 2nd Street",
        "city" : "Santa Clara",
        "state" : "CA",
        "postalCode" : "11111"
    }
}
JSUtil - Global

JSUtil is a class of shortcuts for common JavaScript routines.

Script includes and business rules that are marked as Application = "global" and Accessible from = "All applications" can be used in scoped scripts.

JSUtil is not available in scoped scripts.

The JSUtil API is available in server-side scripts.

JSUtil - doesNotHave(Object item)

Checks if item is null or is undefined.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>Object</td>
<td>The object to check</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the specified object is null or undefined.</td>
</tr>
</tbody>
</table>

```
var x = "the quick brown fox";
var y = ";
var z;

gs.print("x = \" + x + ", JSUtil.doesNotHave(x) = " + JSUtil.doesNotHave(x));
```
gs.print("y = "' + y + ", JSUtil.doesNotHave(y) = " + JSUtil.doesNotHave(y));
gs.print("z = "' + z + ", JSUtil.doesNotHave(z) = " + JSUtil.doesNotHave(z));

Output:
x = 'the quick brown fox', JSUtil.doesNotHave(x) = false
y = '', JSUtil.doesNotHave(y) = false
z = 'undefined', JSUtil.doesNotHave(z) = true

**JSUtil - escapeAttr(String text)**

Escape ampersands commonly used to define URL attributes.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>The text</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The text with ampersands properly escaped.</td>
</tr>
</tbody>
</table>

```javascript
var attr = "sysparm_query-active=true&sysparm_view=special";
gs.print(JSUtil.escapeAttr(attr));
```

Output: This is the returned text. If the text is displayed in the application, the page will render the escaped ampersand with a single ampersand.

sysparm_query-active=true&sysparm_view=special

**JSUtil - escapeText(String text)**

Escapes invalid XML characters such as "< > &".

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>The text</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The text with escape characters added.</td>
</tr>
</tbody>
</table>

```javascript
var html = "<b>This is my title</b>";

gs.print(JSUtil.escapeText(html));
```

Output: This is the value returned. If the result is displayed in the application, the page renders the brackets back so it appears that it is not escaped.

```javascript
<b>This is my title</b>
```

JSUtil - getBooleanValue(GlideRecord now_GR, String field)

Returns the value in a boolean GlideRecord field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>A GlideRecord</td>
</tr>
<tr>
<td>field</td>
<td>String</td>
<td>The field from which to retrieve the boolean value.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns the value in a boolean GlideRecord field, returns true if value of field is true, &quot;true&quot;, 1, or &quot;1&quot;.</td>
</tr>
</tbody>
</table>

```javascript
var inc = new GlideRecord("incident");

//get an active incident
inc.addActiveQuery();
inc.setLimit(1);
inc.query();
inc.next();

gs.print(JSUtil.getBooleanValue(inc, "active"));
```

Output: true
**JSUtil - has(Object item)**

Checks if item is not null and is not undefined.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>Object</td>
<td>The Object to check</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the specified object is not null and is not undefined.</td>
</tr>
</tbody>
</table>

```javascript
var x = "the quick brown fox";
var y = "";
var z;

gs.print("x = " + x + ", JSUtil.has(x) = " + JSUtil.has(x));
gs.print("y = " + y + ", JSUtil.has(y) = " + JSUtil.has(y));
gs.print("z = " + z + ", JSUtil.has(z) = " + JSUtil.has(z));
```

**Output:**

```
x = 'the quick brown fox', JSUtil.has(x) = true
y = '', JSUtil.has(y) = true
z = 'undefined', JSUtil.has(z) = false
```

**JSUtil - instance_of(Object item, String class)**

Checks to see if the specified object is a member of the specified class.

For JavaScript objects, this method behaves exactly like the JavaScript operator "instanceof", but also supports Java objects.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>Object</td>
<td>The object to check</td>
</tr>
<tr>
<td>class</td>
<td>String</td>
<td>The class to check</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the specified object is a member of the specified class.</td>
</tr>
</tbody>
</table>

```javascript
var a = ['a', 'b', 'c'];
var b = 10;
var c = new GlideRecord("incident");

gs.print("JSUtil.instance_of(a,'Array') = " + JSUtil.instance_of(a,Array));
gs.print("JSUtil.instance_of(a,'String') = " + JSUtil.instance_of(a,String));

gs.print("JSUtil.instance_of(b,'String') = " + JSUtil.instance_of(b,String));

js.print("JSUtil.instance_of(c,'GlideRecord') = " + JSUtil.instance_of(c,GlideRecord));
```

**Output:**

JSUtil.instance_of(a,'Array') = true
JSUtil.instance_of(a,'String') = false
JSUtil.instance_of(b,'String') = false
JSUtil.instance_of(c,'GlideRecord') = true

### JSUtil - isJavaObject(Object value)

Checks if the specified object is a Java class.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Object</td>
<td>The object to check</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the specified object is an instance of a Java class.</td>
</tr>
</tbody>
</table>

```javascript
var tu = new TableUtils("incident");
var classes = tu.getHierarchy(); //Java ArrayList
var tables = ["task", "incident"]; //JavaScript Array
```
gs.print("JSUtil.isJavaObject(classes) = " + JSUtil.isJavaObject(classes));
gs.print("JSUtil.isJavaObject(tables) = " + JSUtil.isJavaObject(tables));

Output:

JSUtil.isJavaObject(classes) = true
JSUtil.isJavaObject(tables) = false

**JSUtil - logObject(Object obj, String name)**

Logs all the properties in the given object: name, type, and value.

Output is written to the console if you are running from a background script or have debug logging enables. The output is also written to the system log.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>Object</td>
<td>The object for which to enumerate properties</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Optional name for the logged object</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var arr = ['a','b','c'];

var inc = new GlideRecord('incident');
//get an active incident
inc.addActiveQuery();
inc.setLimit(1);
inc.query();
inc.next();

JSUtil.logObject(arr, "arr");
JSUtil.logObject(inc, "inc");
```

Output:

Log Object: arr
Array of 3 elements
[0]: string = a
JSUtil - nil(Object item)
Checks if item is null, undefined, or evaluates to the empty string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>Object</td>
<td>The object to check</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the item is null, undefined, or evaluates to the empty string.</td>
</tr>
</tbody>
</table>

```javascript
var x = "the quick brown fox";
var y = "";
var z;

gs.print("x = " + x + ", JSUtil.nil(x) = " + JSUtil.nil(x));
gs.print("y = " + y + ", JSUtil.nil(y) = " + JSUtil.nil(y));
gs.print("z = " + z + ", JSUtil.nil(z) = " + JSUtil.nil(z));
```

Output:

- x = 'the quick brown fox', JSUtil.nil(x) = false
- y = '', JSUtil.nil(y) = true
- z = 'undefined', JSUtil.nil(z) = true

JSUtil - notNil(Object item)
Checks if item is null, undefined, or evaluates to the empty string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>Object</td>
<td>The object to check</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>--------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Boolean</td>
<td>True if the item exists and is not empty.</td>
<td></td>
</tr>
</tbody>
</table>

```
var x = "the quick brown fox";
var y = "";
var z;

gs.print("x = '" + x + "', JSUtil.notNil(x) = " + JSUtil.notNil(x));
gs.print("y = '" + y + "', JSUtil.notNil(y) = " + JSUtil.notNil(y));
gs.print("z = '" + z + "', JSUtil.notNil(z) = " + JSUtil.notNil(z));
```

**Output:**

```
x = 'the quick brown fox', JSUtil.notNil(x) = true
y = '', JSUtil.notNil(y) = false
z = 'undefined', JSUtil.notNil(z) = false
```

**JSUtil - toBoolean(Object item)**

Converts the specified object to a Boolean.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>Object</td>
<td>The object to convert</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>If the specified object is a boolean, it is passed through. Non-zero numbers return true. Null or undefined return false. Strings return true only if exactly equal to 'true'.</td>
</tr>
</tbody>
</table>

```
var zero = 0;
var one = 1;
var number = 12;
var trueBoolean = true;
var trueString = "true";
var otherString = "random text";
```
gs.print("JSUtil.toBoolean(zero) = "+JSUtil.toBoolean(zero));
gs.print("JSUtil.toBoolean(one) = " + JSUtil.toBoolean(one));
gs.print("JSUtil.toBoolean(number) = " + JSUtil.toBoolean(number));
gs.print("JSUtil.toBoolean(trueBoolean) = " + JSUtil.toBoolean(trueBoolean));
gs.print("JSUtil.toBoolean(trueString) = " + JSUtil.toBoolean(trueString));
gs.print("JSUtil.toBoolean(otherString) = " + JSUtil.toBoolean(otherString));

Output:

JSUtil.toBoolean(zero) = false
JSUtil.toBoolean(one) = true
JSUtil.toBoolean(number) = true
JSUtil.toBoolean(trueBoolean) = true
JSUtil.toBoolean(trueString) = true
JSUtil.toBoolean(otherString) = false

JSUtil - type_of(Object value)

Determines the type of the specified object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Object</td>
<td>The object to check</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String | The type of the specified object.  
  • 'null' if the given value is null or undefined  
  • 'string' if the given value is a primitive string or a String wrapper instance  
  • 'number' if the given value is a primitive number or a Number wrapper instance  
  • 'boolean' if the given value is a primitive boolean or a Boolean wrapper instance  
  • 'function' if the given value is a function  
  • 'object' otherwise |
var a = ["a","b","c"];
var b = 10;
var c = new GlideRecord("incident");
var d = true;
var e;

gs.print("JSUtil.type_of(a) = "+JSUtil.type_of(a));
gs.print("JSUtil.type_of(b) = "+JSUtil.type_of(b));
gs.print("JSUtil.type_of(c) = "+JSUtil.type_of(c));
gs.print("JSUtil.type_of= "+JSUtil.type_of(d));
gs.print("JSUtil.type_of(e) = "+JSUtil.type_of(e));

Output:

JSUtil.type_of(a) = object
JSUtil.type_of(b) = number
JSUtil.type_of(c) = object
JSUtil.type_of= boolean
JSUtil.type_of(e) = null

**JSUtil - unescapeAttr(String text)**

Restore ampersands from escaped text.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>The text</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The text with escape characters removed.</td>
</tr>
</tbody>
</table>

var attr = "sysparm_query-active=true\&sysparm_view=special";

gs.print(JSUtil.unescapeAttr(attr));

Output:

sysparm_query-active=true&sysparm_view=special
JSUtil - unescapeText(String text)

Removes escape characters.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>The text to process.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The text without escape characters.</td>
</tr>
</tbody>
</table>

```javascript
var html = "&lt;b&gt;This is my title&lt;/b&gt;";
gs.print(JSUtil.unescapeText(html));
```

Output: This is the value returned. If the text is displayed in the application the page, it renders the html tags and displays the text in bold.

```html
<b>This is my title</b>
```

KMFCryptoOperation API - Scoped, Global

The KMFCryptoOperation class provides methods for performing cryptographic operations using the KMF cryptographic module.

To use this API, you must have already created and configured a Key Management Framework (KMF) cryptographic module with one or more cryptographic specifications and created/imported its associated key. For details, see Cryptographic module overview.

The KMFCryptoOperation object generated using this API represents a cryptographic operation, such as a Symmetric Encryption. Use the KMFCryptoOperations() method to create this object, the builder methods to set properties on the object, and the doOperation() method to execute the operation.

You can use this API in both scoped and global applications. You must always specify the `sn_kmf_ns` namespace when calling this API.

KMFCryptoOperation - KMFCryptoOperation(String cryptoModuleName, String operationName)

Creates a KMFCryptoOperation object for the specified module and operation.
This API leverages builder methods. Builder methods update properties on the KMFCryptoOperation object, such as changing the desired output format of the data. Not all builder methods are valid for all operations. The builder methods available for each operation are noted in the parameters table below.

The following builder methods are valid for all operation types:

- `withInputFormat()`
- `withOutputFormat()`
- `withOutputType()`

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cryptoModuleName</td>
<td>String</td>
<td>Name of the Key Management Framework (KMF) cryptographic module to use. You must create this module before calling this method. For details, see Create a cryptographic module.</td>
</tr>
<tr>
<td>operationName</td>
<td>String</td>
<td>Name of the operation to perform. Valid values (not case-sensitive):</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ASYMMETRIC_DECRYPTION: Data decryption using an asymmetric-key algorithm. Requires a KMF cryptographic module with an Asymmetric Data Decryption cryptographic purpose.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Additional builder methods: withAdditionalInput()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default input format: Formatted - Formatted to the KMF specifications</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default output format: KMFBASE64 - Base64 encoded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default output type: String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ASYMMETRIC_ENCRYPTION: Data encryption using an asymmetric-key algorithm. Requires a KMF cryptographic module with an Asymmetric Data Encryption cryptographic purpose.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Additional builder methods: withAdditionalInput()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default input format: KMFBASE64 - Base64 encoded</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Default output format: Formatted - Formatted to the KMF specifications</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Default output type: String. Output can also be an KMFEncryptionPayload object. RSA and ECIES are compatible with both. For additional information on the KMFEncryptionPayload object, see withAdditionalInput().</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ASYMMETRIC_UNWRAPPING: Key unwrapping using an asymmetric-key algorithm. Requires a KMF cryptographic module with an Asymmetric Key Unwrapping cryptographic purpose.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Additional builder methods: withAlgorithm()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default input format: Formatted - Formatted to the KMF specifications</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default output format: KMBase64 - Base64 encoded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default output type: String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ASYMMETRIC_WRAPPING: Key wrapping using an asymmetric-key algorithm. Requires a KMF cryptographic module with an Asymmetric Key Wrapping cryptographic purpose.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Additional builder methods: withAlgorithm(), withSysId()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default input format: KMBase64 - Base64 encoded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default output format: Formatted - Formatted to the KMF specifications</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default output type: String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MAC_GENERATION: Generation of a Message Authentication Code (MAC). Symmetric-key algorithm based to provides data integrity and authentication. Requires a KMF cryptographic module with a Symmetric Authenticity cryptographic purpose.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Additional builder methods: None
- Default input format: KMFBase64 - Base64 encoded
- Default output format: Formatted - Formatted to the KMF specifications
- Default output type: String

- MAC_VERIFICATION: Verification of a MAC. Symmetric-key algorithm based to provide data integrity and authentication. Requires a KMF cryptographic module with a Symmetric Authenticity cryptographic purpose.
  - Additional builder methods:
    - withAdditionalInput()
  - Default input format: KMFBase64 - Base64 encoded
  - Default output format: KMFNone - No decoding
  - Default output type: Boolean

- SIGNATURE_GENERATION: Generation of a digital signature. Asymmetric-key algorithm based to provide data integrity and authentication. Requires a KMF cryptographic module with a Signature Generation cryptographic purpose.
  - Additional builder methods: None
  - Default input format: KMFBase64 - Base64 encoded
  - Default output format: Formatted - Formatted to the KMF specifications
  - Default output type: String

- SIGNATURE_VERIFICATION: Verification of a digital signature. Asymmetric-key algorithm based to provide data integrity and authentication. Requires a KMF cryptographic module with a Signature Verification cryptographic purpose.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>◦ Additional builder methods:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ withAdditionalInput()</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Default input format: KMFBase64</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Base64 encoded</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Default output format: KMFNone</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ No decoding</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Default output type: Boolean</td>
<td></td>
</tr>
<tr>
<td>SYMMETRIC_ENCRYPTION:</td>
<td>Data encryption using a symmetric-key algorithm. If the algorithm is not equality preserving, only formatted output is allowed. Requires a KMF cryptographic module with a Symmetric Data Encryption/Decryption cryptographic purpose.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Additional builder methods: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Default input format: KMFBase64</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Base64 encoded</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Default output format: Formatted</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Formatted to the KMF specifications</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Default output type: String</td>
<td></td>
</tr>
<tr>
<td>SYMMETRIC_DECRYPTION:</td>
<td>Data decryption using a symmetric-key algorithm. If the algorithm is not equality preserving, KMFBase64 input is allowed. Requires a KMF cryptographic module with a Symmetric Data Encryption/Decryption cryptographic purpose.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Additional builder methods: None</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Default input format: Formatted</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Formatted to the KMF specifications</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Default output format: KMFBase64</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Base64 encoded</td>
<td></td>
</tr>
<tr>
<td></td>
<td>◦ Default output type: String</td>
<td></td>
</tr>
<tr>
<td>SYMMETRIC_WRAPPING:</td>
<td>Key wrapping using a symmetric-key algorithm. If the algorithm is not equality preserving, only formatted output is allowed. Requires a KMF cryptographic module</td>
<td></td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>with a Symmetric Key Wrapping/Unwrapping cryptographic purpose.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Additional builder methods: withAlgorithm() and withSysId()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default input format: KMFBase64 - Base64 encoded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default output format: Formatted - Formatted to the KMF specifications</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default output type: String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SYMMETRIC_UNWRAPPING: Key unwrapping using a symmetric-key algorithm. If the algorithm is not equality preserving, KMFBase64 input is allowed. Requires a KMF cryptographic module with a Symmetric Key Wrapping/Unwrapping cryptographic purpose.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Additional builder methods: withAlgorithm()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default input format: Formatted - Formatted to the KMF specifications</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default output format: KMFBase64 - Base64 encoded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>◦ Default output type: String</td>
</tr>
</tbody>
</table>

This example instantiates a KMFCryptoOperation object for the module `global.sj_cm` to perform a Symmetric Encryption operation. You must include the namespace for both global and scoped applications.

```javascript
var op = new sn_kmf_ns.KMFCryptoOperation("global.sj_cm","SYMMETRIC_ENCRYPTION");
```

This example shows how to specify options to update the default output type and output format.

```javascript
var op = new sn_kmf_ns.KMFCryptoOperation("global.sj_cm","SYMMETRIC_ENCRYPTION")
 .withOutputType("STRING").withOutputFormat("FORMATTED");

var cipherText=op.doOperation("hi");
```
This example shows how to perform an Asymmetric Encryption operation using an Integrated Encryption Scheme (EC-IES). Note that long values, such as `signature`, have been truncated and replaced with an ellipse for readability.

```javascript
var op = new sn_kmf_ns.KMFCryptoOperation("global.sj_cm","ASYMMETRIC_ENCRYPTION")
  .withInputFormat("KMFNONE")
  .withOutputType("PAYLOAD");

var cipherText = op.doOperation("hi");

/*
cipherText contains an object similar to this JSON: {
  "signature": "pkg...",
  "ephemeral_key": "BDi...",
  "ciphertext": "afFS..."
}
*/
```

This example shows how to perform an Asymmetric Decryption operation using EC-IES.

```javascript
var op = new sn_kmf_ns.KMFCryptoOperation("global.sj_cm","ASYMMETRIC_DECRYPTION")
  .withAdditionalInput({
    "signature": "pkg...",
    "ephemeral_key": "BDi..."
  })
  .withOutputFormat("KMFNONE");

var clearText = op.doOperation("afFS...");
```

** KMFCryptoOperation - doOperation(Object data) **

Performs the cryptographic operation defined by the current KMFCryptoOperation object on the supplied data and returns the result.

** Parameters **

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Object</td>
<td>Required except if the <code>withSysId()</code> builder method has previously been called on the associated KMFCryptoOperation object. Input data on which to perform the cryptographic operation.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depends on the operation type.</td>
<td>Data results after performing the operation specified in the associated KMFCryptoOperation object.</td>
</tr>
<tr>
<td>• MAC_VERIFICATION and</td>
<td></td>
</tr>
<tr>
<td>SIGNATURE_VERIFICATION:</td>
<td></td>
</tr>
<tr>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>• All others: String</td>
<td></td>
</tr>
</tbody>
</table>

This example uses the `doOperation()` to create a MAC.

```javascript
var op = new sn_kmf_ns.KMFCryptoOperation("global.sj_cm","MAC_GENERATION");
var data = "aGk=";
var mac = op.doOperation(data);
```

**KMFCryptoOperation - withAdditionalInput(Object additionalInput)**

Sets the additional input needed to perform the cryptographic operation.

For example, during a Message Authentication Code (MAC) verification, use this method to pass in the generated MAC tag. Similarly, during signature verification, use it to pass in the signature. You can also use this method to pass additional data, a KMFEncryptionPayload object, when performing an asymmetric operation with an integrated cipher, such as Elliptic Curve Integrated Encryption Scheme (EC-IES.)

ℹ️ **Note:** The additional input does not have to be in the same format as what is currently set on the KMF Crypto Operation object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalInput</td>
<td>String or Object</td>
<td>Optional, except for Asymmetric Decryption operations when using EC-IES. Additional input data needed to perform the cryptographic operation specified in the KMFCryptoOperation object. Supported string formats:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FORMATTED: Formatted to the Key Management Framework (KMF) specifications.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• KMFBASE64: Base64 encoded.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalInput.ciphertext</td>
<td>String (Base64)</td>
<td>Required for Asymmetric Decryption, optional for all other operations. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If using an RSA algorithm: RSA ciphertext</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If using an EC-IES algorithm: Integrated AES ciphertext</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Available from the Asymmetric Encryption operation when the output type is set to <strong>payload</strong>.</td>
</tr>
<tr>
<td>additionalInput.derivation_secret</td>
<td>String (Base64)</td>
<td>Optional, only used for the Asymmetric Encryption or Asymmetric Decryption operations with EC-IES. Shared secret to use during the key derivation process of the integrated scheme.</td>
</tr>
<tr>
<td>additionalInput.ephemeral_key</td>
<td>String (Base64)</td>
<td>Required for the Asymmetric Decryption operation when using EC-IES, optional for all other operations. Ephemeral public key to use during the basic agreement process of the integrated scheme.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Available from the Asymmetric Encryption operation when the output type is set to <strong>payload</strong>.</td>
</tr>
<tr>
<td>additionalInput.ephemeral_key_format</td>
<td>String</td>
<td>Optional, only used for the Asymmetric Encryption or Asymmetric Decryption operations with EC-IES. Overrides the format of the public key represented by the <code>ephemeral_key</code> parameter. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• x962</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• der</td>
</tr>
</tbody>
</table>

**KMFEncryptionPayload object format:**

```
{
  "ciphertext": String,
  "derivation_secret": String,
  "ephemeral_key": String,
  "ephemeral_key_format": String,
  "signature": String
}
```
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalInput.signature</td>
<td>String (Base64)</td>
<td>Required for the Asymmetric Decryption operation with EC-IES, optional for all others. The signature of the ciphertext to validate using the signature verification process of the integrated scheme. Available from the Asymmetric Encryption operation when the output type is set to <code>payload</code>.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example uses `withAdditionalInput()` to add a string-based signature to the KMFCryptoOperation object.

```javascript
var signature = "John Doe";
var op = new sn_kmf_ns.KMFCryptoOperation("global.sj_cm","SIGNATURE_VERIFICATION")
  .withAdditionalInput(signature);

var value = GlideStringUtil.base64Encode("Text to encode"); // Default input format is KMFBase64
var result = op.doOperation(String(value));
```

This example uses `withAdditionalInput()` to add a signature and ephemeral key to the KMFCryptoOperation object. Note that long values, such as those in the `doOperation()` call and `payload` description, have been truncated and replaced with an ellipse for readability.

```javascript
var payload = new sn_kmf_ns.KMFCryptoPayload();
payload.signature = "pkg...");
payload.ephemeral_key = " BDi...";
payload.ephemeral_key_format = "x962";

var op = new sn_kmf_ns.KMFCryptoOperation("global.sj_cm","ASYMMETRIC_DECRYPTION")
  .withAdditionalInput(payload)
  .doOperation("afFS...");
```

KMFCryptoOperation - `withAlgorithm(String algorithm)`

Sets the algorithm associated with the key material to wrap.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>algorithm</td>
<td>String</td>
<td>Algorithm to use. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AES: Symmetric key type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• EC: Asymmetric key type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• HMAC: Symmetric key type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• RSA: Asymmetric key type</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example uses `withAlgorithm()` to change the encryption algorithm used to EC.

```javascript
var op = new sn_kmf_ns.KMFCryptoOperation("global.sj_cm", "ASYMMETRIC_WRAPPING").withAlgorithm("EC");
var value = GlideStringUtil.base64Encode("Sample key"); // Default input format is KMFBase64
var result = op.doOperation(String(value));
```

**KMFCryptoOperation - withInputFormat(String inputFormat)**

Sets the data format for the input data on which the cryptographic operation will be performed. Uses the specified format when decoding the data.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputFormat</td>
<td>String</td>
<td>Format of the input data. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• FORMATTED: Formatted to the Key Management Framework (KMF) specifications.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• KMFBASE64: Base64 encoded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• KMFNONE: No encoding.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Default: Value determined by the operation specified when the KMFCryptoOperation object was instantiated. For more information, see KMFCryptoOperation - KMFCryptoOperation(String cryptoModuleName, String operationName).</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example uses `withInputFormat()` to change the input format to have no encoding.

```javascript
var op = new sn_kmf_ns.KMFCryptoOperation("global.sj_cm","SYMMETRIC_ENCRYPTION").withInputFormat("KMFNone");
var result = op.doOperation("Text with no encoding"); // Pass in unencrypted String
```

**KMFCryptoOperation - withOutputFormat(String outputFormat)**

Sets the data format of the output data that is returned by the cryptographic operation. Uses the specified format when encoding the data.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| outputFormatString | String  | Format of the output data. Valid values:
  
  • FORMATTED: Formatted to the Key Management Framework (KMF) specifications.
  
  • KMFBASE64: Base64 encoded.
  
  • KMFNONE: No decoding. Only supported for MAC_VERIFICATION and SIGNATURE_VERIFICATION.
  
  Default if this method is not called: Value determined by the operation specified when the
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KMFCryptoOperation object was instantiated. For more information, see KMFCryptoOperation - KMFCryptoOperation(String cryptoModuleName, String operationName).</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example uses `withOutputFormat()` to set the output format of the decryption to KMFNone (default is KMFBase64.)

```java
var op = new sn_kmf_ns.KMFCryptoOperation("global.sj_cm","SYMMETRIC_DECRYPTION").withOutputFormat("KMFNone");
var clear_data = op.doOperation(<FORMATTED_data>); // Pass in default of FORMATTED data
```

### KMFCryptoOperation - `withOutputType(String outputType)`

Sets the data type for the output data returned after the cryptographic operation is performed.

**Note:** When you instantiate the KMFCryptoOperation object for `MAC_VERIFICATION` or `SIGNATURE_VERIFICATION` operations, you must also call this method, passing boolean, to set the correct output type or an exception is thrown when you execute the operation.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>outputType</td>
<td>String</td>
<td>Type of output data. Not all output types are applicable to all operations. For an unsupported type, an exception is thrown. Valid values (not case-sensitive):</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|      |      | - String: Not valid for MAC_VERIFICATION or SIGNATURE_VERIFICATION operations.  
|      |      | - Boolean: Only valid for MAC_VERIFICATION or SIGNATURE_VERIFICATION operations.  
|      |      | - Payload: Only valid for the ASYMMETRIC_ENCRYPTION operation. Use this output type for EC-IES. |

**Note:** When specifying an output of **Payload**, the output of the `doOperation()` method is a KMFEncryptionPayload object. For more information on the structure of this object, see `withAdditionalInput()`.

Default: Value determined by the operation, specified when the KMFCryptoOperation object was instantiated. For more information, see `KMFCryptoOperation(String cryptoModuleName, String(operationName))`.

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example uses `withOutputType()` to set the output type for MAC_VERIFICATION to Boolean.

```javascript
var op = new sn_kmf_ns.KMFCryptoOperation("global.sj_cm","MAC_VERIFICATION")
   .withOutputType("Boolean").withAdditionalInput(<MAC>);
var value = GlideStringUtil.base64Encode("Text to sign"); // Default input type is KMFBase64
var result = op.doOperation(String(value));
```

**KMFCryptoOperation - withSysId(String sysId)**

Sets the sys_id of the key to wrap on the KMFCryptoOperation object. Applicable to symmetric and asymmetric wrapping of keys.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysId</td>
<td>String</td>
<td>Sys_id of the key to wrap. Located in the Module Key [sys_kmf_module_key] table.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example uses `withSysId()` to define the key to wrap.

```javascript
var op = new sn_kmf_ns.KMFCryptoOperation("global.sj_cm","SYMMETRIC_WRAPPING").withSysId("0d06ce525b231010f86db341d81c777");
var wrappedKey = operation.doOperation(); // No need to pass data when using withSysId()
```

### Line - Scoped, Global

Creates a Line object using methods to draw a line in a PDF.

This API is part of the ServiceNow PDF Generation Utilities plugin (com.snc.apppdfgenerator) and is provided within the sn_pdfgeneratorutils namespace. The plugin is activated by default.

This API is a component used with the Document API to generate a PDF.

#### Line - Line()

Instantiates a new `Line` object.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

The following examples shows how to create a `Line` object.

```javascript
var line = new sn_pdfgeneratorutils.Line();
```

#### Line - drawLine(Document document, Number pageNo, Number xPos, Number yPos, Number width, Number lineWidth)

Places a line on a document page.
The following example shows how to create a line at the lower margin of a document page. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var num = 1;
var xpos = 100;
var ypos = 100;
var width = 450;
var linewidth = 2.5;

document.addNewPage();

var line = new sn_pdfgeneratorutils.Line();

line.drawLine(document, num, xpos, ypos, width, linewidth);

document.saveAsAttachment("incident", "<sys_id>", "line.pdf");
```
Line – setColor(Color color)

Sets the color of a line.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>Color</td>
<td>Line color.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to create a line and set its color in a document. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var num = 1;
var xpos = 100;
var ypos = 100;
var width = 450;
var linewidth = 2.5;

var color = new sn_pdfgeneratorutils.Color([0.5,0.5,0.8]);   //given as array of RGB values;
document.addNewPage();
var line = new sn_pdfgeneratorutils.Line();
line.setColor(color);
line.drawLine(document, num, xpos, ypos, width, linewidth);
document.saveAsAttachment("incident", "<sys_id>", "lineWithColor.pdf");
```

LinterCheckAstNode API - Scoped, Global

Provides methods for getting abstract syntax tree (AST) node details in linter checks.
This API is included with the Instance Scan (com.glide.instance_scan) plugin. For more information, see Instance Scan.

Use the methods in this class to run linter checks on AST node types by adding code to the Script field in the Linter Check form. For more information, see Advanced linter check scripts.

Create a linter check to identify any issues in a script. When a linter check is run on a record, an abstract syntax tree for its code is generated. You can use the abstract syntax tree to analyze issues with the code.

Access methods in this API using the Instance Scan engine.rootNode object.

**LinterCheckAstNode - getNameIdentifier()**

Retrieves the string value of a name node type. A name node represents a simple identifier that is not a keyword, such as a function or variable name.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

The following example uses the Findings API incrementWithNode() method in a linter check. Use this method in the Script field of the Linter Check form.

```javascript
(function(engine) {
    engine.rootNode.visit(function(node) {
        if (node.getTypeName() === "NAME" &&
            node.getNameIdentifier() === "soughtFunction" &&
            node.getParent().getTypeName() === "CALL") {
            engine.finding.incrementWithNode(node);
        }
    });
})(engine);
```

**LinterCheckAstNode - getParent()**

Gets the parent node object of the accessed node.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Parent node object of the accessed node.</td>
</tr>
</tbody>
</table>

The following example uses the Findings API `incrementWithNode()` method in a linter check. Use this method in the Script field of the Linter Check form.

```javascript
(function(engine) {
    engine.rootNode.visit(function(node) {
        if (node.getTypeName() === "NAME" &&
            node.getNameIdentifier() === "soughtFunction" &&
            node.getParent().getTypeName() === "CALL") {
            engine.finding.incrementWithNode(node);
        }
    });
})(engine);
```

### LinterCheckAstNode - getTypeName()

Gets the type of the accessed node.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Type name of the accessed node. For example, a function call in the source is tokenized as a node with the type name of CALL. See Token class for all node type names.</td>
</tr>
</tbody>
</table>

The following example uses the Findings API `incrementWithNode()` method in a linter check. Use this method in the Script field of the Linter Check form.
LinterCheckAstNode - visit(Function callbackFunction)

Accesses each node in the subtree starting from this node and executes a given callback function on each node.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callbackFunction</td>
<td>Function</td>
<td>Callback function to be executed on each node in the subtree of this node. This callback function takes a node as a parameter which is the node to be visited.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example uses the Findings API incrementWithNode() method in a linter check. Use this method in the Script field of the Linter Check form.

```javascript
(function(engine) {
    engine.rootNode.visit(function(node) {
        if (node.getTypeName() === "NAME" &&
            node.getNameIdentifier() === "soughtFunction" &&
            node.getParent().getTypeName() === "CALL") {
            engine.finding.incrementWithNode(node);
        }
    });
})(engine);
```
Logger - Global

Inbound email actions can use this JavaScript function to append messages to the email log.

The Logger class provides methods that add standard, warning, and error messages to the email log. The added message has its source set to email.<Sys ID of incoming email>.

Logger - log(String msg)

Appends the specified message to the email log.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Text to append to the email log. These should be information type messages.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

logger.log("Some information");

Logger - logError(String msg)

Appends the specified error message to the email log file.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Error message to append to the email log</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
logger.logError("Some error");

**Logger - logWarning(String msg)**

Appends the specified warning message to the email log file.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Warning message to append to the email log</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

logger.logWarning("Some warning");

**LSOFParser - Global**

Parses the output of the `lsof` command: `lsof -i4TCP -n -P -F pcnfT`

Use this API with a discovery script when you need to parse `lsof` output.

**LSOFParser - error(String msg)**

Generates the specified error message.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>The error message</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**LSOFParser - initFileDescriptor()**

Initializes the file descriptor process.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**LSOFParser - initProcess()**

Initializes the parser process.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**LSOFParser - on_p(String line)**

Sets the current PID.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>line</td>
<td>String</td>
<td>The PID</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
LSOFParser - on_c(String line)
Sets the current command.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>line</td>
<td>String</td>
<td>The command</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

LSOFParser - on_f(String line)
Sets the current file descriptor.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>line</td>
<td>String</td>
<td>The file descriptor</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

LSOFParser - on_n(String line)
Sets the current address.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>line</td>
<td>String</td>
<td>The address</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**LSOFParser - on_T(String line)**

Sets the current state.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>line</td>
<td>String</td>
<td>The state</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**LSOFParser - on_endFileDescriptor()**

Closes the file descriptor process.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**LSOFParser - on_endProcess()**

Closes the parser process.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

LSOFParser - parse(String lsofOutput)

Parses the specified `lsof` output.

Results are available in connections and listeners arrays, and errors are recorded in `errorString()` and `isValid()`.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lsofOutput</td>
<td>String</td>
<td>The <code>lsof</code> output</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Messaging - Scoped

Perform actions in a third-party messaging application.

This class requires the Messaging Notification plugin (com.glide.notification.messaging) and an integration with a third-party messaging application such as Slack or Teams. Use these methods in an action script in the Message Actions [messaging_observer_handler] table.

Use the `sn_notification` namespace to access the Messaging API.
Scoped Messaging - send(GlideRecord messagingApplication, String recipient, GlideRecord messagingContent, GlideRecord target)

Sends a custom message to a third-party application in response to a messaging event. For example, you can send a custom welcome message to a Slack channel when the Now Actions application installs.

Use this method in an action script in the Message Actions [messaging_observer_handler] table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>messagingApplication</td>
<td>GlideRecord</td>
<td>Third-party application to send a message to from the Messaging Entities [messaging_application] table.</td>
</tr>
<tr>
<td>recipient</td>
<td>String</td>
<td>Recipient of the message. When the instance receives an inbound message, you can send a response to a Slack channel, Team, or individual user ID found in the inbound payload.</td>
</tr>
<tr>
<td>messagingContent</td>
<td>GlideRecord</td>
<td>Message content to send from the Messaging Contents [messaging_content] table.</td>
</tr>
<tr>
<td>target</td>
<td>GlideRecord</td>
<td>Record used to define dynamic field values in the message. Table must match the Target table field in the Messaging Contents record. If the Messaging Contents record does not use a target table, set the value to null.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

//Send a message to a Teams user
var app = new GlideRecord('messaging_application');
app.get('1f2d26527f4213007f005212bdfa9102');

var content = new GlideRecord('messaging_content');
content.get('17f1f9617320130082999cfd7bf6a706');

sn_notification.Messaging.send(app, '29:1ojsqDg1xuA_jZ70PD12_6E7mn7P6Mo0wK7z0n21b1L-SaNXYVI1cR716qnc11AGvdmhy2-kXh76IEVpUHXdz3w', content, null);

//Send a message to a Slack user
var app = new GlideRecord('messaging_application');
app.get('5d2e38c07f6113007f005212bdafa9160');

var content = new GlideRecord('messaging_content');
content.get('69c48ba77310130082999cfd7bf6a7af');

sn_notification.Messaging.send(app, 'U8P706QFQ', content, null);

//Send a message to a Slack channel
var app = new GlideRecord('messaging_application');
app.get('5d2e38c07f6113007f005212bdafa9160');

var content = new GlideRecord('messaging_content');
content.get('69c48ba77310130082999cfd7bf6a7af');

sn_notification.Messaging.send(app, 'CA6232N65', content, null);

**MIDServer - Global**

Encapsulates the notion of a MID server.

Use these methods in server scripts to manage a MID server using Ajax.

**MIDServer - getByName(String name)**

Returns the specified MID server.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>GlideRecord</td>
</tr>
</tbody>
</table>
**MIDServer - getForPing(DiscoverySchedule schedule, DiscoveryRange range)**

Returns the MID Server with the specified schedule and range set.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>schedule</td>
<td>DiscoverySchedule</td>
<td>The discovery schedule</td>
</tr>
<tr>
<td>range</td>
<td>DiscoveryRange</td>
<td>The discovery range set</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The MID server</td>
</tr>
</tbody>
</table>

**MIDServer - getDefault(DiscoverySchedule schedule)**

Returns the MID server associated with the specified schedule (the MID server with the same name). If there is no associated MID server, gets the next MID server in the ecc_agent table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>schedule</td>
<td>DiscoverySchedule</td>
<td>The discovery schedule</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The MID server</td>
</tr>
</tbody>
</table>

**MIDServer - hostname**

The name of the MID server's host.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hostname</td>
<td>String</td>
<td>The name of the MID server's host.</td>
</tr>
</tbody>
</table>
**MIDServer - hostOS**
The operating system of the MID server's host.

<table>
<thead>
<tr>
<th>Field</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>hostOS</td>
<td>String</td>
<td>The operating system of the MID server's host.</td>
</tr>
</tbody>
</table>

**MIDServer - ip**
The IP address of the MID server's host.

<table>
<thead>
<tr>
<th>Field</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ip</td>
<td>String</td>
<td>The IP address of the MID server's host.</td>
</tr>
</tbody>
</table>

**MIDServer - name**
Name of the MID server.

<table>
<thead>
<tr>
<th>Field</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>name</td>
<td>String</td>
<td>Name of the MID server</td>
</tr>
</tbody>
</table>

**MIDServer - network**
The network containing the MID server's host.

<table>
<thead>
<tr>
<th>Field</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>network</td>
<td>String</td>
<td>The network containing the MID server's host. For example, &quot;10.10.10.0/24&quot; or &quot;10.10.10.0/255.255.255.0&quot;.</td>
</tr>
</tbody>
</table>

**MIDServer - routerIP**
The IP address of the MID server host's default router.
Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>routerIP</td>
<td>String</td>
<td>The IP address of the MID server host’s default router.</td>
</tr>
</tbody>
</table>

MIDServer - status

The MID server's status ("Up" or "Down").

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>String</td>
<td>The MID server's status (&quot;Up&quot; or &quot;Down&quot;).</td>
</tr>
</tbody>
</table>

MIDServer - sysID

The sys_id of the MID server record.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysID</td>
<td>String</td>
<td>The sys_id of the MID server record.</td>
</tr>
</tbody>
</table>

MIDServer - url

The URL the MID server uses to contact the instance.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>The URL the MID server uses to contact the instance.</td>
</tr>
</tbody>
</table>

MIDServer - version

Version of the MID server (WAR name).
### Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Version of the MID server (WAR name).</td>
</tr>
</tbody>
</table>

**MIDServer - windowsDomain**

Windows domain of the MID server’s host (if it is a Windows machine).

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>windowsDomain</td>
<td>String</td>
<td>The Windows domain of the MID server's host.</td>
</tr>
</tbody>
</table>

**MIDServerAjax - Global**

Provides AJAX functionality for sending a test probe to the MID server.

Use in server scripts to test a MID server using AJAX.

#### MIDServerAjax - ajaxFunction_testProbe()

Sends a test probe to the MID server.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Contains the agent name, test probe ID, topic, name, and source.</td>
</tr>
</tbody>
</table>

```javascript
var msaj = new MIDServerAjax();
msaj.ajaxFunction_testProbe();
```

**MIDServerAjax - MIDServerAjax()**

Creates an instance of MIDServerAjax.
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**MIDServerCluster - Global**

Finds the MID server clusters for a given MID server and reassigns the jobs if necessary.

If the MID server agent is up, this class gathers all MID servers (including the original agent) in the load balance clusters that the original agent is part of. If the MID server agent is down and the cluster is load balanced, the class gathers the other agents. If the server agent is down, and the cluster is failover, it gathers the next failover agent. The class then randomly returns the MID server in the final list of MID servers.

Use in server scripts to get MID server cluster information.

**MIDServerCluster - clusterExists()**

Determines if a cluster exists.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if a cluster exists; otherwise, false.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var midgr = new GlideRecord("ecc_agent");
midgr.addQuery("name", "allwinmid");
midgr.query();
midgr.next();

var mscl = new MIDServerCluster(midgr);
gs.print(mscl.clusterExists());

Output: true
```
**MIDServerCluster - getClusterAgent()**

Gets the cluster agent.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The agent name</td>
</tr>
</tbody>
</table>

// Assuming "mid1" and "mid2" are in a cluster
var midgr = new GlideRecord("ecc_agent");
midgr.addQuery("name", "mid1");
midgr.query();
midgr.next();

var mscl = new MIDServerCluster(midgr);
gs.print(mscl.getClusterAgent());

Output: mid2

**MIDServerCluster - MIDServerCluster(GlideRecord agent)**

Brief description of the method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent</td>
<td>GlideRecord</td>
<td>The MID server agent</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
**MIDServerFileSync - Global**

Checks to see if there are any changes to attachments on tables that are instances of MID server synchronized files (ecc_agent_sync_file) and, if so, notifies the MID servers of a change.

The tables that extend the sync tables are MID Server JAR file (ecc_agent_jar) and MID Server MIB File (ecc_agent_mib).

Use in server scripts to notify MID servers.

### MIDServerFileSync - MIDServerFileSync()

Creates an instance of MIDServerFileSync.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### MIDServerFileSync - notifyMIDServers(String table)

Checks for any changes to attachments on the specified table that are instances of MID server synchronized files and, if so, notifies the MID servers of a change.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>The table to check</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var msfs = new MIDServerFileSync();
msfs.notifyMIDServers('ex.table.name');
```

**MIDServerFinder - Global**

Finds a list of MID servers for given IP addresses.

Use in server scripts to get the server list.
MIDServerFinder - getMIDServers()

Gets the names of MID servers available for a given range and capability.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of MID server names. If no MID servers are available, returns an empty array.</td>
</tr>
</tbody>
</table>

```javascript
var msf = new MIDServerFinder();
msf.setRanges('10.10.10.1-10.10.11.254');
var msnames = msf.getMIDServers();

for(var i=0; i<msnames.length; i++) {
    gs.print(msnames[i]);
}
```

MIDServerFinder - getMIDServersBySysId()

Gets the sys_ids of MID servers available for a given range and capability.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of MID server sys_ids. If no MID servers are available, returns an empty array.</td>
</tr>
</tbody>
</table>

```javascript
var msf = new MIDServerFinder();
msf.setRanges('10.10.10.1-10.10.11.254');
```
var msids = msf.getMIDServersBySysId();
for(var i=0; i<msids.length; i++) {
    gs.print(msids[i]);
};

MIDServerFinder - getStatusMessage()
Gets the state of the finder operation.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The status message</td>
</tr>
</tbody>
</table>

var msf = new MIDServerFinder();
gs.print(msf.getStatusMessage());

MIDServerFinder - MIDServerFinder()
Creates an instance of MIDServerFinder.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
**MIDServerFinder - setActive(Boolean flag)**

Sets whether to look for active or inactive MID servers. By default, searches are for active MID servers unless inactive is specified by this method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flag</td>
<td>Boolean</td>
<td>If true, look for inactive MID servers. If false, do not look for inactive MID servers.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var msf = new MIDServerFinder();
msf.setActive('true');
```

**MIDServerFinder - setCapabilities(Array capabilities)**

Sets the technologies for which to look.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>capabilities</td>
<td>Array</td>
<td>Contains a list of capabilities, for example - capabilities = [&quot;ssh&quot;,&quot;wmi&quot;,&quot;snmp&quot;,&quot;os_domain&quot;:&quot;disco&quot;],{&quot;phase&quot;:1} ]</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var msf = new MIDServerFinder();
msf.setRanges('10.10.10.1-10.10.11.254');
var capab = ["ssh","wmi","snmp"];
msf.setCapabilities(capab);
```
**MIDServerFinder - setDebug(Boolean onOrOff)**

Turns debugging on or off.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOrOff</td>
<td>Boolean</td>
<td>True to turn on debugging; false to turn debugging off.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var msf = new MIDServerFinder();
msf.setDebug('true');
```

**MIDServerFinder - setRanges(String ranges)**

Sets the range of IP addresses for which to look.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ranges</td>
<td>String</td>
<td>A comma-separated list in one of these formats.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IP addresses (10.10.10.1, 10.10.10.2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IP networks (10.10.10.0/23)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IP ranges (10.10.1-10.10.11.254)</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var msf = new MIDServerFinder();
msf.setRanges('10.10.10.1-10.10.11.254');
```
MidServerFinder - setRangesByIPOrHostname(String ipOrHostname)

Determines if the input is a single IP or a hostname, and passes the IP or multiple IPs to the setRanges() method.

If the value of ipOrHostname is an IP address, it is passed into the setRanges() method. If the value is a hostname, we look up the DNS table (cmdb_ip_address_dns_name) to try resolving the hostname. The result, either an IP or multiple IPs, is then passed into the setRanges() method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipOrHostname</td>
<td>String</td>
<td>The IP address, IP range, or host name.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var msf = new MidServerFinder();
msf.setRangesByIPOrHostname('10.10.10.1-10.10.11.254');
```

MidServerManage - Global

Allows you to manage a MID server, by sending messages to start, stop, upgrade, get logs, send test probes, and send system messages.

Use in server scripts to send messages to a MID server.

MidServerManage - grab_logs(String agentName, String logs)

Gets the logs for the specified MID server.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentName</td>
<td>String</td>
<td>The MID server name</td>
</tr>
<tr>
<td>logs</td>
<td>String</td>
<td>The type of log to get (Optional)</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The log message</td>
</tr>
</tbody>
</table>

```
var msm = new MIDServerManage();
msm.grab_logs('serv1');
```

Output: Background message, type:info, message: Grabbing MID Server Logs

**MIDServerManage - MIDServerManage()**

Creates an instance of MIDServerManage.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**MIDServerManage - restart(String agentName)**

Restarts the specified MID server.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentName</td>
<td>String</td>
<td>The MID server name</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
var msm = new MIDServerManage();
msm.restart('serv1');
```

**MIDServerManage - stop(String agentName)**

Stops the specified MID server.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentName</td>
<td>String</td>
<td>The MID server name</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var msm = new MIDServerManage();
msm.stop('serv1');
```

#### MIDServerManage - system_msg(String agentName, String cmd, String name)

Creates a system message for the specified MID server.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentName</td>
<td>String</td>
<td>The MID server name</td>
</tr>
<tr>
<td>cmd</td>
<td>String</td>
<td>The value of the probe source field</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the probe</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var msm = new MIDServerManage();
msm.test_probe('mid.server.serv1', 'command', 'probe_id');
```

#### MIDServerManage - text_probe(String agentName, String probeID, String topic, String ename, String source)

Sends a test probe for the specified MID server.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentName</td>
<td>String</td>
<td>The MID server name</td>
</tr>
<tr>
<td>probeId</td>
<td>String</td>
<td>The ID of the probe to send</td>
</tr>
<tr>
<td>topic</td>
<td>String</td>
<td>The information for the topic field</td>
</tr>
<tr>
<td>ename</td>
<td>String</td>
<td>The name for the probe</td>
</tr>
<tr>
<td>source</td>
<td>String</td>
<td>The information for the source field</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The probe for the specified MID server</td>
</tr>
</tbody>
</table>

```javascript
var msm = new MIDServerManage();
msm.test_probe('mid.server.serv1', 'probe_id', 'topic text', 'name', 'source text');
```

### MIDServerManage - upgrade(String agentName)

Upgrades the specified MID server.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentName</td>
<td>String</td>
<td>The MID server name</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var msm = new MIDServerManage();
msm.upgrade('serv1');
```

### MIDServerSelector - Global

Finds a MID server based on capabilities and target IP address.

Use in server scripts to find a MID server with specific capabilities.
**MIDServerSelector - findAgent(String target)**

Finds the MID server based on the capabilities and target IP address.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>String</td>
<td>The target IP address</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var mss = new MIDServerSelector();
var capab = ["ssh","wmi","snmp"];
mss.setCapabilities(capab);
mss.findAgent('100.101.10.10');
```

**MIDServerSelector - getError()**

Gets the error messages for the current MID server.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var mss = new MIDServerSelector();
var capab = ["ssh","wmi","snmp"];
mss.setCapabilities(capab);
mss.findAgent('100.101.10.10');
mss.getError();
```
**MIDServerSelector - getWarning()**

Gets the warning messages for the current MID server.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var mss = new MIDServerSelector();
var capab = ["ssh","wmi","snmp"];
mss.setCapabilities(capab);
mss.findAgent('100.101.10.10');
mss.getWarning();
```

**MIDServerSelector - MIDServerSelector()**

Creates an instance of MIDServerSelector.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**MIDServerSelector - setCapabilities(Array capabilities)**

Sets the technologies for which to look.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| capabilities | Array     | A list of capabilities, for example - capabilities = \[
|          |            | "ssh","wmi","snmp",{"os_domain":"disco"},{"phase":1}\].               |

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var mss = new MIDServerSelector();
var capab = ["ssh","wmi","snmp"];
mss.setCapabilities(capab);
```

**MLPredictor - Global**

Provides utility methods for Predictive Intelligence predictions.

⚠️ **Note:** This API has been deprecated and is intended to be removed in a future release. Refer to Using ML APIs for the most recent guidelines.

The MLPredictor API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the sn_ml namespace.

This class contains all methods necessary to get prediction results from input data.

**MLPredictor - applyPrediction(GlideRecord now_GR, Array solutions)**

Sets predicted values from an array of specified solutions to a specified record.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>The record on which to apply the array of predicted solutions.</td>
</tr>
<tr>
<td>solutions</td>
<td>Array</td>
<td>Specified solution objects associated with the GlideRecord.</td>
</tr>
</tbody>
</table>
The following example demonstrates using the applyPrediction() method in a business rule.

```javascript
(function executeRule(current, previous /*null when async*/) {
    var mlPredictor = new MLPredictor();
    // Get the list of active solutions for the glide record table
    var solutions = mlPredictor.findActiveSolutionsForRecord(current);
    // Run prediction and apply predicted value to the record
    mlPredictor.applyPrediction(current, solution);
})(current, previous);
```

MLPredictor - applyPredictionForSolution(GlideRecord now_GR, Object solution)
Applies a predicted value from a specified classification solution to the specified GlideRecord.

For each solution in the GlideRecord, call this method to predict the results and set the field value on the incident to those results.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>GlideRecord object containing values on which to run a prediction and apply the results.</td>
</tr>
<tr>
<td>solution</td>
<td>Object</td>
<td>Classification solution object to be executed.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True upon prediction success, error otherwise.</td>
</tr>
</tbody>
</table>

To use this template, copy the Incident Based Prediction (Template) business rule, make the new record Active, and follow the commented instructions to initialize the solutionNames variable.
function executeRule(current, previous /*null when async*/) {
    var solutionNames = ["solution1", "solution2", ...];

    /* For domain separation (MSP) use case */
    // var solutionNames;
    // if (current.sys_domain == "A")
    //    solutionNames = ["solution_A1", "solution_A2", ...];
    // else if (current.sys_domain == "B")
    //    solutionNames = ["solution_B1", "solution_B2", ...];
    // else
    //    ...

    var predictor = new MLPredictor();
    var info = "";
    solutionNames.forEach(function(solutionName) {
        var solution = predictor.findActiveSolution(solutionName);
        if (!solution)
            return;
        /* The next line of code does the prediction and updates the current record. */
predictor.applyPredictionForSolution(current, solution);

        /* If no prediction is done, do not build the prediction info message. */
        if (!predictor.applyPredictionForSolution(current, solution))
            return;

        /* If user doesn't have 'itil' role, we don't build prediction info message. */
        if (!gs.hasRole('itil'))
            return;

        /* Building prediction info message */
        var fieldName = solution.getPredictedField();
        var fieldLabel = current.getElement(fieldName).getED().getLabel();
        var predictedDisplayValue = current.getDisplayValue(fieldName);
        var msg = gs.getMessage("Predicted {0} for {1}.", [predictedDisplayValue, fieldLabel]);
        if (info.length > 0)
            info += " ";
        info += msg;
    });
    /* Print out prediction info message on screen. */
    if (info.length > 0) {
        var incidentUrl = "<a href=" + current.getLink() + ">
" + current.number + ":</a>";
        gs.addInfoMessage(incidentUrl + " " + info);
}
**MLPredictor - findActiveSolution(String solutionName)**

Gets the solution object.

This method only returns the solutions if the ml_solution definition and solution are active.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>solutionName</td>
<td>String</td>
<td>Name of the ml_solution record.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Solution object for the specified solutionName if the ml_solution definition and solution is active, null otherwise.</td>
</tr>
</tbody>
</table>

This example takes a hard-coded value and passes it to a specified Machine Learning model. You can use the outcome and confidence variables to set values or other business logic.

```javascript
var solutionName = 'ml_incident_assignment';
var shortDescriptionValue = "Unable to connect!"
var input = {
    short_description : shortDescriptionValue
};
var MLP = new MLPredictor();
var solution = MLP.findActiveSolution(solutionName);
var predictedOutcome = solution.predictText(input);
var outcome = predictedOutcome.predictedValue();
var confidence = predictedOutcome.confidence();
gs.info("Predicted value: " + outcome)
gs.info("Confidence: " + confidence)
```

**Output:**
**MLPredictor - findActiveSolutionsForRecord(GlideRecord now_GR)**

Gets active solutions for a table in a specified GlideRecord.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>GlideRecord from which to collect active solution objects.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array of active solution objects associated with the table that the specified record is for.</td>
</tr>
</tbody>
</table>

This script passes a GlideRecord (such as an incident), and processes all solutions for the GlideRecord, returning the values for each.

```javascript
/* This is only to get a hard-coded GR */
var current = new GlideRecord('incident');
current.get('965c9e5347c12200e0ef563dbb9a7156');
var predictor = new MLPredictor();
var solutions = predictor.findActiveSolutionsForRecord(current);
solutions.forEach(function(solution) {
  var outcome = solution.predict(current);
  /* Use this to set the field to the predicted value in the GlideRecord */
  var fieldName = solution.getPredictedField();
  current[fieldName] = outcome.predictedValue();
  current.update();
});
gs.info("Predicted value: " + outcome.predictedValue())
gs.info("Confidence: " + outcome.confidence())
```
**MLPredictor - getPredictedValue(Object solution, Object outcome)**

Gets the predicted value for a specified solution based on the specified prediction outcome.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>solution</td>
<td>Object</td>
<td>Solution from which to get the predicted value.</td>
</tr>
<tr>
<td>outcome</td>
<td>Object</td>
<td>Prediction outcome results for the specified solution (var outcome = solution.predict(now_GR)).</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Predicted value for specified solution based on the specified outcome of the prediction.</td>
</tr>
<tr>
<td>Null</td>
<td>Returns null if the prediction confidence does not satisfy thresholds.</td>
</tr>
</tbody>
</table>

**MLPredictor - getPredictions(GlideRecord now_GR, Object solution, Object options)**

Gets predictions for a specified solution.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>GlideRecord to be predicted.</td>
</tr>
<tr>
<td>solution</td>
<td>Object</td>
<td>Solution object to be executed.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional JSON object key value pair with the following properties:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• options.top_n: If provided, returns results up to the expected number of predictions, otherwise default is read from the glide.platform_ml.max_num_predictions system property.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• options.apply_threshold: Checks the threshold value (solution threshold for similarity, class level threshold for classification) for the solution and applies it to the result set. Default value is true.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array of predicted outcome objects</td>
</tr>
</tbody>
</table>

The following example calls a specified solution and executes predictions on it.

```javascript
function printOutcomeArr(outcomeArr) {
  gs.print('################## Results ##################
  for (var i=0; i<outcomeArr.length; i++) {
    var outcome = outcomeArr[i];
    gs.print((i+1) + ' : ' + outcome.predictedValue() + ', ' + outcome.predictedValueSysId() + ', ' + outcome.confidence());
  }
}

var solutionName = 'ml_x_anc_global_prop_flip_test';
var predictor = new MLPredictor();
var solution = predictor.findActiveSolution(solutionName);

var now_GR = new GlideRecord('incident');
own_GR.get('1c741bd70b2322007518478d83673af3');

var options = {};
options.top_n = '10';            // top_n is an integer between 1 and 1000
options.apply_threshold = false; // Value can be set to true or false
```
printOutcomeArr(predictor.getPredictions(now_GR, solution, options));

**MLPredictor - isClassificationSolution(Object solution)**

Identifies if a solution object is a classification type.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>solution</td>
<td>Object</td>
<td>Name of the ML solution.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the input solution is a classification type, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var isClassificationSolution = this.isClassificationSolution(solution);

//classification solution each class has different threshold
//therefore needs to get all the results from ml engine
if (applyThreshold && isClassificationSolution) {
    var maxClassificationTopN = 50;
    outcomeArr = solution.predictTopN(now_GR, maxClassificationTopN);
} else {
    outcomeArr = solution.predictTopN(now_GR, topN);
}

if (outcomeArr === null) {
    //instead of returning null returning empty array
    return [];
}
```

**MLPredictor - isSimilaritySolution(Object solution)**

Identifies if a solution object is a similarity type.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>solution</td>
<td>Object</td>
<td>Name of the ML solution; for example, ml_incident_categorization.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the input solution is a similarity type, false otherwise.</td>
</tr>
</tbody>
</table>

MLPredictor - MLPredictor()

Instantiates a new MLPredictor object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to use an MLPredictor object in a business rule to record final values after a prediction.

```javascript
function executeRule(current, previous /*null when async*/) {
  var predictor = new MLPredictor();
  predictor.recordFinalValuesInPredictionResults(current, "On close");
}(current, previous);
```

MLPredictor - outcome.confidence()

Gets the confidence of the predicted value.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The estimated precision of the prediction as a percentage. For example, 53.84615375762915.</td>
</tr>
</tbody>
</table>

```javascript
var MLP = new MLPredictor();
var solution = MLP.findActiveSolution(solutionName);
var predictedOutcome = solution.predictText(input);
var outcome = predictedOutcome.predictedValue();
var confidence = predictedOutcome.confidence();
gs.info("Predicted value: " + outcome)
gs.info("Confidence: " + confidence)
```

**MLPredictor - outcome.predictedValue()**

Gets the predicted value from the MLPredictor outcome object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Predicted value from the Outcome object.</td>
</tr>
</tbody>
</table>

```javascript
var MLP = new MLPredictor();
var solution = MLP.findActiveSolution(solutionName);
var predictedOutcome = solution.predictText(input);
var outcome = predictedOutcome.predictedValue();
var confidence = predictedOutcome.confidence();
gs.info("Predicted value: " + outcome)
gs.info("Confidence: " + confidence)
```

**MLPredictor - outcome.predictedValueSysId()**

Gets the sys_id of the predicted value.
function printOutcomeArr(outcomeArr) {
    gs.print('################## Results ##################
    for (var i=0; i<outcomeArr.length; i++) {
        var outcome = outcomeArr[i];
        gs.print((i+1) + ' : ' + outcome.predictedValue() + ', ' + outcome.predictedValueSysId() + ', ' + outcome.confidence() + ');
    }
}

MLPredictor - recordFinalValuesInPredictionResults(GlideRecord now_GR, String reason)
Sets final prediction result values to a specified GlideRecord with an optionally specified reason.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>GlideRecord on which to set the final prediction result values.</td>
</tr>
<tr>
<td>reason</td>
<td>String</td>
<td>Optional. Reason for applying results.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

In following example, the `recordFinalValuesInPredictionResults()` method is called when the incident is closed.

(function executeRule(current, previous /*null when async */) {
    var predictor = new MLPredictor();
}
MLPredictor - solution.getCapability()

Gets the capability information of a trained solution.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Definition ID and version of the trained solution, error message otherwise</td>
</tr>
</tbody>
</table>

MLPredictor - solution.getName()

Gets the name of the solution used for prediction.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the solution to use for predictions; for example, ml_incident_categorization</td>
</tr>
</tbody>
</table>

MLPredictor - solution.getPredictedField()

Gets the predicted value of a solution.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
/* Get a hard-coded GR */
var current = new GlideRecord('incident');
current.get('965c9e5347c12200e0ef563dbb9a7156');
var predictor = new MLPredictor();
var solutions = predictor.findActiveSolutionsForRecord(current);
solutions.forEach(function(solution) {
    var outcome = solution.predict(current);
    /* Use this to set the field to the predicted value in the GlideRecord */
    var fieldName = solution.getPredictedField();
    current[fieldName] = outcome.predictedValue();
    current.update();
    gs.info("Predicted value: " + outcome.predictedValue());
    gs.info("Confidence: " + outcome.confidence());
});

MLPredictor - solution.getThreshold(String className)

Gets the solution threshold.

The threshold represents a percentage reflecting the minimum prediction accuracy.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>className</td>
<td>String</td>
<td>A specified categorical value of the solution output field</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Value of the threshold represented as a percentage between 0 and 100.</td>
</tr>
</tbody>
</table>

MLPredictor - solution.getVersion()

Gets the version of the active solution.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Version of the active solution</td>
</tr>
</tbody>
</table>

**MLPredictor - solution.isActive()**

Determines if the specified solution is active.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the solution is active, false otherwise</td>
</tr>
</tbody>
</table>

**MLPredictor - solution.predict(GlideRecord now_GR, Object threshold)**

Gets solution prediction results as an Outcome object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>GlideRecord of the solution input table</td>
</tr>
<tr>
<td>threshold</td>
<td>Object</td>
<td>Threshold value (solution level threshold for similarity, class level threshold for classification)</td>
</tr>
</tbody>
</table>
**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Prediction outcome result of the specified solution (var outcome = solution.predict(now_GR))</td>
</tr>
</tbody>
</table>

solutions.forEach(function(solution) {
    var outcome = solution.predict(current);
    /* Use this to set the field to the predicted value in the GlideRecord
    var fieldName = solution.getPredictedField();
    current[fieldName] = outcome.predictedValue();
    current.update();
    */
    gs.info("Predicted value: " + outcome.predictedValue())
    gs.info("Confidence: " + outcome.confidence())
});

**MLPredictor - solution.predictTopN(GlideRecord now_GR, Object topN)**

Returns a list of outcome objects up to the expected number of predictions. Maximum number 1000 predictions.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>GlideRecord of the solution</td>
</tr>
<tr>
<td>topN</td>
<td>Object</td>
<td>Expected number of predictions, any number over 1000 returns 1000 results</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of outcome objects in an array including GlideRecord, threshold, system ID, and expected number of predictions (topN object)</td>
</tr>
</tbody>
</table>

var isClassificationSolution = this.isClassificationSolution(solution);

//classification solution each class has different threshold
//therefore needs to get all the results from ml engine
if (applyThreshold && isClassificationSolution) {
    var maxClassificationTopN = 50;
}
outcomeArr = solution.predictTopN(now_GR, maxClassificationTopN);
}
else {
outcomeArr = solution.predictTopN(now_GR, topN);
}

if (outcomeArr === null) {
    // instead of returning null returning empty array
    return [];
}

**MLSolution - Global**

Provides methods for handling Predictive Intelligence predictions and retrieving solution objects for all capabilities: similarity, classification, and clustering).

The **MLSolution** API provides methods commonly used among all solution capabilities and methods unique to classification and clustering solutions.

- **Common methods used for all solution objects:**
  - `getCapability()`
  - `getVersion()`
  - `isActive()`
  - `predict()`

- **Methods only used for classification and regression solutions:**
  - `applyPrediction()`

- **Methods only used for clustering solutions:**
  - `getClusterAssignments()`
  - `getClusterForRecord()`
  - `getClusterInfo()`

The **MLSolution** API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the **sn_ml** namespace.

The methods in this object are instantiated using **MLSolutionFactory**.

**Related information**

- **MLSolutionUtil**
- REST API: Get predictions for multiple solutions
**MLSolution - applyPrediction(GlideRecord now_GR)**

Gets the prediction result for a classification solution and applies it to the input GlideRecord if the confidence value is higher than the threshold.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>GlideRecord object containing values on which to run a prediction and apply the results.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the prediction was applied:</td>
</tr>
<tr>
<td></td>
<td>• true: Prediction applied</td>
</tr>
<tr>
<td></td>
<td>• false: Prediction rejected</td>
</tr>
</tbody>
</table>

```javascript
var mlSolution = sn_ml.MLSolutionFactory.getSolution("ml_incident_categorization");
var inputGR = new GlideRecord("incident");
inputGR.get("0ef47232db801300864adfea5e961912");
mlSolution.applyPrediction(inputGR);
```

**MLSolution - getCapability()**

Gets the capability information of a trained solution.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Type of trained solution. Possible values:</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Similarity</td>
<td></td>
</tr>
<tr>
<td>• Classification</td>
<td></td>
</tr>
<tr>
<td>• Clustering</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var mlSolution = sn_ml.MLSolutionFactory.getSolution("ml_x_global_clustering");

// configure optional parameters
var options = {};
options.group_by = 'network';
options.cluster_id = 1;
options.top_n_per_cluster = 3;

if (mlSolution.getCapability() == 'clustering') {
    var results = mlSolution.getClusterAssignments(options);
    // pretty print JSON results
    gs.print(JSON.stringify(JSON.parse(results), null, 2));
}
```

### MLSolution - getClusterAssignments(Object options)

Gets assignments for a clustering solution.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional. Values to use to narrow down the returned results by group and level within a clustering solution. Default: Return cluster memberships for all clusters.</td>
</tr>
<tr>
<td>options.group_by</td>
<td>String</td>
<td>Optional. Identifies the segmentation field for which to retrieve cluster memberships, for example, assignment_group. This field provides the same grouping as options provided in the <strong>Use Group By</strong> check box in the <strong>Clustering Definition</strong> form. The information provided varies based on the</td>
</tr>
</tbody>
</table>
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>table selected in the Table field. For more information, see Create and train a clustering solution.</td>
</tr>
<tr>
<td>options.top_n_per_cluster</td>
<td>Number</td>
<td>Number of top results to receive for each cluster.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Object    | JSON array containing cluster information in increasing order by cluster_id:  
- cluster_id: String. Unique cluster number within a solution of clusters.  
- rec_display_id: String. Record type and number.  
- rec_sys_id: String. Record sys_id.  
- group_by: If grouped, name of the segmentation field associated with this cluster. |

```javascript
var mlSolution = sn_ml.MLSolutionFactory.getSolution("ml_x_global_clustering");

// configure optional parameters
var options = {};
options.group_by = 'network';
options.cluster_id = '1';
// returns top 3 results per cluster
options.top_n_per_cluster = 3;

var results = mlSolution.getClusterAssignments(options);
```

Output:

```json
[  
{"cluster_id":"1","rec_display_id":"Incident: INC0014483","rec_sys_id":"04e33e7adb401300864adfe4e961900"},
"group_by":"network"},
```
MLSolution - getClusterForRecord(GlideRecord now_GR)

Gets the cluster information for a clustering solution.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>Name of the input GlideRecord.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The cluster_id from the Cluster Summary [ml_cluster_summary] table if the record belongs to a cluster. Empty string returns if the record does not belong to a cluster.</td>
</tr>
</tbody>
</table>

```javascript
var mlSolution = sn_ml.MLSolutionFactory.getSolution("solution_name");
var now_GR = new GlideRecord('incident');
if (mlSolution.getCapability() == 'clustering') {
  var clusterId = mlSolution.getClusterForRecord(now_GR);
}
```

MLSolution - getClusterInfo(Object options)

Gets information for a specified clustering solution.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional. Narrows down the returned results by group and level within a clustering solution. Default: Return cluster memberships for all clusters.</td>
</tr>
<tr>
<td>options.group</td>
<td>String</td>
<td>Optional. Identifies the segmentation field for which to retrieve cluster memberships, for example, assignment_group.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>This field provides the same grouping as options provided in the <strong>Use Group By</strong> check box in the <strong>Clustering Definition</strong> form. The information provided varies based on the table selected in the <strong>Table</strong> field. For more information, see Create and train a clustering solution.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JSON array containing cluster information in increasing order by cluster_id:</td>
</tr>
<tr>
<td></td>
<td>- cluster_id: String. Unique cluster number within a solution of clusters.</td>
</tr>
<tr>
<td></td>
<td>- cluster_quality: String. Number from 0 to 100. Higher numbers indicate higher cluster density.</td>
</tr>
<tr>
<td></td>
<td>- cluster_size: String. Number of records in a cluster.</td>
</tr>
<tr>
<td></td>
<td>- group_by: If grouped, name of the segmentation field associated with this cluster.</td>
</tr>
<tr>
<td></td>
<td>- cluster_concept: String. Set of words that describe the cluster in descending order of frequency.</td>
</tr>
</tbody>
</table>

```javascript
var mlSolution = sn_ml.MLSolutionFactory.getSolution("ml_x_global_clustering");

// configure optional parameters
var options = {};
options.group_by = 'network';
options.cluster_id = 1;

var results = mlSolution.getClusterInfo(options);
```

Output:
MLSolution - getVersion()

Gets the version of the active solution.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Version of the active solution.</td>
</tr>
</tbody>
</table>

```javascript
var mlSolution = sn_ml.MLSolutionFactory.getSolution("solution_name");
var solutionVersion = mlSolution.getVersion();
```

MLSolution - isActive()

Determines if a solution is active.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the solution is active.</td>
</tr>
<tr>
<td></td>
<td>• true: Solution active</td>
</tr>
<tr>
<td></td>
<td>• false: Solution inactive</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
var mlSolution = sn_ml.MLSolutionFactory.getSolution("solution_name");

var isActive = mlSolution.isActive();

**MLSolution - predict(Object input, Object options)**

Gets the prediction results from the prediction server given a GlideRecord or an array of key-value pairs.

You can use this method to call prediction with multiple input records because GlideRecord is an iterator.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>Object</td>
<td>GlideRecord or array of JSON objects containing field names and values as key-value pairs.</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional. JSON key-value pair with the following properties:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• top_n: Number. If provided, returns the top results, up to the specified number of predictions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• apply_threshold: Boolean. Checks the threshold value for the solution and applies it to the result set. The threshold value is solution threshold for similarity or class-level threshold for classification. Default value is true.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• custom_results_filter: String. Similarity solutions only. Specifies the allowed set from which results are returned using an encoded query.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JSON object containing the prediction results sorted by sys_id or record_number.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• predictedValue: String.</td>
<td>Value representing the prediction result.</td>
</tr>
<tr>
<td>• predictedSysId: String.</td>
<td>The sys_id of the predicted value. Results can be from any table on which information is being predicted.</td>
</tr>
<tr>
<td>• confidence: Number.</td>
<td>Value of the confidence associated with the prediction. For example, 53.84.</td>
</tr>
<tr>
<td>• threshold: Number.</td>
<td>Value of the configured threshold associated with the prediction.</td>
</tr>
<tr>
<td>• detailedResults: Object.</td>
<td>Similarity solutions only. JSON key-value pair containing details about the matching text indices.</td>
</tr>
</tbody>
</table>

```
var mlSolution = sn_ml.MLSolutionFactory.getSolution("ml_incident_categorization");

// single GlideRecord input
var input = new GlideRecord("incident");
input.get("0ef47232db801300864adfe3e961912");

// configure optional parameters
var options = {};
options.top_n = 3;
options.apply_threshold = false;

var results = mlSolution.predict(input, options);
// pretty print JSON results
gs.print(JSON.stringify(JSON.parse(results), null, 2));
```

```
var mlSolution = sn_ml.MLSolutionFactory.getSolution("ml_incident_categorization");

// multiple GlideRecord input
var input = new GlideRecord("incident");
input.addQuery("sys_created_onONLast
    week@javascript:gs.beginningOfLastWeek()@javascript:gs.endOfLastWeek()");
input.query();

// configure optional parameters
var options = {};
```
options.top_n = 3;
options.apply_threshold = false;

var results = mlSolution.predict(input, options);
// pretty print JSON results
gs.print(JSON.stringify(JSON.parse(results), null, 2));

Output:

```javascript
{
  input_gr_sys_id1: [
    {
      predictedValue : xxx,
      predictedSysId : xx0,
      confidence : xxx,
      threshold : xxx,
      detailedResults : [.....]
    },
    {
      predictedValue : yyy,
      predictedSysId : xx1,
      confidence : xxx,
      threshold : xxx,
      detailedResults : [.....]
    }
  ],
  input_gr_sys_id2 : [
    {
      predictedValue : xxx,
      predictedSysId : xx0,
      confidence : xxx,
      threshold : xxx,
      detailedResults : [.....]
    }
  ],
  ...
}
```

var mlSolution = sn_ml.MLSolutionFactory.getSolution("ml_incident_categorization");

// key-value pairs input
var input = ["short_description":"my email is not working"],
(short_description:"need help with password");

// configure optional parameters
var options = {};
options.top_n = 3;
options.apply_threshold = false;

var results = mlSolution.predict(input, options);
// pretty print JSON results
gs.print(JSON.stringify(JSON.parse(results), null, 2));

**MLSolutionFactory - Global**

Factory class to get an MLSolution scriptable object.

The MLSolutionFactory API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the `sn_ml` namespace. For more information, see Predictive Intelligence.

For usage guidelines, refer to **MLSolutionFactory scriptable objects**.

**Related information**

- **MLSolution**
- **MLSolutionUtil**
- REST API: Get predictions for multiple solutions

**MLSolutionFactory - getSolution(String solutionName, Object options)**

Gets an MLSolution object for a specified solution name.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>solutionName</td>
<td>String</td>
<td>Name of the solution.</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional. options.version: If provided, creates MLSolution instance for provided version of solutionName.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td><strong>MLSolution</strong> object of the specified solution.</td>
</tr>
</tbody>
</table>
// basic usage
var mlSolution = sn_ml.MLSolutionFactory.getSolution("ml_incident_categorization");

// using options.version parameter
var options = {};
options.version = 1;
var mlSolution = sn_ml.MLSolutionFactory.getSolution("ml_incident_categorization", options);

**MLSolutionResult - Global**

Provides methods for managing cluster information and members of a clustering solution. You can embed the results in business logic.

ℹ️ **Note:** This API has been deprecated and is intended to be removed in a future release. Refer to Using ML APIs for the most recent guidelines.

**MLSolutionResult - findActiveSolution(String solutionName)**

Returns the solution object.

This method only returns the solutions if the ml_solution definition and solution are active (that is, trained). For information, see Create and train a clustering solution.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>solutionName</td>
<td>String</td>
<td>Name of the clustering ml_solution record.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Clustering solution object for the specified solutionName if the ml_solution definition and solution is active, null otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var solutionName = 'ml_incident_assignment';
var MLS = new MLSolutionResult();
var solution = MLS.findActiveSolution(solutionName);
```
MLSolutionResult - getClusterAssignments(String solutionName, Object options)

Returns all members of a clustering solution.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>solutionName</td>
<td>String</td>
<td>Name of active cluster solution.</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional. Narrows down results by group and level.</td>
</tr>
<tr>
<td>options.segmentation_field</td>
<td>String</td>
<td>Optional. Identifies the segmentation field. This field provides the same grouping as the options provided via the &quot;Use Group By&quot; check box in the Clustering Solution Definitions table. Information provided varies by table selected in the Table field. For information, see Create and train a clustering solution.</td>
</tr>
<tr>
<td>options.cluster_id</td>
<td>String</td>
<td>Optional. The sys_id from the ml_cluster_summary table.</td>
</tr>
<tr>
<td>options.rec_sys_id</td>
<td>String</td>
<td>Optional. The sys_id from the table record the cluster solution is based on.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array of outcome objects including:</td>
</tr>
<tr>
<td></td>
<td>• segmentation – Field name by which to group data</td>
</tr>
<tr>
<td></td>
<td>• cluster_num – Unique cluster number within a solution of clusters (that is, label)</td>
</tr>
<tr>
<td></td>
<td>• rec_sys_id – The sys_id from the table record that the cluster solution is based on</td>
</tr>
<tr>
<td></td>
<td>• rec_display_id – Name of the record associated with the record sys_id.</td>
</tr>
</tbody>
</table>

The following example shows how to return all cluster members for a solution without setting values for the options object.

```javascript
var solutionName = "<Name_of_Active_Cluster_Solution>";
var solutionResult = new MLSolutionResult();
var outcome_array = solutionResult.getClusterAssignments(solutionName);
for (var i = 0; i < outcome_array.length; i++) {
  // Process each outcome object...
}
```
The following example shows how to return all cluster members for one record using options.rec_sys_id.

```javascript
var now_GR = new GlideRecord('incident');
now_GR.get('sys_id');

var solutionName = "solution_example";
var solutionResult = new MLSolutionResult();
var options = { "rec_sys_id": now_GR.getUniqueValue() };
var outcome_array = solutionResult.getClusterAssignments(solutionName, options);
for (var i = 0; i < outcome_array.length; i++) {
    gs.print(outcome_array [i].segmentation + ' ' + outcome_array [i].cluster_num + ' ' + outcome_array [i].rec_sys_id + ' ' + outcome_array [i].rec_display_id);
}
```

### MLSolutionResult - getClusterInfo(String solutionName, Object options)

Returns all outcome information for a clustering solution.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>solutionName</td>
<td>String</td>
<td>Name of active cluster solution.</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional. Narrows down results. Default: Return cluster memberships.</td>
</tr>
<tr>
<td>options.segmentation_field</td>
<td>String</td>
<td>Optional. Identifies the segmentation field. Information provided varies by table selected in the Table field. For information, see Create and train a clustering solution.</td>
</tr>
<tr>
<td>options.cluster_id</td>
<td>String</td>
<td>Optional. The sys_id from the ml_cluster_summary table.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array of outcome objects including:</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• segmentation – Field name by which to group data</td>
<td></td>
</tr>
<tr>
<td>• cluster_num – Unique cluster number within a solution of clusters (that is, label)</td>
<td></td>
</tr>
<tr>
<td>• total_members – Number of records in cluster (that is, size)</td>
<td></td>
</tr>
<tr>
<td>• cluster_quality – Cluster quality percentile value</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var solutionName = "solution_example";
var solutionResult = new MLSolutionResult();
var outcome_array = solutionResult.getClusterInfo(solutionName);
for (var i = 0; i < outcome_array.length; i++) {
  gs.print(outcome_array[i].segmentation + ' ' + outcome_array[i].cluster_num + ' ' + outcome_array[i].total_members + ' ' + outcome_array[i].cluster_quality);
}
```

**MLSolutionResult - MLSolutionResult()**

Instantiates a new MLSolutionResult object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**MLSolutionUtil - Global**

The **MLSolutionUtil** API provides methods for getting Predictive Intelligence predictions.

The **MLSolutionUtil** API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the **sn_ml** namespace.

For more information, see **Using ML APIs**.

**Related information**

- MLSolution
- MLSolutionFactory
**MLSolutionUtil - MLSolutionUtil()**

Instantiates a new MLSolutionUtil object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
var mlSolutionUtil = new MLSolutionUtil();
```

**MLSolutionUtil - getPredictions(Object input, Array solutionNames, Object options)**

Gets predictions for one or more specified solutions.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>Object</td>
<td>GlideRecord or array of JSON objects as key-value pairs.</td>
</tr>
<tr>
<td>solutionNames</td>
<td>Array</td>
<td>Array of solution names to retrieve predictions from.</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional. JSON object key-value pair with the following properties:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• top_n: Number. If provided, returns the top results, up to the specified number of predictions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• apply_threshold: Boolean. Checks the threshold value for the solution and applies it to the result set. The threshold value is solution threshold for similarity or class-level threshold for classification. Default value is true.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• custom_results_filter: String. Similarity solutions only. Specifies the allowed set from which results are returned using an encoded query.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Array</td>
<td>JSON key-value pair containing the prediction result grouped by solution name and sorted by sys_id or record_number.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• predictedValue: String. Value representing the prediction result.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• predictedSysId: String. The sys_id of the predicted value. Results can be from any table on which information is being predicted.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• confidence: Number. Value of the confidence associated with the prediction. For example, 53.84.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• threshold: Number. Value of the configured threshold associated with the prediction.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• detailedResults: Object. Similarity solutions only. JSON key-value pair containing details about the matching text indices.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var solutionNames = ['solution1', 'solution2'];

var input = new GlideRecord("incident");
input.get("0ef47232db801300864adfe5e961912");

// configure optional parameters
var options = {};
options.top_n = 3;
options.apply_threshold = false;

var mlSolutionUtil = new MLSolutionUtil();
var results = mlSolutionUtil.getPredictions(input, solutionNames, options);

// pretty print JSON results
gs.print(JSON.stringify(JSON.parse(results), null, 2));
```

**Output:**

```javascript
{
  solution1: {
    input_gr_sys_id1: {
      predictedValue : xxx,
      predictedSysId : xx0,
      confidence : xxx,
      threshold : xxx,
      detailedResults : [.....]
    }
  }
```
MobileDeepLinkGenerator - Global

Embed a link to a mobile app list or form applet.

Use this API in a global server-side script to create a link to a list or form applet within these apps:

- Now Mobile
- Mobile Agent app
- Mobile Onboarding
Note:
The Mobile Onboarding is being deprecated!

With the Now Platform Rome release in September 2021, we started phasing out support for the Mobile Onboarding. Customers may no longer activate it, and we are not offering enhancements or non-critical bug fixes. Mobile onboarding features are available in the Now Mobile app for HR Service Delivery.

For example, create a link from an email, push notification, or Virtual Agent conversation that opens a record or list of records in the app. The form or list only includes data that the user has access rights to view.

For an example of this API used in a push notification, see Configure push notification message content.

MobileDeepLinkGenerator - MobileDeepLinkGenerator(String clientType)
Instantiates a MobileDeepLinkGenerator object for a specific mobile app.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clientType</td>
<td>String</td>
<td>Name of the app to create a link to. Options include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• agent: Creates a link to a list or screen in the Mobile Agent app.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• request: Creates a link to a list or screen in the Now Mobile app.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• onboarding: Creates a link to a list or screen in the Mobile Onboarding.</td>
</tr>
</tbody>
</table>

Note:
The Mobile Onboarding is being deprecated!

With the Now Platform Rome release in September 2021, we started phasing out support for the Mobile Onboarding. Customers may no longer activate it, and we are not offering enhancements or non-critical bug fixes. Mobile onboarding features are available in the Now Mobile app for HR Service Delivery.
```javascript
var glide = new global.MobileDeepLinkGenerator('onboarding');

MobileDeepLinkGenerator - getScreenLink(String documentId, Object uiParams)
Create a link to a list applet.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>documentId</td>
<td>String</td>
<td>Sys ID of the applet record in the Applets [sys_sg_screen] table.</td>
</tr>
<tr>
<td>uiParams</td>
<td>Object</td>
<td>Optional. JSON object containing any URL parameters to include in the link.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Determine which URL parameters to include by inspecting the URL you want to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>link to. In general, URL parameters filter records in the list. For example,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>var param = {'&lt;company-sys-id&gt;': 'servicenow'}, where <code>&lt;company-sys-id&gt;</code> is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the Sys ID of a company UI parameter from the UI Parameters [sys_sg_ui_parameter]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>table for a specific mobile screen.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Link to the mobile app list applet.</td>
</tr>
</tbody>
</table>

```javascript
var uiParams = {
    "b250294ab3c12300a0d56ad4c6a8dc8c": "servicenow",
    "7250294ab3c12300a0d56ad4c6a8dc90": "santa clara"
}

var deepLinkGenerator = new global.MobileDeepLinkGenerator("request");
var link = deepLinkGenerator.getScreenLink("a75df1920f2033001befa68ca8767e50", uiParams);

MobileDeepLinkGenerator - getFormScreenLink(String formScreenId, String tableName, String recordSysId)
Create a link to a form applet.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>formScreenId</td>
<td>String</td>
<td>Sys ID of the applet record in the Applets [sys_sg_screen] table.</td>
</tr>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table containing the record in the recordSysId parameter.</td>
</tr>
<tr>
<td>recordSysId</td>
<td>String</td>
<td>Sys ID of the record to open in the applet.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Link to the record in the mobile app form applet.</td>
</tr>
</tbody>
</table>

```javascript
var deepLinkGenerator = new global.MobileDeepLinkGenerator("agent");
var link = deepLinkGenerator.getFormScreenLink("<sys_id>", current.getTableName(),
current.getValue("sys_id");
```

## MonitorMIDServer - Global

Monitors MID servers using the heartbeat probe to check for MID servers that are down, sends heartbeat requests, and kills old heartbeat requests.

Use in a server script to monitor MID servers.

### MonitorMIDServer - killOldRequests()

If there are any heartbeat probe requests that haven’t been processed, cancels them so that they don’t accumulate when a MID server is down for a while.

## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var mmsv = new MonitorMIDServer();
mmsv.killOldRequests();

MonitorMIDServer - markDowners()
Marks any non-responding MID servers as being down, by determining when the most recent heartbeat was sent and finding all non-responding servers.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var mmsv = new MonitorMIDServer();
mmsv.markDowners();

MonitorMIDServer - monitor()
Performs the functions required for a scheduled MID server monitor cycle, including killing old requests, marking the MID servers that are down, and sending heartbeat requests to all servers.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var mmsv = new MonitorMIDServer();
mmsv.monitor();
**MonitorMIDServer - MonitorMIDServer()**

Creates an instance of MonitorMIDServer.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**MonitorMIDServer - sendHeartbeatRequests()**

Sends heartbeat requests to all MID servers and notes when by updating the server status record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var mmsv = new MonitorMIDServer();
mmsv.sendHeartbeatRequests();
```

**NameValuePairs - Global**

Converts between maps and strings representing the name/value pairs in the map.

The string form is: `<name> = <value>, <name> = <value, ... where <name> is a string name, optionally surrounded by double quotes (Microsoft-style), and <value> is a string value, also optionally surrounded by double quotes. The following are all examples of valid name/value strings:

- name = value
- name = "My Value", name = value
- "My Name = this" = "My Value", "My Value, all the time", name = value
- "My Name" = "This" "name"", name = value
In the third example, the quoted values contain equals and commas. In the fourth example, the quoted value contains a quote. Instances have the following properties initialized:

- map: the map (JavaScript object being used as a hashmap) form of the name/value pairs, with Microsoft-style quotes removed
- string: the string form of the name/value pairs, with Microsoft-style quoting.

Use with any server-side script where you need convert name/value pairs between maps and strings.

**NameValuePairs - mapToString()**

Produces `this.string` from `this.map`.

---

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

---

**NameValuePairs - NameValuePairs(String stringForm)**

Creates an instance of the class initialized with the specified string.

---

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stringForm</td>
<td>String</td>
<td>The values to use when creating the object.</td>
</tr>
</tbody>
</table>

---

**NameValuePairs - NameValuePairs(Object mapForm)**

Creates an instance of the class initialized with the specified hash map.

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
NameValuePairs - quotify(Object value)
If the specified value contains double quotes, equals, commas, or spaces, this method surrounds the value with double quotes and doubles up any enclosed double quotes.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Object</td>
<td>The value to process</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The value with added quotes.</td>
</tr>
</tbody>
</table>

NameValuePairs - stringToMap()
Produces this.map from this.string.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**NetwareHandler API - Global**
Implements an SNMP handler for Netware OS.
This handler is designed to be invoked by Classify.java as an interim step toward completely moving the SNMP sensors into JavaScript.

Use this API for SNMP classification.

**NetwareHandler - classifyAndIdentify()**

Classifies and identifies the SNMP device.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

**Notify - Global**

The **Notify** API allows you to interact with Notify calls and SMS messages using scripts.

Access the global **Notify** class and its associated methods in the **SNC** namespace.

**Notify - call(String notifyPhoneNumber, String toPhoneNumber, GlideRecord conferenceCall, GlideRecord conferenceCallRecord, String userSysId, String groupSysId, GlideRecord sourceRecord)**

Calls the specified E.164-compliant telephone number.

In addition, this method can automatically add the specified phone number to a specified conference call.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>notifyPhoneNumber</td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>toPhoneNumber</td>
</tr>
<tr>
<td>conferenceCall</td>
</tr>
<tr>
<td>userSysId</td>
</tr>
<tr>
<td>groupSysId</td>
</tr>
<tr>
<td>sourceRecord</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example illustrates how to initiate a call to another phone number.
This example illustrates how to initiate a call to list of telephone numbers and automatically join those numbers into a new conference call.

```
var notify = new SNC.Notify();
var from = '+14041234567';
var participants = ['+31612345678', '+31623456789', '+31687654321'];

// set up a conference call
var conferenceCall = notify.conferenceCall();

// set up the outbound calls for all conference call participants
for (var i in participants) {
    var to = participants[i];
    notify.call(from, to, conferenceCall);
}
```

This example illustrates how to initiate a new conference call.

```
SNC.Notify.call('+15413970605', '+91406XXXXXXX', SNC.Notify.conferenceCall(), null, null, null);
```

This example illustrates how to initiate a new conference call using a user record.

```
var sysUserGr = new GlideRecord('sys_user');
sysUserGr.get('active conference sys id');

if (conferenceGr.isValid) {
    SNC.Notify.call('+15413970605', '+91406XXXXXXX', SNC.Notify.conferenceCall(),
    sysUserGr.getUniqueValue(), null, null);
}
```

This example illustrates how to initiate a new conference call with a user, group, and source record.

```
var sysUserGr = new GlideRecord('sys_user');
sysUserGr.get('active sys user sys id');

var sysUserGroupGr = new GlideRecord('sys_user_group');
sysUserGroupGr.get('active sys user group sys id');
```
var incidentGr = new GlideRecord('incident');
incidentGr.get('incident sys_id');

if (conferenceGr.isValid) {
    SNC.Notify.call('+15413970605', '+91406XXXXXXX',
    SNC.Notify.conferenceCall(),
    sysUserGr.getUniqueValue(),
    sysUserGroupGr.getUniqueValue(),
    incidentGr.getUniqueValue());
}

**Scoped equivalent**

To use the `call()` method in a scoped application, use the corresponding scoped method: `NotifyScoped - call(String notifyPhoneNumber, String toPhoneNumber, GlideRecord conferenceCall, GlideRecord conferenceCallRecord, String userSysId, String groupSysId, GlideRecord sourceRecord)`.

**Notify - conferenceCall(GlideRecord sourceRecord)**

Creates a new conference call GlideRecord.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceRecord</td>
<td>GlideRecord</td>
</tr>
<tr>
<td>Optional. Record that initiated the request to create the conference call. Used to populate the source and table fields on notify_conference_call record.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
</table>

var notify = new SNC.Notify();
var from = '+14041234567';
var participants = ['-31612345678', '+31623456789', '+31687654321'];

// set up a conference call
var conferenceCall = notify.conferenceCall();

// set up the outbound calls for all conference call participants
for (var i in participants) {
  var to = participants[i];
  notify.call(from, to, conferenceCall);
}

### Scoped equivalent

To use the `conferenceCall()` method in a scoped application, use the corresponding scoped method: `NotifyScoped - conferenceCall(GlideRecord sourceRecord)`.

### Related reference

- **Notify - call(String notifyPhoneNumber, String toPhoneNumber, GlideRecord conferenceCall, GlideRecord conferenceCallRecord, String userSysId, String groupSysId, GlideRecord sourceRecord)**
  - Resumes a call after it was put in a queue (on hold).
  - Use this method to resume calls that were put in a queue with the `queueCall()` method.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>callRecord</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

The following example shows how to reactivate a call that was put on hold.

```javascript
var notifyCallGr = new GlideRecord('notify_call');
notifyCallGr.get('active participant sys id');
if (notifyCallGr.isValid) {
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Scoped equivalent

To use the `dequeueCall()` method in a scoped application, use the corresponding scoped method: `NotifyScoped - dequeueCall(GlideRecord callRecord)`.

`Notify - forwardCall(GlideRecord call, String destination, String dtmf)`
Forwards the specified call to a different call recipient.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>call</td>
<td>GlideRecord or String</td>
<td>Notify call record or the telephony provider call ID, of the call to be forwarded.</td>
</tr>
<tr>
<td>destination</td>
<td>GlideRecord or String</td>
<td>Notify phone number record or an E.164-compliant phone number, of the caller to which to forward the call.</td>
</tr>
<tr>
<td>dtmf</td>
<td>String</td>
<td>Dual Tone - Multi Frequency (DTMF) code to send upon call connection.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to forward a call to another phone number.

```javascript
var callID = 'CA92374b5aa561dab476a7001db6026edc'; // Twilio Call ID
var phoneNumber = '+91406xxxxxxx';
var dtmfTones = null;

var notifyCallGr = new GlideRecord('notify_call');
notifyCallGr.get('active participant sys id');

if (notifyCallGr.isValid) {
    SNC.Notify.forwardCall(notifyCallGr(or) callID, phoneNumber, dtmfTones)
}
```
Scoped equivalent

To use the `forwardCall()` method in a scoped application, use the corresponding scoped method: NotifyScoped - `forwardCall(GlideRecord call, String destination, String dtmf)`.

Notify - `getAvailableClients(String notifyNumber)`

Returns a list of client sessions that are available to receive calls.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyNumber</td>
<td>String</td>
<td>Valid Notify phone number.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>GlideRecord from the <code>notify_client_session</code> table for the specified phone number. Returns &quot;0&quot; if there are no available client sessions.</td>
</tr>
</tbody>
</table>

The following example shows how to use the `getAvailableClients()` method to index into the `notify_client_session` table and then iterate across all available Notify clients.

```javascript
var clientSessionGr = SNC.Notify.getAvailableClients('+185xxxxxxxx');
// Here clientSessionGr is of type GlideRecord on 'notify_client_session' table.

var isLoggedinUserAvailable = false;
while (clientSessionGr.next()) {
    if (clientSessionGr.user == gs.getUserID())
        isLoggedinUserAvailable = clientSessionGr.available;
}
gs.info('isLoggedinUserAvailable - ' + isLoggedinUserAvailable);
```

Scoped equivalent

To use the `getAvailableClients()` method in a scoped application, use the corresponding scoped method: NotifyScoped - `getAvailableClients(String notifyNumber)`. 
**Notify - getParentCallID(GlideRecord callRecord)**

Returns the sys_id of a specified call’s parent call.

Any call started by forwarding another call, such as with the Forward workflow activity, is considered a child of the original call. The original call is the parent call.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callRecord</td>
<td>GlideRecord</td>
<td>Record on the Notify Call [notify_call] table for which to return the call status.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Unique sys_id of the parent call record.</td>
</tr>
</tbody>
</table>

This example shows how to obtain the parent call of the specified call.

```javascript
var callRecord = new GlideRecord('notify_call');
callRecord.get("0f4f5863ff13310014ecffffff28");

var notify = new SNC.Notify();
var parentCallID = notify.getParentCallID(callRecord);
```

**Notify - getPhoneNumbers()**

Returns all phone numbers and short codes available to Notify.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>List</td>
<td>List of NotifyPhoneNumber objects, each object representing one phone number available to Notify.</td>
</tr>
</tbody>
</table>
```
var list = SNC.Notify.getPhoneNumbers();
for (var i = 0; i < list.size(); i++) {
  var num = list.get(i);
  gs.info(num.getNumber())
}
```

**Scoped equivalent**

To use the `getPhoneNumbers()` method in a scoped application, use the corresponding scoped method: `NotifyScoped - getPhoneNumbers()`.

**Notify - getTokens(GlideRecord, record)**

Returns client tokens for any active telephony drivers for use in WebRTC or mobile clients.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>GlideRecord</td>
<td>GlideRecord to use to identify the Notify client, such as a group record or a user record.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Web RTC tokens for the supported drivers, as a JSON string with the following format: {driverName1: “token1”, driverName2: “token2”}, such as “TwilioDirect”:“eyJhxxxx.eyJleHAiOiIxxxx.7fejxxx_mbLxxx”</td>
</tr>
</tbody>
</table>

This example shows how to obtain the client tokens for the currently logged in user.

```javascript
// get Notify client Tokens per active Notify driver for the currently logged in user
var json = new SNC.Notify().getTokens();

// Parse the JSON that was return into a tokens object
var tokens = JSON.parse(json);

// Log line
gs.log('Notify client tokens for the currently logged in user');

// iterate over the driver tokens
for (var driver in tokens) {
  // do something with tokens
} // end for
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
gs.log(driver + ' Driver token: ' + tokens[driver]);
}

This example shows how to obtain the client tokens for every Notify group.

```javascript
// instantiate Notify
var notify = new SNC.Notify();

// get all Notify Groups
var notifyGroup = new GlideRecord("notify_group");
notifyGroup.query();

// iterate over all notify groups
while (notifyGroup.next()) {
    // generate Notify Client tokens per active Notify Driver for this group
    var json = notify.getTokens(notifyGroup);
    var tokens = JSON.parse(json);

    for (var driver in tokens) {
        gs.log(gs.getMessage("Notify Client token for {0} driver and Notify Group '{1}': {2}",
            [driver, notifyGroup.getValue('name'), tokens[driver]]));
    }
}
```

**Scoped equivalent**

To use the `getTokens()` method in a scoped application, use the corresponding scoped method: `NotifyScoped - getTokens(GlideRecord, record)`.

**Notify - getTokenTTL(String owner)**

Returns the maximum amount of time that a client session stays active for a specified telephony driver before automatically timing out.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>owner</td>
<td>String</td>
<td>Name of the telephony driver for which to retrieve the session length.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Twilio: for the old driver</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TwilioDirect: for the new driver</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The following example shows how to properly call this method and the associated response. It also shows what is returned if an invalid driver is passed.

```javascript
var owner = "TwilioDirect";  // Valid driver
var ttl = SNC.Notify.getTokenTTL(owner);
gs.info("Token TTL for " + owner + " --> " + ttl);

owner = "Abcxyz";  // Invalid driver
ttl = SNC.Notify.getTokenTTL(owner);
// For an invalid driver, we throw NoSuchNotifyDriverException saying that Abcxyzdriver is not available
// and return the default value of TTL
gs.info("Token TTL for " + owner + " --> " + ttl);
```

### Scoped equivalent

To use the `getTokenTTL()` method in a scoped application, use the corresponding scoped method: `NotifyScoped - getTokenTTL(String owner)`.

#### Notify - hasCapability(String notifyPhoneNumber, String capability)

Determines whether the specified phone number has the specified capability.

The telephony driver associated with the phone number contains a list of all of the capabilities of the phone.

**Note:** In the base system, the Notify JS driver only has 'show_speakers' as a capability; this can be modified.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyPhoneNumber</td>
<td>String</td>
<td>Phone number for which to check for the specified capability.</td>
</tr>
<tr>
<td>capability</td>
<td>String</td>
<td>Capability to detect. The string text must be an exact match to what is in the phone.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the specified phone has the specified capability.</td>
</tr>
<tr>
<td></td>
<td>• true: phone has the capability</td>
</tr>
<tr>
<td></td>
<td>• false: phone does not have the capability</td>
</tr>
</tbody>
</table>

This example shows how to check if a phone has a specific capability.

```javascript
// Each driver has a defined set of capabilities.

var capability = 'show_speakers';
gs.info(SNC.Notify.hasCapability('+185xxxxxxxx', capability)); // true

capability = 'send_sms';
gs.info(SNC.Notify.hasCapability('+185xxxxxxxx', capability)); // false
```

### Scoped equivalent

To use the `hasCapability()` method in a scoped application, use the corresponding scoped method: `NotifyScoped - hasCapability(String notifyPhoneNumber, String capability)`.

### Notify - `kick(GlideRecord participant)`

Removes the specified caller from the current Notify conference call.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>participant</td>
<td>GlideRecord</td>
</tr>
<tr>
<td></td>
<td>GlideRecord object containing the Notify Participant [notify_participant] record of the caller to remove from the conference call.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
The following example shows how to remove a participant from a conference call.

```javascript
var notifyParticipantGr = new GlideRecord('notify_participant');
notifyParticipantGr.get('active participant sys id');

if (notifyParticipantGr.isValid) {
    SNC.Notify.kick(notifyParticipantGr);
}
```

**Scoped equivalent**

To use the `kick()` method in a scoped application, use the corresponding scoped method: `NotifyScoped - kick(GlideRecord participant)`.

**Notify - modifyCall(GlideRecord callRecord, NotifyAction notifyAction)**

Performs one or more actions on an active Notify phone call.

Available actions that you can perform on calls include queuing or dequeueing the call, reading text, playing audio, or forwarding the call. You can also create custom actions. Call the `modifyCall()` method after you have specified all the actions that you want to apply to the specified call. Refer to the `NotifyAction` API documentation for more information about available actions.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>callRecord</code></td>
<td>GlideRecord</td>
<td>Notify Call [notify_call] record of the call for which to apply the actions.</td>
</tr>
<tr>
<td><code>notifyAction</code></td>
<td>NotifyAction</td>
<td>NotifyAction object describing one or more actions to perform on the call. Create this object by calling one or more of the <code>NotifyAction()</code> methods.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to modify a call.

```javascript
// get the most recent call record
var callRecord = new GlideRecord('notify_call');
```
callRecord.orderByDesc('sys_created_on');
callRecord.query();

if (callRecord.next()) {
    // instantiate notify action container class
    var notifyAction = new SNC.NotifyAction();
    // The call is already in progress. Now, we want to modify the call behavior by putting
    this call in a queue.
    // So, we add a queue action to queue the call
    notifyAction.addQueue('my fancy queue');

    // modify the call by passing in the above action, putting the call in a queue
    new SNC.Notify().modifyCall(callRecord, notifyAction);
} else {
    gs.log('no such call record');
}

**Scoped equivalent**

To use the modifyCall() method in a scoped application, use the corresponding
scoped method: NotifyScoped - modifyCall(GlideRecord callRecord, NotifyAction notifyAction).

**Notify - mute(GlideRecord participantRecord)**

Mutes the specified conference call participant.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>participantRecord</td>
<td>GlideRecord</td>
<td>GlideRecord from the notify_participant table for the participant to mute.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to mute a caller.

```javascript
var notifyParticipantGr = new GlideRecord('notify_participant');
notifyParticipantGr.get('active participant sys id');
```
if (notifyParticipantGr.isValid) {
    SNC.Notify.mute(notifyParticipantGr);
}

Scoped equivalent
To use the `mute()` method in a scoped application, use the corresponding scoped method: `NotifyScoped - mute(GlideRecord participantRecord)`.

**Notify - queueCall(GlideRecord callRecord)**

Puts the specified call into a queue (on hold).

Resume a queued call using the `dequeueCall()` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callRecord</td>
<td>GlideRecord</td>
<td>GlideRecord object of the Notify Call record (notify_call table) to put on hold.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to put a call on hold (in the queue).

```javascript
var notifyCallGr = new GlideRecord('notify_call');
notifyCallGr.get('active participant sys id');
if (notifyCallGr.isValid) {
    SNC.Notify.queueCall(notifyCallGr);
}
```

Scoped equivalent
To use the `queueCall()` method in a scoped application, use the corresponding scoped method: `NotifyScoped - queueCall(GlideRecord callRecord)`.
**Notify - sendBulkSMS(NotifyPhoneNumber notifyPhoneNumber, String toPhoneNumbers, String messageBody, GlideRecord source)**

Sends a specified SMS message to the specified list of Notify clients (phone numbers).

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyPhoneNumber</td>
<td>NotifyPhoneNumber</td>
<td>Notify phone number from which the SMS message is being sent.</td>
</tr>
<tr>
<td>toPhoneNumbers</td>
<td>String</td>
<td>Comma separated list of phone numbers to which to send the SMS message. Format: E.164</td>
</tr>
<tr>
<td>messageBody</td>
<td>String</td>
<td>SMS text to send.</td>
</tr>
<tr>
<td>source</td>
<td>GlideRecord</td>
<td>Source record that prompted this SMS message, such as an incident.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Null</td>
</tr>
</tbody>
</table>

The following example shows how to send a bulk SMS message.

```java
var incidentGr = new GlideRecord('incident');
incidentGr.get(active incident sys_id);
if (incidentGr.isValid()) {
   SNC.Notify.sendBulkSMS('+15413970605', ['+919885XXXXXX', '+919775XXXXXX'], 'Test automation message', incidentGr);
}
```

**Scoped equivalent**

To use the `sendBulkSMS()` method in a scoped application, use the corresponding scoped method: **NotifyScoped - sendBulkSMS(NotifyPhoneNumber notifyPhoneNumber, String toPhoneNumbers, String messageBody, GlideRecord source)**.
Notify - sendSMS(NotifyPhoneNumber notifyPhoneNumber, String toPhoneNumber, String messageBody, GlideRecord source)

Sends an SMS text message to an E.164-compliant phone number.

This method creates a new record on the Notify Message [notify_message] table and associates it with the source record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyPhoneNumber</td>
<td>String</td>
<td>Notify phone number or short code to which to send this SMS message.</td>
</tr>
<tr>
<td>toPhoneNumber</td>
<td>String</td>
<td>E.164-compliant phone number to which to send the SMS message.</td>
</tr>
<tr>
<td>messageBody</td>
<td>String</td>
<td>Body of the SMS text message.</td>
</tr>
<tr>
<td>source</td>
<td>GlideRecord</td>
<td>Source record that prompted this SMS message, such as an incident.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Unique message SID; stored in the Notify Message [notify_message] record as message_id.</td>
</tr>
</tbody>
</table>

The following example shows how to send an SMS message.

```javascript
var incidentGr = new GlideRecord('incident');
incidentGr.get('active incident sys_id');
if (incidentGr.isValid()) {
    SNC.Notify.sendSMS('+15413970605', '+919885XXXXXX', 'Test automation message', incidentGr);
}
```

**Scoped equivalent**

To use the `sendSMS()` method in a scoped application, use the corresponding scoped method: NotifyScoped - sendSMS(NotifyPhoneNumber notifyPhoneNumber, String toPhoneNumber, String messageBody, GlideRecord source).
Notify - unmute(GlideRecord participantRecord)

Unmutes the specified conference call participant.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>participantRecord</td>
<td>GlideRecord</td>
<td>GlideRecord from the notify_participant table for the participant to unmute.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to unmute a caller.

```javascript
var notifyParticipantGr = new GlideRecord('notify_participant');
notifyParticipantGr.get('active participant sys id');

if (notifyParticipantGr.isValid) {
    SNC.Notify.unmute(notifyParticipantGr);
}
```

**Scoped equivalent**

To use the `unmute()` method in a scoped application, use the corresponding scoped method: `NotifyScoped - unmute(GlideRecord participantRecord)`.

**Notify - Scoped**

The `NotifyScoped` API allows you to interact with Notify calls and SMS messages using scripts.

Access the scoped `Notify` class and its associated methods from the `sn_notify` namespace.

`NotifyScoped - call(String notifyPhoneNumber, String toPhoneNumber, GlideRecord conferenceCall, GlideRecord conferenceCallRecord, String userSysId, String groupSysId, GlideRecord sourceRecord)`

Calls the specified E.164-compliant telephone number.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyPhoneNumber</td>
<td>String</td>
<td>Notify phone number from which to make the call. When you initiate a call, the outgoing call workflow for the number group associated with this number runs. Ensure this workflow includes a <strong>join conference call</strong> activity to connect the user to the conference call.</td>
</tr>
<tr>
<td>toPhoneNumber</td>
<td>String</td>
<td>Phone number to call. Called numbers are added to the conference call.</td>
</tr>
<tr>
<td>conferenceCall</td>
<td>GlideRecord</td>
<td>Optional. If this parameter is passed in, the callers identified in the <strong>toPhoneNumber</strong> parameter are automatically joined into the conference call identified by this record.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GlideRecord for the Notify Call [notify_call] table which identifies the conference call record. This record is automatically added to the outgoing call workflow scratchpad as the workflow.scratchpad.conference_call variable.</td>
</tr>
<tr>
<td>userSysId</td>
<td>String</td>
<td>Optional. Unique identifier (sys_id) of the user associated with the call.</td>
</tr>
<tr>
<td>groupSysId</td>
<td>String</td>
<td>Optional. Unique identifier (sys_id) of the group associated with the call.</td>
</tr>
<tr>
<td>sourceRecord</td>
<td>GlideRecord</td>
<td>Optional. Source record that prompted this call.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example illustrates how to initiate a call to another phone number.
var from = '+14048007337';
var to = '+31646810495';

// set up call
new sn_notify.NotifyScoped().call(from, to);

This example illustrates how to initiate a call to a list of telephone numbers and automatically join those numbers into a new conference call.

var notify = new sn_notify.NotifyScoped();
var from = '+14041234567';
var participants = ['+31612345678', '+31623456789', '+31687654321'];

// set up a conference call
var conferenceCall = notify.conferenceCall();

// set up the outbound calls for all conference call participants
for (var i in participants) {
    var to = participants[i];
    notify.call(from, to, conferenceCall);
}

NotifyScoped - conferenceCall(GlideRecord sourceRecord)
Creates a new conference call GlideRecord.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceRecord</td>
<td>GlideRecord</td>
<td>Optional. Record that initiated the request to create the conference call. Used to populate the source and table fields on notify_conference_call record.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
var participants = ['+31612345678', '+31623456789', '+31687654321'];

// set up a conference call
var conferenceCall = notify.conferenceCall();

// set up the outbound calls for all conference call participants
for (var i in participants) {
    var to = participants[i];
    notify.call(from, to, conferenceCall);
}

**NotifyScoped - dequeueCall(GlideRecord callRecord)**
Resumes a call after it was put in a queue (on hold).

Use this method to resume calls that were put in a queue with the `queueCall()` method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callRecord</td>
<td>GlideRecord</td>
<td>GlideRecord object on the Notify Call [notify_call] table with the held call you want to resume.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to reactivate a call that was put on hold.

```javascript
var notifyCallGr = new GlideRecord('notify_call');
notifyCallGr.get('active participant sys id');

if (notifyCallGr.isValid) {
    sn_notify.NotifyScoped.dequeueCall(notifyCallGr);
}
```

**NotifyScoped - forwardCall(GlideRecord call, String destination, String dtmf)**
Forwards the specified call to a different call recipient.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>call</td>
<td>GlideRecord or String</td>
<td>Notify call record or the telephony provider call ID, of the call to be forwarded.</td>
</tr>
<tr>
<td>destination</td>
<td>GlideRecord or String</td>
<td>Notify phone number record or an E.164-compliant phone number, of the caller to which to forward the call.</td>
</tr>
<tr>
<td>dtmf</td>
<td>String</td>
<td>Dual Tone - Multi Frequency (DTMF) code to send upon call connection.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to forward a call to another phone number.

```javascript
var callID = 'CA92374b5aa561dab476a7001db6026edc'; // Twilio Call ID
var phoneNumber = '+91406xxxxxxx';
var dtmfTones = null;

var notifyCallGr = new GlideRecord('notify_call');
notifyCallGr.get('active participant sys id');

if (notifyCallGr.isValid) {
    sn_notify.NotifyScoped.forwardCall(notifyCallGr(or) callID, phoneNumber, dtmfTones)
}
```

**NotifyScoped - getAvailableClients(String notifyNumber)**

Returns a list of client sessions that are available to receive calls.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyNumber</td>
<td>String</td>
<td>Valid Notify phone number.</td>
</tr>
</tbody>
</table>
The following example shows how to use the getAvailableClients() method to index into the notify_client_session table and then iterate across all available Notify clients.

```javascript
var clientSessionGr = sn_notify.NotifyScoped.getAvailableClients('+185xxxxxxxx');
// Here clientSessionGr is of type GlideRecord on 'notify_client_session' table.

var isLoggedInUserAvailable = false;
while (clientSessionGr.next()) {
    if (clientSessionGr.user == gs.getUserID())
        isLoggedInUserAvailable = clientSessionGr.available;
}
gs.info('isLoggedInUserAvailable - ' + isLoggedInUserAvailable);
```

**NotifyScoped - getPhoneNumbers()**

Returns all phone numbers and short codes available to Notify.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>List</td>
<td>List of NotifyPhoneNumber objects, each object representing one phone number available to Notify.</td>
</tr>
</tbody>
</table>

This example illustrates how to obtain the Notify phone numbers and then iterate over the list.

```javascript
// instantiate notify
var notify = new sn_notify.NotifyScoped();
```
// get all available phone numbers
var phoneNumber = notify.getPhoneNumber();

// iterate over phone numbers
for (var i = 0; i < phoneNumber.size(); i++) {
    var number = phoneNumber.get(i);

    // perform any actions using each phone number
}

**NotifyScoped - getShortCodes()**

Returns all short codes available to Notify.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example illustrates how to obtain the Notify short codes and then iterate over the list.

// instantiate notify
var notify = new sn_notify.NotifyScoped();

// get all available shortcodes
var shortCodes = notify.getShortCodes();

// iterate over phone numbers
for (var i = 0; i < shortCodes.size(); i++) {
    var shortCode = shortCodes.get(i);
    gs.log(shortCode.getNumber());
}
//perform any actions using each shortcode
}

**NotifyScoped - getTokens(GlideRecord, record)**

Returns client tokens for any active telephony drivers for use in WebRTC or mobile clients.

This method uses the currently logged-in user record as the client.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>GlideRecord</td>
<td>GlideRecord to use to identify the Notify client, such as a group record or a user record.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Web RTC tokens for the supported drivers, as a JSON string with the following format: <code>{driverName1: “token1”, driverName2: “token2”}</code>, such as “TwilioDirect”:“eyJhxxxx.eyJleHAiOiIxxxx.7fejxxx_mbLxxx”</td>
</tr>
</tbody>
</table>

This example shows how to obtain the client tokens for the currently logged in user.

```javascript
// get Notify client Tokens per active Notify driver for the currently logged in user
var json = new sn_notify.NotifyScoped().getTokens();

// Parse the JSON that was return into a tokens object
var tokens = JSON.parse(json);

// Log line
gs.log('Notify client tokens for the currently logged in user');

// iterate over the driver tokens
for (var driver in tokens) {
    gs.log(driver + ' Driver token: ' + tokens[driver]);
}
```

This example shows how to obtain the client tokens for every Notify group.
// instantiate Notify
var notify = new sn_notify.NotifyScoped.Notify();

// get all Notify Groups
var notifyGroup = new GlideRecord("notify_group");
notifyGroup.query();

// iterate over all notify groups
while (notifyGroup.next()) {
    // generate Notify Client tokens per active Notify Driver for this group
    var json = notify.getTokens(notifyGroup);
    var tokens = JSON.parse(json);
    for (var driver in tokens) {
        gs.log(gs.getMessage("Notify Client token for {0} driver and Notify Group '{1}': {2}",
            [driver, notifyGroup.getValue('name'), tokens[driver]]));
    }
}

This example illustrates how to obtain the client tokens for the active telephony drivers.

var notify = new sn_notify.NotifyScoped();
var now_GR = new GlideRecord('sys_user');
if (now_GR.get(gs.getUserID())) {
    gs.info(notify.getTokens(now_GR));
}

Output:

{"TwilioDirect":"eyJhxxxx.eyJleHAiOiIxxxx.7fejxxx_mbLxxx"}

**NotifyScoped - getTokenTTL(String owner)**

Returns the maximum amount of time that a client session stays active for a specified telephony driver before automatically timing out.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>owner</td>
<td>String</td>
<td>Name of the telephony driver for which to retrieve the session length. Valid values:</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Twilio</td>
<td></td>
<td>for the old driver</td>
</tr>
<tr>
<td>TwilioDirect</td>
<td></td>
<td>for the new driver</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>Maximum length of the session (in seconds).</td>
</tr>
<tr>
<td>Default:</td>
<td>1800 seconds</td>
</tr>
</tbody>
</table>

The following example shows how to properly call this method and the associated response. It also shows what is returned if an invalid driver is passed.

```javascript
var owner = "TwilioDirect";  // Valid driver
var ttl = SNC.Notify.getTokenTTL(owner);
gs.info("Token TTL for " + owner + " --> " + ttl);

owner = "Abcxyz";  // Invalid driver
ttl = SNC.Notify.getTokenTTL(owner);
// For an invalid driver, we throw NoSuchNotifyDriverException saying that Abcxyzdriver is not available
// and return the default value of TTL
gs.info("Token TTL for " + owner + " --> " + ttl);
```

**NotifyScoped - hasCapability(String notifyPhoneNumber, String capability)**

Determines whether the specified phone number has the specified capability.

The telephony driver associated with the phone number contains a list of all of the capabilities of the phone.

⚠️ **Note:** In the base system, the Notify JS driver only has 'show_speakers' as a capability; this can be modified.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyPhoneNumber</td>
<td>String</td>
<td>Phone number for which to check for the specified capability.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>capability</td>
<td>String</td>
<td>Capability to detect. The string text must be an exact match to what is in the phone.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean| Flag that indicates whether the specified phone has the specified capability.  
  • true: phone has the capability  
  • false: phone does not have the capability |

This example shows how to check if a phone has a specific capability.

```javascript
// Each driver has a defined set of capabilities.

var capability = 'show_speakers';
gs.info(sn_notify.NotifyScoped.hasCapability('+185xxxxxxxx', capability)); // true

capability = 'send_sms';
gs.info(sn_notify.NotifyScoped.hasCapability('+185xxxxxxxx', capability)); // false
```

**NotifyScoped - kick(GlideRecord participant)**

Removes the specified caller from the current Notify conference call.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>participant</td>
<td>GlideRecord</td>
<td>GlideRecord object containing the Notify Participant [notify_participant] record of the caller to remove from the conference call.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
This example shows how to remove a participant from a call.

```javascript
var participant = new GlideRecord('notify_participant');
participant.get('<sys_id>');
if (participant.isValid()) {
    new sn_notify.NotifyScoped().kick(participant);
}
```

**NotifyScoped - modifyCall(GlideRecord callRecord, NotifyAction notifyAction)**

Performs one or more activities on an active Notify phone call.

Use this method to change the behavior of a call. For example, transferring a call, playing audio, or forcing a hangup.

⚠️ **Note:** The scoped implementation of this method only supports custom Notify activities. Unlike the global implementation, it does not provide a NotifyAction API. For details on how to create a custom Notify activity, see Notify workflow activities.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callRecord</td>
<td>GlideRecord</td>
<td>Notify Call [notify_call] record of the call for which to apply the actions.</td>
</tr>
<tr>
<td>notifyAction</td>
<td>NotifyAction</td>
<td>NotifyAction object describing one or more activities to perform on the call.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**NotifyScoped - mute(GlideRecord participantRecord)**

Mutes the specified conference call participant.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>participantRecord</td>
<td>GlideRecord</td>
<td>GlideRecord from the notify_participant table for the participant to mute.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to mute a caller.

```javascript
var notifyParticipantGr = new GlideRecord('notify_participant');
notifyParticipantGr.get('active participant sys id');

if (notifyParticipantGr.isValid) {
    sn_notify.NotifyScoped.mute(notifyParticipantGr);
}
```

**NotifyScoped - queueCall(GlideRecord callRecord)**

Puts the specified call into a queue (on hold).

Resume a queued call using the `dequeueCall()` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callRecord</td>
<td>GlideRecord</td>
<td>GlideRecord object of the Notify Call record (notify_call table) to put on hold.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var call = new GlideRecord('notify_call');
call.get('<call record sys_id>');
```
if (call.isValid()) {
    new sn_notify.NotifyScoped().queueCall(call);
}

NotifyScoped - sendBulkSMS(NotifyPhoneNumber notifyPhoneNumber, String toPhoneNumbers, String messageBody, GlideRecord source)

Sends a specified SMS message to the specified list of Notify clients (phone numbers).

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyPhoneNumber</td>
<td>NotifyPhoneNumber</td>
<td>Phone number from which the SMS message is being sent.</td>
</tr>
<tr>
<td>toPhoneNumbers</td>
<td>String</td>
<td>Comma separated list phone numbers to which to send the SMS message. Format: E.164-compliant</td>
</tr>
<tr>
<td>messageBody</td>
<td>String</td>
<td>SMS text to send.</td>
</tr>
<tr>
<td>source</td>
<td>GlideRecord</td>
<td>Source record that prompted this SMS message, such as an incident.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Null</td>
</tr>
</tbody>
</table>

This example shows how to send an SMS message to multiple phone numbers (bulk SMS).

```java
var incidentGr = new GlideRecord('incident');
incidentGr.get('active incident sys_id');
if (incidentGr.isValid()) {
    sn_notify.NotifyScoped.sendBulkSMS('+15413970605', ['+919885XXXXXX', '+919775XXXXXX'],
    'Test automation message', incidentGr);
}
```
NotifyScoped - sendSMS(NotifyPhoneNumber notifyPhoneNumber, String toPhoneNumber, String messageBody, GlideRecord source)

Sends an SMS text message to an E.164-compliant phone number.

This method creates a new record on the Notify Message [notify_message] table and associates it with the source record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyPhoneNumber</td>
<td>NotifyPhoneNumber</td>
<td>Notify phone number or short code to which to send this SMS message.</td>
</tr>
<tr>
<td>toPhoneNumber</td>
<td>String</td>
<td>E.164-compliant phone number to which to send the SMS message.</td>
</tr>
<tr>
<td>messageBody</td>
<td>String</td>
<td>SMS text message.</td>
</tr>
<tr>
<td>source</td>
<td>GlideRecord</td>
<td>Source record that prompted this SMS message, such as an incident.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Unique message SID; stored in the Notify Message [notify_message] record as message_id.</td>
</tr>
</tbody>
</table>

The following example shows how to send an SMS message.

```java
var incidentGr = new GlideRecord('incident');
incidentGr.get('active incident sys_id');
if (incidentGr.isValid()) {
    sn_notify.NotifyScoped.sendSMS('+15413970605', '+919885XXXXXX', 'Test automation message', incidentGr);
}
```

NotifyScoped - unmute(GlideRecord participantRecord)

Unmutes the specified conference call participant.
This example shows how to unmute a specified call participant.

```javascript
var notifyParticipantGr = new GlideRecord('notify_participant');
notifyParticipantGr.get('active participant sys id');

if (notifyParticipantGr.isValid) {
    sn_notify.NotifyScoped.unmute(notifyParticipantGr);
}
```

**NotifyAction - Global**

The `NotifyAction` API allows you to define actions to send to a telephony provider.

You add actions to a `NotifyAction` object by calling the respective add function for each type of action. Each add function returns an Action object, such as a `SayAction` object for the `addSay()` function. Refer to each method example for information about returned objects.

⚠️ **Note:** The `NotifyAction` API is only valid within global applications. To utilize this type of functionality within scoped applications, you must create custom Notify workflow activities. For additional information on creating these activities, see Notify workflow activities.

**NotifyAction - addConference()**

Adds a conference action to move the current call into the current conference call.

---

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>participantRecord</td>
<td>GlideRecord</td>
<td>GlideRecord from the notify_participant table for the participant to unmute.</td>
</tr>
</tbody>
</table>
This example demonstrates how to add a conference action and set the name of the conference.

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// add a conference call action and set its name
var conference = notifyAction.addConference();
conference.setName('Brown Bag: Week 3');
```

**NotifyAction - addConference.setEndOnExit(Boolean endOnExit)**

Defines whether the conference call should end when a specified caller exits the conference call.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| endOnExit | Boolean | Flag that indicates whether the conference call should end when the specified caller exits the current conference call.  
  • true: end the conference call when the specified caller exits the conference call  
  • false: Default. End the conference call once all participants exit |
This example demonstrates how to add a conference call action and then set it so the conference call ends when the specified caller exits.

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// add a conference call action and set the caller that starts the meeting
var conference = notifyAction.addConference();

// retrieve the participant for which the conference call should exit when they leave
var notifyParticipantGr = new GlideRecord('notify_participant');
notifyParticipantGr.get('active participant sys id');
if (notifyParticipantGr.isValid) {
    conference.setEndOnExit(true);
}
```

**NotifyAction - addConference.setHangupOnStar(Boolean hangupOnStar)**

Defines whether the conference call should end when a participant presses the star (*) key.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| hangupOnStart | Boolean | Flag that indicates whether the conference call should end when a participant presses the star (*) key. Valid values:  
• true: end the conference call  
• false: Default. Don't end the conference call |
This example demonstrates how to add a conference action and then record the conference call.

```
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// add a conference call action and set the hang up action
var conference = notifyAction.addConference();
conference.setHangupOnStar(true);
```

**NotifyAction - addConference.setMuted(Boolean muted)**

Defines whether the specified caller should be muted in the current conference call.

If you do not call this method, the caller is not muted by default.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>muted</td>
<td>Boolean</td>
<td>Flag that indicates whether the specified caller should be muted in the current conference call.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: caller should be muted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Default. Caller should not be muted</td>
</tr>
</tbody>
</table>

This example demonstrates how to add a conference action and then mute a specified caller.

```
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();
```
// add a conference call action and set it to mute the specified participant
var conference = notifyAction.addConference();

var notifyParticipantGr = new GlideRecord('notify_participant');
notifyParticipantGr.get('active participant sys id');

if (notifyParticipantGr.isValid) {
    conference.setMuted(true);
}

**NotifyAction - addConference.setName(String name)**

Sets the name of the current conference call to the specified name.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name to associate with the current conference call.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example demonstrates how to add a conference action and set the name of the conference call.

// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// add a conference call action and set its name
var conference = notifyAction.addConference();
conference.setName('Brown Bag: Week 3');

**NotifyAction - addConference.setRecord(Boolean record)**

Defines whether the associated conference call should be recorded.

If you do not call this method, the conference call is not recorded by default.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>Boolean</td>
<td>Flag that indicates whether the current conference call should be recorded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: start recording the conference call</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Default. Conference call is not recorded. Recording is stopped if it is currently being recorded.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example demonstrates how to add a conference action and then record the conference call.

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// add a conference call action and set it to be recorded
var conference = notifyAction.addConference();
conference.setRecord(true);
```

**NotifyAction - addConference.setStartOnEnter(Boolean startOnEnter)**

Defines whether the conference call should start when the specified caller joins the conference call.

By default, whenever there are two or more callers, the conference call starts. To make it so the conference call only starts when a specific caller joins, you must call this method for each of the other callers and set the value to “false”. By doing this, the conference call will not start until the desired person joins the conference call.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startOnEnter</td>
<td>Boolean</td>
<td>Flag that indicates whether the conference call should start when the selected caller joins the current conference call.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• true: Default. Start the conference call when the specified caller joins the conference call.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• false: Start the conference call once it is added</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example demonstrates how to add a conference action and then set it so the conference call does not start until the specified caller joins.

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// add a conference call action and set the caller that starts the meeting
var conference = notifyAction.addConference();

// retrieve the participant for which the conference call should start when they arrive
var notifyParticipantGr = new GlideRecord('notify_participant');
notifyParticipantGr.get('active participant sys id');
if (notifyParticipantGr.isValid) {
  conference.setStartOnEnter(true);
}
```

NotifyAction - addDial()

Forwards a call to a specified phone number or Notify Client.

Once the addDial action is created, the associated phone number (setPhoneNumber()) or Notify Client (setClientRecord()) must also be set.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This example shows how to make an outbound call and record the call.

```java
// Initialize NotifyAction
var notifyAction = new SNC.NotifyAction();

// Call addDial() to connect to another party - this returns an object of type DialAction
var dialAction = notifyAction.addDial();

// Call setPhoneNumber(String phoneNumber) in DialAction.java to specify the phone number to dial
dialAction.setPhoneNumber('+919765xxxxxxx');

// Invoke setRecord(Boolean record) to record the call to this new number +919765xxxxxxx
dialAction.setRecord(true);
```

**NotifyAction - addDial.setCallerID(String callerID)**

Defines the caller ID for the outgoing call.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callerID</td>
<td>String</td>
<td>Caller identifier to set for the outgoing call.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to make an outbound call and set a call timeout.

```java
// Initialize NotifyAction
var notifyAction = new SNC.NotifyAction();

// Call addDial() to connect to another party - this returns an object of type DialAction
var dialAction = notifyAction.addDial();
```
// Call setPhoneNumber(String phoneNumber) in DialAction.java to specify the phone number to dial
dialAction.setPhoneNumber('+919765xxxxxxx');

// Set the caller ID
dialAction.setCallerID('Planning Conf Call');

### NotifyAction - addDial.setClientRecord(String tableName, String sysID)

Sets the current caller to a Notify caller by specifying the table in which to find the Notify caller record and the caller's unique identifier.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table that contains the desired caller's information.</td>
</tr>
<tr>
<td>sysID</td>
<td>String</td>
<td>Unique identifier (sys_id) of the desired Notify caller.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to set the current caller to a Notify caller using setClientRecord().

// set up a dial action to forward the call to the specified client
var action = new SNC.NotifyAction();
var dial = action.addDial();
dial.setClientRecord(notifyClientRecord.getTableName(),
    notifyClientRecord.getUniqueValue());
dial.setTimeout(activity.vars.timeout);
dial.setRecord(activity.vars.record);

### NotifyAction - addDial.setDTMF(String value)

Defines the DTMF tones to play when the call connects.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>Valid DTMF digits to play when the call connects.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to make an outbound call and defines the DTMF tones to play when the call connects.

```java
// Initialize NotifyAction
var notifyAction = new SNC.NotifyAction();

// Call addDial() to connect to another party - this returns an object of type DialAction
var dialAction = notifyAction.addDial();

// Call setPhoneNumber(String phoneNumber) in DialAction.java to specify the phone number to dial
dialAction.setPhoneNumber('+919765xxxxxxx');

// DTMF tones to play when call connects
dialAction.setDTMF("1246AF");
```

### NotifyAction - addDial.setHangupOnStar(Boolean hangupOnStar)

Defines whether the call should end when the star (*) key is pressed.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hangupOnStar</td>
<td>Boolean</td>
<td>Flag that indicates whether the call should end when the star (*) key is pressed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: end the call</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: don’t end the call</td>
</tr>
</tbody>
</table>
This example shows how to make an outbound call and set the hang up key to star.

```java
// Initialize NotifyAction
var notifyAction = new SNC.NotifyAction();

// Call addDial() to connect to another party - this returns an object of type DialAction
var dialAction = notifyAction.addDial();

// Call setPhoneNumber(String phoneNumber) in DialAction.java to specify the phone number to dial
dialAction.setPhoneNumber('+919765xxxxxxx');

// End call by pressing star
dialAction.setHangupOnStar(true);
```

**NotifyAction - addDial.setPhoneNumber(String phoneNumber)**

Defines the phone number to call.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phoneNumber</td>
<td>String</td>
<td>E.164-compliant phone number to call.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to make an outbound call and set a call timeout.

```java
// Initialize NotifyAction
var notifyAction = new SNC.NotifyAction();

// Call addDial() to connect to another party - this returns an object of type DialAction
var dialAction = notifyAction.addDial();
```
// Call setPhoneNumber(String phoneNumber) in DialAction.java to specify the phone number to
dial
dialAction.setPhoneNumber('+919765xxxxxxx');

**NotifyAction - addDial.setRecord(Boolean record)**

Defines whether the outgoing call should be recorded.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>Flag</td>
<td>Flag that indicates whether the outgoing call should be recorded. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: record the outgoing call</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: do not record the outgoing call</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to make an outbound call and record the call.

// Initialize NotifyAction
var notifyAction = new SNC.NotifyAction();

// Call addDial() to connect to another party - this returns an object of type DialAction
var dialAction = notifyAction.addDial();

// Call setPhoneNumber(String phoneNumber) in DialAction.java to specify the phone number to
dial
dialAction.setPhoneNumber('+919765xxxxxxx');

// Record the call
dialAction.setRecord(true);

**NotifyAction - addDial.setTimeout(Integer timeout)**

Sets the number of seconds after which the outgoing call times out.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeout</td>
<td>Integer</td>
<td>Number of seconds after which the outgoing call times out. Default: 30</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to make an outbound call and set a call timeout.

```java
// Initialize NotifyAction
var notifyAction = new SNC.NotifyAction();

// Call addDial() to connect to another party - this returns an object of type DialAction
var dialAction = notifyAction.addDial();

// Call setPhoneNumber(String phoneNumber) in DialAction.java to specify the phone number to dial
dialAction.setPhoneNumber('+919765xxxxxxx');

// Set the number of seconds to wait before timing out
dialAction.setTimeout(45);
```

### NotifyAction - addGather()

Presents a specified interactive phone menu to the caller.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GatherAction</td>
<td>Action added to the NotifyAction object. Use the GatherAction object to define the menu settings and options to present to the user.</td>
</tr>
</tbody>
</table>

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// present the user with a menu
var gather = notifyAction.addGather();
gather.setNumberOfDigits(1);    // the user can type 1 digit
gather.setFinishKey('*');       // # or *, useful for > 1 digit
gather.setTimeout(10);          // time to enter answer, in seconds

// add first menu item
var usSay = gather.addSay();
usSay.setText('Press 1 for english');
usSay.setLanguage('en-US');

// add second menu item
var nlSay = gather.addSay();
nlSay.setText('Kies 2 voor Nederlands');
nlSay.setLanguage('nl-NL');

// add third menu item
var frSay = gather.addSay();
frSay.setText('Choisissez 3 pour le français. ');
frSay.setLanguage('fr-FR');

// and finish off with an applause
var play = gather.addPlay();
```

**NotifyAction - addGather.addPlay()**

Plays an audio file on the call.

Refer to the NotifyAction addPlay() method for a description of the supported child methods.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PlayAction</td>
<td>Action added to the NotifyAction object. Use the PlayAction object to define the audio file URL and the number of times to loop the audio.</td>
</tr>
</tbody>
</table>

```javascript
// instantiate NotifyAction var notifyAction = new SNC.NotifyAction();

// Create the gather action object var gather = notifyAction.addGather();

// Play an audio file var play = gather.addPlay();
play.setURL('http://www.wavsource.com/snds_2015-04-12_5971820382841326/sfx/applause_y.wav');
```

**NotifyAction - addGather.addSay()**

Defines the text-to-speech to read on the call.

Refer to the NotifyAction `addSay()` method for a description of the supported child methods.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SayAction</td>
<td>Action added to the NotifyAction object. Use the SayAction object to define the text and language to read.</td>
</tr>
</tbody>
</table>
```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// present the user with a menu
var gather = notifyAction.addGather();
gather.setNumberOfDigits(1);  // the user can type 1 digit
gather.setTimeout(20);       // time to enter answer, in seconds

// add first menu item
var gatherSay = gather.addSay();
gatherSay.setText('Press 1 for english');
gatherSay.setLanguage('en-US');
```

**NotifyAction - addGather.setFinishKey(String finishKey)**

Defines the key that the caller inputs to denote the end of their input.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>finishKey</td>
<td>String</td>
<td>Key that denotes the end of caller input.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0-9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• # (default)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• *</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to add a gather action and define the key that denotes the end of the caller input.

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

var gather = notifyAction.addGather();
gather.setNumberOfDigits(4);  // the user can type four digit
gather.setFinishKey('#');    // # or *, useful for > 1 digit
```
**NotifyAction - addGather.setNumberOfDigits(Integer numberOfDigits)**

Defines the number of digits to collect.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numberOfDigits</td>
<td>Integer</td>
<td>Number of digits to collect. Zero is an invalid value.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to add a gather action and define the number of key strokes to collect.

```java
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// present the user with a menu
var gather = notifyAction.addGather();
gather.setNumberOfDigits(4);    // the user can type four digit
gather.setFinishKey('#');       // # or *, useful for > 1 digits
gather.setTimeout(20);          // time to enter answer, in seconds
```

**NotifyAction - addGather.setTimeout(Integer timeout)**

Defines the amount of time after which gathering of input will timeout.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeout</td>
<td>Integer</td>
<td>Number of seconds to wait for caller input before timing out. Default: 10</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
This example shows how to add a gather action and define the input timeout value.

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// present the user with a menu
var gather = notifyAction.addGather();
gather.setNumberOfDigits(4);    // the user can type 1 digit
gather.setFinishKey('#');       // # or *, useful for > 1 digits
gather.setTimeout(20);          // time to enter answer, in seconds
```

**NotifyAction - addHangUp()**

Ends an active phone call.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HangUpAction</td>
<td>Action added to the NotifyAction object.</td>
</tr>
</tbody>
</table>

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// hang up
notifyAction.addHangUp();
```

**NotifyAction - addQueue()**

Queues the call, which puts the call on hold.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QueueAction</td>
<td>Action added to the NotifyAction object. Use the QueueAction object to define the queue name, and queueing or dequeueing behavior.</td>
</tr>
</tbody>
</table>

This example shows how to add the call to the specified queue.

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// queue the call
var queue = notifyAction.addQueue();
queue.setName('my queue');
```

This example shows how to remove the call from the queue.

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// dequeue the call
var queue = notifyAction.addQueue();
queue.setDequeue(true);
```

**NotifyAction - addQueue.setDequeue(Boolean dequeue)**

Removes the call from the current call queue (takes it off of "hold").

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dequeue</td>
<td>Boolean</td>
<td>Flag that indicates whether to remove the current call from the queue.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: remove the call from the queue</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: call is put in the queue</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
This example shows how to remove the call from the queue.

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// dequeue the call
var queue = notifyAction.addQueue();
queue.setDequeue(true);
```

### NotifyAction - addQueue.setName(String name)

Defines the name associated with a queue.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name to associate with the queue.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to define the name of a queue.

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// queue the call
var queue = notifyAction.addQueue();
queue.setName('my queue');
```

### NotifyAction - addPlay()

Plays an audio file on the call.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>PlayAction</td>
<td>Action added to the NotifyAction object. Use the PlayAction object to define the audio file URL and number of times to loop the audio.</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// add a play action
var play = notifyAction.addPlay();
play.setURL('http://www.moviesounds.com/2001/imsorry.wav');
play.setLoop(1);
```

**NotifyAction - addPlay.setLoop(Integer loop)**

Defines the number of times to play (loop through) the audio file.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>loop</td>
<td>Integer</td>
<td>Number of times to play the audio file.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// add a play action
var play = notifyAction.addPlay();
play.setURL('http://www.moviesounds.com/2001/imsorry.wav');
play.setLoop(2);
```

**NotifyAction - addPlay.setURL(String url)**

Defines the URL where to obtain the audio file to play.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>URL of the audio file to play.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// add a play action
var play = notifyAction.addPlay();
play.setURL('http://www.moviesounds.com/2001/imsorry.wav');
play.setLoop(1);```

#### NotifyAction - addRecord()

Adds an action to record the call to the current NotifyAction object.

The recording automatically ends when the call is completed or when a specified terminate is pressed (`setFinishKey()`). The recording is then placed in the notify_record table for the associated call.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example demonstrates how to record a call.

```javascript
// First we initialize NotifyAction
var notifyAction = new SNC.NotifyAction();```
// Call addRecord() of NotifyAction - This returns an object of type RecordAction
var recordAction = notifyAction.addRecord();

// Optional. Define the key that callers use to stop the recording
recordAction.setFinishKey('#'); // Stop the call recording when caller presses the '#' key.

**NotifyAction - addRecord.setFinishKey(String finishKey)**

Defines the key that terminates the recording.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>finishKey</td>
<td>String</td>
<td>Key that terminates the recording.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0-9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• # (default)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• *</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example demonstrates how to record a call and set the call termination key.

```javascript
// First we initialize NotifyAction
var notifyAction = new SNC.NotifyAction();

// Then we call addRecord() of NotifyAction
var recordAction = notifyAction.addRecord();

// Set the key that terminates the recording
recordAction.setFinishKey('#'); // This means that we stop the call recording when user
                                  // presses the '#' key.
```

**NotifyAction - addRecord.setMaxDuration(Integer seconds)**

Defines the maximum length of the recording.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>seconds</td>
<td>Integer</td>
<td>Maximum length of the recording in seconds. Default: 3600</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example demonstrates how to record a call and set the call termination key.

```javascript
// First we initialize NotifyAction
var notifyAction = new SNC.NotifyAction();

// Then we call addRecord() of NotifyAction
var recordAction = notifyAction.addRecord();

// Set the maximum length of the recording
recordAction.setMaxDuration(4800);
```

**NotifyAction - addRecord.setTimeout(Integer timeout)**

Sets the number of seconds of silence, after which, the recording ends.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeout</td>
<td>Integer</td>
<td>Number of seconds of silence on the call, after which the recording ends. Default: 10</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example demonstrates how to record a call and set the recording timeout value.

```javascript
// First we initialize NotifyAction
var notifyAction = new SNC.NotifyAction();
```
// Then we call addRecord() of NotifyAction
var recordAction = notifyAction.addRecord();

// Set the recording timeout value
recordAction.setTimeout(360);

**NotifyAction - addReject()**

Rejects an incoming call.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RejectAction</td>
<td>Action added to the NotifyAction object. Use the RejectAction object to define the reason for rejecting the call.</td>
</tr>
</tbody>
</table>

// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// reject the call
var rejectAction = notifyAction.addReject();
rejectAction.setReason('busy'); // 'busy' or 'rejected'

**NotifyAction - addReject.setReason(String reason)**

Defines the reason why the call was rejected.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reason</td>
<td>String</td>
<td>Reason why the call was rejected. Valid values:</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• busy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• rejected</td>
<td></td>
<td>All other values are ignored.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// reject the call
var rejectAction = notifyAction.addReject();
rejectAction.setReason('busy'); // 'busy' or 'rejected'
```

**NotifyAction - addSay()**

Defines the text-to-speech to read on the call.

Multiple languages are supported with text-to-speech. Available languages depend on the telephony provider.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SayAction</td>
<td>Action added to the NotifyAction object. Use the SayAction object to define the text and language to read.</td>
</tr>
</tbody>
</table>

This example demonstrates reading text in several languages.
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// add a say action to say something in US English
var usSay = notifyAction.addSay();
usSay.setText('Welcome. I can speak English');
usSay.setLanguage('en-US');

// add a say action to say something in Dutch
var nlSay = notifyAction.addSay();
nlSay.setText('Ik spreek ook vloeiend nederlands');
nlSay.setLanguage('nl-NL');

// and german
var deSay = notifyAction.addSay();
deSay.setText('Und ich kann auch deutsch sprechen');
deSay.setLanguage('de-DE');

**NotifyAction - addSay.setLanguage(String language)**

Defines the language in which to speak the text.

Use this method in conjunction with the `setText()` method to define the verbiage to speak.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>language</td>
<td>String</td>
<td>ISO 3166 language code that defines the language in which to speak the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>associated text. For example, &quot;en-US&quot; or &quot;nl-NL&quot;.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example demonstrates reading text in several languages.

// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// add a say action to say something in US English
notifyAction.addSay().setText('Welcome. I can speak english');
notifyAction.addSay().setText('Ik spreek ook vloeiend nederlands');
notifyAction.addSay().setText('Und ich kan auch deutsch sprechen');

**NotifyAction - addSay.setText(String text)**

Defines the text to be read within the current call.

Use this method in conjunction with the setLanguage() method to define the language in which to speak the provided text.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>Text to read aloud within the current call.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example demonstrates reading text in several languages.

// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// add a say action to say something in US English
var usSay = notifyAction.addSay();
usSay.setText('Welcome. I can speak english');
usSay.setLanguage('en-US');

// add a say action to say something in Dutch
var nlSay = notifyAction.addSay();
nlSay.setText('Ik spreek ook vloeiend nederlands');
nlSay.setLanguage('nl-NL');

// and german
var deSay = notifyAction.addSay();
deSay.setText('Und ich kan auch deutsch sprechen');
deSay.setLanguage('de-DE');
nlSay.setText('Ik spreek ook vloeiend nederlands');
nlSay.setLanguage('nl-NL');

// and german
var deSay = notifyAction.addSay();
deSay.setText('Und ich kann auch deutsch sprechen');
deSay.setLanguage('de-DE');

### NotifyAction - addSMS()

**Sends an SMS message.**

ℹ️ **Note:** When using this function with an active call, you do not need to call the `setTo` function on the returned SMSAction object. The SMS is automatically sent to the caller.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMSAction</td>
<td>Action added to the NotifyAction object. Use the SMSAction object to define the message text and the phone number to which to send the message.</td>
</tr>
</tbody>
</table>

// Instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// Define where to send the SMS to
var number = new GlideElementPhoneNumber();
number.setPhoneNumber('+31612345678', true);

// Add an SMS action
var sms = notifyAction.addSMS();
sms.setMessage('Lorem ipsum dolor sit amet, consectetur adipiscing elit.');
sms.setTo(number);
**NotifyAction - addSMS.setMessage(String message)**

Defines the text of the SMS message to send.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Text of the SMS message to send.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// define where to send the sms to
var number = new GlideElementPhoneNumber();
number.setPhoneNumber('+31612345678', true);

// add a SMS action
var sms = notifyAction.addSMS();
sms.setMessage('Lorem ipsum dolor sit amet, consectetur adipiscing elit.');
sms.setTo(number);
```

**NotifyAction - addSMS.setTo(String to)**

Sets the phone number to which to sent the SMS message.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>to</td>
<td>String</td>
<td>E.164-compliant phone number to which to send the SMS message.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
// instantiate NotifyAction
var notifyAction = new SNC.NotifyAction();

// define where to send the sms to
var number = new GlideElementPhoneNumber();
number.setPhoneNumber('+31612345678', true);

// add a SMS action
var sms = notifyAction.addSMS();
sms.setMessage('Lorem ipsum dolor sit amet, consectetur adipiscing elit.');</sms.setTo(number);

NotifyAction - append(NotifyAction action)

Appends the specified NotifyAction object to the current client's NotifyAction object.

There are two types of NotifyActions: terminal and non-terminal. Once you append a terminal action to a client's NotifyAction object, you cannot append any additional actions. Non-terminal actions include:

- PlayAction
- RecordAction
- SayAction
- SMSAction

All others NotifyActions are terminal. If you try to add another NotifyAction after a terminal action, the call will fail.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>NotifyAction</td>
<td>NotifyAction object to append to the NotifyAction object of the current caller.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
NotifyAction - fromJson(String json)

Deserialize a NotifyAction object from a JSON string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>json</td>
<td>String</td>
<td>A JSON string representation of a NotifyAction object.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example demonstrates deserializing a NotifyAction object.

```javascript
var json = ".... some json obtained from toJs...";

// instantiate notify action
var notifyAction = new SNC.NotifyAction();

// deserialize and reconstruct the notify action instance
notifyAction.fromJson(json);
```

This example demonstrates both serializing and deserializing a NotifyAction object.

```javascript
// instantiate notify action
var notifyAction = new SNC.NotifyAction();

// add a queue
var queue = notifyAction.addQueue();
queue.setName('myQueueName');
queue.setDequeue(false);

// serialize to json
var json = notifyAction.toJson();
gs.log('serialization result: ' + json);

// instantiate a new notify action
var newAction = new SNC.NotifyAction();

// deserialize the json generated above
newAction.fromJson(json);
```
// serialize the new object and log the result
newJson = newAction.toJson();
gs.log('new serialization result: ' + newJson);
gs.log('the same: ' + (json == newJson));

Output: *** Script: serialization result: {"fClassName":"NotifyAction","fActions": [{"fClassName":"QueueAction","fDequeue":true,"fQueueName":"myQueueName"}]}
*** Script: new serialization result: {"fClassName":"NotifyAction","fActions": [{"fClassName":"QueueAction","fDequeue":true,"fQueueName":"myQueueName"}]}
*** Script: the same: true

**NotifyAction - setCallRecord(GlideRecord callRecord)**

Defines the Notify call record in which to add subsequent actions.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callRecord</td>
<td>GlideRecord</td>
<td>GlideRecord containing the record of the caller (within the notify_call table) for which to add actions. This caller stays in affect until this method is called again with a different caller.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to set the caller to which you want to add actions.

```java
public NotifyAction runIncomingCallWorkflow(NotifyPhoneNumber notifyPhoneNumber,
   GlideRecord callRecord) throws NoWorkflowConfiguredException,
   NoSuchNotifyGroupRecordException {
    NotifyAction notifyAction = runWorkflow(notifyPhoneNumber, COL_INCOMING_CALL_WF,
       callRecord);
    notifyAction.setCallRecord(callRecord);
    return notifyAction;
}
```

**NotifyAction - toJson()**

Serialize the NotifyAction object to a JSON string.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>JSON representation of this NotifyAction object.</td>
</tr>
</tbody>
</table>

This example demonstrates serializing a NotifyAction object.

```javascript
// instantiate notify action
var notifyAction = new SNC.NotifyAction();

// add one or more notify actions
// ...

// and serialize to json
var json = notifyAction.toJson();
```

This example demonstrates both serializing and deserializing a NotifyAction object.

```javascript
// instantiate notify action
var notifyAction = new SNC.NotifyAction();

// add a queue
var queue = notifyAction.addQueue();
queue.setName('myQueueName');
queue.setDequeue(false);

// serialize to json
var json = notifyAction.toJson();
gs.log('serialization result: ' + json);

// instantiate a new notify action
var newAction = new SNC.NotifyAction();

// deserialize the json generated above
newAction.fromJson(json);

// serialize the new object and log the result
```
newJson = newAction.toJson();

gs.log('new serialization result: ' + newJson);

gs.log('the same: ' + (json == newJson));

Output: *** Script: serialization result: {"fClassName":"NotifyAction","fActions":[{"fClassName":"QueueAction","fDequeue":true,"fQueueName":"myQueueName"}]}

*** Script: new serialization result: {"fClassName":"NotifyAction","fActions":[{"fClassName":"QueueAction","fDequeue":true,"fQueueName":"myQueueName"}]}

*** Script: the same: true

**NotifyClient - Client**

The NotifyClient API allows you use Notify telephony functionality, such as making and receiving calls, from a web browser.

Several NotifyClient methods take a callback function as a parameter. Because NotifyClient calls are made asynchronously, these methods cannot return a value directly. Use the callback function to parse the returned data, such as by assigning variables or making other API calls.

**NotifyClient - addEventListener(String event, Function fn)**

Registers an event handler to listen for changes in a Notify client.

Using this method you can register multiple listeners. Each listener must be a separate method call.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
<td>String</td>
<td>Name of the event to listen for. Instead of passing strings, use the constants defined in SNC.Notify.STD_EVENTS.</td>
</tr>
</tbody>
</table>

- **CALL_START**: call has started and is in progress.
- **CALL_CANCEL**: caller canceled the call.
- **CALL_INIT**: WebRTC connected to a call (incoming or outgoing).
- **CALL_DISCONNECT**: current call has been disconnected.
- **ERROR**: Error occurred. Parameters: message(string), errCode(string)
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>◦ message</td>
<td></td>
<td>error message to display.</td>
</tr>
<tr>
<td>◦ errCode</td>
<td></td>
<td>Optional. Associated error code.</td>
</tr>
</tbody>
</table>

• INCOMING_CALL: Call is coming in. Parameters: from(string), to(string), callId(string), parentId(string), sysId(string), isFromClient(boolean)
  ◦ from: Caller's phone number.
  ◦ to: called phone number.
  ◦ callId: SID of the call.
  ◦ parentId: parent notify_call reference. If skipParentId is set to true, this parameter should not be passed.
  ◦ sysId: WebRTC-to-WebRTC calls only. Unique identifier (sys_id) of the caller.
  ◦ isFromClient: WebRTC-to-WebRTC calls only. Flag that indicates whether the call is from another WebRTC client.

• CALL_MUTE: client is muted.
• CALL_UNMUTE: client is unmuted.
• OFFLINE: WebRTC session is not active.
• ONLINE: WebRTC session is ready. Must be set after calling the init() method.

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Function to use to de-register a listener.</td>
</tr>
</tbody>
</table>

This example shows how to register multiple listeners.

```javascript
jQuery(function () {

  var notifyConfig = {
    autoLoadScriptResources: true // This will take care of auto loading the JS resources needed by the client (if any)
  };
```

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
var client = new SNC.Notify.Client(notifyConfig, true); // The second argument ensures that the proper client for the given number is auto determined
client.setCallerId('valid_notify_long_number', function () {
    // This is called after the client has been determined.

    if (!notifyConfig.vendor) // Means this number has no compatible client
        return;

    client.addEventListener(SNC.Notify.STD_EVENTS.ONLINE, function () {
        // Ability to call is available
    });
    client.addEventListener(SNC.Notify.STD_EVENTS.OFFLINE, function () {
        // Ability to call is _not_ available right now
    });
    client.addEventListener(SNC.Notify.STD_EVENTS.ERROR, function (msg, code) {
        // Some error happened
    });
    //... register other event handlers here

    client.init(); // This is important to call this.
});
});

This example shows how to de-register a listener.

var dereg = notifyClient.addEventListener(SNC.Notify.STD_EVENTS.ONLINE, function () {
    ...
});
dereg();
// The event listener function is no longer triggered.

NotifyClient - call(Object identifier)
Calls the specified phone number or the phone number associated with a specified user.

_note:_ When checking the status of a call/connection, always compare against the constants provided by SNC.Notify.Status.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>identifier</td>
<td>Object</td>
<td>JSON object that contains either a phone number to call or the sys_id of a WebRTC user. Passing a user sys_id</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>causes the call to be made through browser-to-browser communication.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You can obtain the user sys_id from the Notify WebRTC Session table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> If you provide both a phone number and user sys_id, the method only uses the phone number.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example demonstrates passing a phone number as the function parameter.

```javascript
notifyClient.call({
    phoneNumber: "+18001112223"
});
```

This example demonstrates passing a user record sys_id as the function parameter.

```javascript
notifyClient.call({
    userId: "6816f79cc0a8016401c5a33be04be441"
});
```

This example shows a button click handler.

```javascript
$j("#pickupCallBtn").on("click", function() {
    notifyClient.hangupCall();
});
```

This example shows an event handler.

```javascript
onConnect: function(status) {
    // webRTC receives a call connection event (incoming or outgoing).
    if (status == SNC.Notify.Status.OPEN) {
        setStatus(getTimeStamp() + " -- Successfully established call");
        showHangupButton();
    }
},
```
NotifyClient - Client(Object notifyConfig, Boolean initializeVendorClientLazily)

Instantiates a new Notify WebRTC Client object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>initializeVendorClientLazily</td>
<td>Boolean</td>
<td>Flag that indicates whether to use the autoSelectVendorCallback function passed in the setCallerId() method to automatically set the caller's associated vendor (notifyConfig.vendor does not need to be defined in the constructor).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Default. Do not use the autoSelectVendorCallback function to set the caller's vendor. The vendor must be set in the constructor.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Use the autoSelectVendorCallback function to define the vendor when the caller ID is set.</td>
</tr>
<tr>
<td>notifyConfig</td>
<td>Object</td>
<td>JSON object that contains the configuration settings for the Notify WebRTC Client.</td>
</tr>
<tr>
<td>notifyConfig.autoLoadScriptResources</td>
<td>Boolean</td>
<td>Flag that indicates how to load the core JS library needed by the vendor client.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Default. Use vendor specific codes to load the required vendor JS library (enables backwards compatibility).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Use notifyClient.js to load the core JS library.</td>
</tr>
<tr>
<td>notifyConfig.callerId</td>
<td>Number</td>
<td>Registered Notify number to use. Do not directly set this value. Use the method...</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyClient.setCallerID()</td>
<td></td>
<td><code>notifyClient.setCallerID()</code> to set this value.</td>
</tr>
<tr>
<td>notifyConfig.forceRefreshToken</td>
<td>Boolean</td>
<td>Flag that indicates whether to auto-renew expired client tokens.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not automatically renew client tokens when they expire.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Default. Automatically renew client tokens when they expire.</td>
</tr>
<tr>
<td>notifyConfig.skipParentId</td>
<td>Boolean</td>
<td>Flag that indicates whether to immediately invoke the onIncoming caller for incoming calls.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Default. Do not immediately invoke the onIncoming event handler.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Immediately invoke the onIncoming event handler. By setting this flag, if there is another call, where the <code>&lt;Dial&gt;&lt;Client&gt;</code> Twiml caused the incoming call, then setting this flag causes the system to auto poll the backend. This auto poll obtains the parent notify_call reference.</td>
</tr>
<tr>
<td>notifyConfig.vendor</td>
<td>Constant</td>
<td>Vendor to which the caller belongs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SNC.Notify.Vendor.TWILIO_DIRECT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SNC.Notify.Vendor.TWILIO (older, deprecated Twilio driver)</td>
</tr>
</tbody>
</table>

The following example shows how to create the NotifyClient constructor, register various event listeners, and initialize the client driver.

```javascript
jQuery(function () {

```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
```javascript
var notifyConfig = {
    autoLoadScriptResources: true // This will take care of auto loading the JS resources needed by the client (if any)
};
var client = new SNC.Notify.Client(notifyConfig, true); // The second argument ensures that the proper vendor for the given number is auto determined
client.setCallerId('valid_notify_long_number', function () {
    // This is called after the vendor has been determined.
    
    if (!notifyConfig.vendor) // Means this number has no compatible vendor
        return;

    client.addEventListener(SNC.Notify.STD_EVENTS.ONLINE, function () {
        // Ability to call is available
    });
    client.addEventListener(SNC.Notify.STD_EVENTS.OFFLINE, function () {
        // Ability to call is _not_ available right now
    });
    client.addEventListener(SNC.Notify.STD_EVENTS.ERROR, function (msg, code) {
        // Some error happened
    });
    //... register other event handlers here
    //Show UI elements which can be used to invoke client.call() and other APIs
    client.init(); // This is important to call this.
});
```

**NotifyClient - destroy()**

Kills the current Notify client, rendering it unusable.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>
**NotifyClient - forwardCall(Object argument)**

Forwards an ongoing incoming or outgoing phone call to either a different phone number or a different WebRTC client.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>argument</td>
<td>Object</td>
<td>JSON object that contains the necessary information for forwarding the call to either a phone number or a WebRTC client (user sys_id). You can obtain this sys_id from the Notify WebRTC Session table.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example demonstrates forwarding a call to a different phone number. The dtmf attribute allows you to send DTMF dial tones to the receiving number.

```javascript
var arg = {
    type: "number",
    id: "+17012345678",
    dtmf: "1234"
}
client.forwardCall(arg);
```

This example demonstrates forwarding a call to a different Notify client.

```javascript
var arg = {
    type: "userId",
    id: "6816f79cc0a8016401c5a33be04be441"
}
client.forwardCall(arg);
```

**NotifyClient - getAvailableClients(Function callback)**

Returns a list of clients available to accept calls.

This method excludes the current client from the list. The equivalent `Notify-getAvailableClients()` method does not filter any user.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callback</td>
<td>Function</td>
<td>Function to use to parse the list of clients. This function accepts a single parameter, an array of JSON objects with the following format:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[{sys_id: &quot;...&quot;, // user's sys_id} name: &quot;...&quot; // user's name}</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**NotifyClient - getParentId(String callId, Function callback)**

Returns the parent call identifier for a specified call identifier, if one exists.

Depending on the telephony provider, there may be a delay before the parent call identifier is returned; therefore you must provide a callback function.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callId</td>
<td>String</td>
<td>Unique identifier of the call for which to return the parent call identifier.</td>
</tr>
<tr>
<td>callback</td>
<td>Function</td>
<td>Function that obtains the JSON object that contains either the parent call identifier or an error message if the identifier could not be obtained after several tries.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Parent call identifier.</td>
</tr>
</tbody>
</table>

This example shows how to use this method to obtain the parent call identifier.
This example shows the contents of the jsonObj parameter.

```javascript
{
  parentId: "xyz",
  error: "msg"
}
```

**NotifyClient - getStatus()**

Returns the normalized status of the current call.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Current status of the call. The values returned by the telephony provider API are normalized by replacing the returned driver value with its equivalent value as defined in SNC.Notify.Status.</td>
</tr>
</tbody>
</table>

This example shows how to obtain the status of the current Notify client.

```javascript
clientStatus = notifyClient.getStatus();
```

**NotifyClient - hangupCall()**

End the current call.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to hang up a call.

```
$\$(\#\text{pickupCallBtn}).on(\"click\", function() {
    notifyClient.hangupCall();
});
```

**NotifyClient - init()**

Initializes the client driver.

For example, when using the Twilio client, it invokes the method `Twilio.Device.setup()`. Call this method after the user has interacted with the page. This initialization process is asynchronous, therefore, you must provide an ONLINE event handler. This handler is called when the setup process is complete and the system is ready to take or make calls.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to initialize the Notify client.

```
$\$(function() {
    notifyClient = new SNC.Notify.Client( notifyConfig );
    notifyClient.setCallerId( '+31858889170' );
    notifyClient.init();
});
```

**NotifyClient - mute(Boolean muted)**

Mute or unmute the current client.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>muted</td>
<td>Boolean</td>
<td>Mutes or unmutes the current call.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: (or any non-true value) unmutes the current call.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: mutes the current call.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to send mute the current call.

```
notifyClient.mute( "true" );
```

### NotifyClient - pickUpCall()

Answers and connects to an incoming call from a WebRTC client.

Call this method when there is a notification of an incoming call.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to pickup a call.

```
$\$("#pickUpCallBtn").on("click", function() {
    notifyClient.pickupCall();
});
```
**NotifyClient - sendDtmf(String digits)**
Send one or more DTMF-valid digits over the current call.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>digits</td>
<td>String</td>
<td>One or more DTMF-valid digits.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to send DTMF signals to the current call.

```java
notifyClient.SendDtmf( "1246AF" ) {} );
```

**NotifyClient - setClientAvailable(Boolean available)**
Sets the availability of an active WebRTC client agent.

This type of availability is different than an agent being in a call. In this case, an active WebRTC client may be connected and not on a call, but may not want to receive calls.

Calling this method updates the **Available** field value on the Notify Client Connected Session [notify_client_session] record associated with this client session. You can get a list of available clients using the **getAvailableClients()** method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>available</td>
<td>boolean</td>
<td>Flag that indicates whether an active WebRTC client wants to receive calls.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: client does not want to receive calls.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: client does want to receive calls.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

NotifyClient - setCallerId(String value, Function autoSelectVendorCallback)

Sets the caller ID for the current client session.

You can change or update the caller ID at any time however, the caller ID must belong to the same vendor.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>Phone number to use to make and receive calls.</td>
</tr>
<tr>
<td>autoSelectVendorCallback</td>
<td>Function</td>
<td><strong>Optional.initializeVendorClientLazily</strong> must be set to &quot;true&quot; in the constructor to use this function, otherwise an error is thrown. Name of the callback function to call once the vendor is automatically set for the specified phone number. With this option, the vendor does not need to be specified in the constructor (notifyConfig.vendor). Auto vendor selection is an asynchronous operation. Therefore, this callback is required to indicate when it is safe to call notifyConfig.init(), as this method requires that the vendor be set before it is called. In addition, you must also check if notifyConfig.vendor has been set in the callback to ensure that a vendor has been specified.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to set the caller ID. This example assumes that the vendor is set in the constructor.

```javascript
$j(function() {
    notifyClient = new SNC.Notify.Client( notifyConfig );
    notifyClient.setCallerId( '+31858889170' );
    notifyClient.init();
});
```

**NotifyConferenceUtil - Scoped, Global**

The NotifyConferenceUtil API provides methods to manage Notify conference calls and SMS messages for various telephony service providers, such as Zoom and WebEx.

Using the NotifyConferenceUtil API you can:

- Create new conference calls
- Add participants by phone number or user ID
- Remove participants from a conference call
- Mute participants in a conference call
- Unmute participants in a conference call
- Obtain the capabilities of a specified service provider
- End a conference call

You can use this API in both scoped and global scripts. To use this API you must activate the Conference Notify plugin (com.snc.notify) which requires a separate subscription. For details on activating this plugin, see Activate Notify.

**NotifyConferenceUtils - addToConferenceByPhoneNumber(String toNumber, GlideRecord confGR)**

Adds a participant to a specified conference call using their phone number to identify the participant.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>toNumber</td>
<td>String</td>
<td>Phone number of the participant to add to the conference call.</td>
</tr>
<tr>
<td>confGR</td>
<td>GlideRecord</td>
<td>GlideRecord of the conference call to which to add the specified participant. These records are located in the Notify Conference Call [notify_conference_call] table.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Results of the conference action.</td>
</tr>
<tr>
<td></td>
<td><strong>&lt;action&gt;.status</strong>: Status of the conference action.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>• Valid values:</td>
</tr>
<tr>
<td></td>
<td>◦ true: Conference action succeeded</td>
</tr>
<tr>
<td></td>
<td>◦ false: Conference action failed</td>
</tr>
<tr>
<td></td>
<td><strong>&lt;action&gt;.successMessages</strong>: If status is true, success message(s), else empty.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
<tr>
<td></td>
<td><strong>&lt;action&gt;.warnMessages</strong>: If status is false, any warning messages thrown during processing.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
<tr>
<td></td>
<td><strong>&lt;action&gt;.errorMessages</strong>: If status is false, any error messages thrown during processing.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
</tbody>
</table>

## Example

```javascript
function () {
  var confGR = new GlideRecord('notify_conference_call');
  confGR.get('76d3364d0b5133008e64aabcb4673a6d');
}  ```
var confUtils = new NotifyConferenceUtils();
var actionResult = confUtils.addToConferenceByPhoneNumber("+917799555331", confGR)
if (actionResult.status)
  gs.info('Participant has been added to conference');
else {
  gs.info('join operation failed');
  actionResult.warnMessages.forEach(function (msg) {
    gs.info(msg);
  });
  actionResult.errorMessages.forEach(function (msg) {
    gs.info(msg);
  })
}
})

NotifyConferenceUtils - addToConferenceByUserId(String userId, GlideRecord confGR)

Adds a participant to the conference call referenced by the passed in GlideRecord using their unique user identifier.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userId</td>
<td>String</td>
<td>Sys ID of the participant to add to the specified conference call. This information is located in the User [sys_user] table.</td>
</tr>
<tr>
<td>confGR</td>
<td>GlideRecord</td>
<td>GlideRecord of the conference call to add the specified participant. These records are located in the Notify Conference Call [notify_conference_call] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Results of the conference action.</td>
</tr>
<tr>
<td>&lt;action&gt;.status</td>
<td>Status of the conference action.</td>
</tr>
<tr>
<td>• Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>• Valid values:</td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>◦ true:</td>
<td>Conference action succeeded</td>
</tr>
<tr>
<td>◦ false:</td>
<td>Conference action failed</td>
</tr>
</tbody>
</table>

<action>.successMessages: If `status` is true, success message(s), else empty.
- Data type: Array of Strings

<action>.warnMessages: If `status` is false, any warning messages thrown during processing.
- Data type: Array of Strings

<action>.errorMessages: If `status` is false, any error messages thrown during processing.
- Data type: Array of Strings

**Example**

```javascript
(function () {
    var confGR = new GlideRecord('notify_conference_call');
    confGR.get('76d3364d0b5133008e64aabcb4673a6d');

    var confUtils = new NotifyConferenceUtils();
    var actionResult = confUtils.addToConferenceByUserId(gs.getUserID(), confGR);

    if (actionResult.status)
        gs.info('Participant has been added to conference');
    else {
        gs.info('join operation failed');
        actionResult.warnMessages.forEach(function (msg) {
            gs.info(msg);
        });
        actionResult.errorMessages.forEach(function (msg) {
            gs.info(msg);
        });
    }
})();
```
**NotifyConferenceUtils - doConferenceAction(String action, Object data)**

Performs the specified conference call action, such as starting/ending a conference call or joining, removing, muting, or unmuting participants from a conference call.

You can start a new conference call and add participants within a single call to this method or call the method multiple times to start the call and then manage participants separately. In addition, through the passed in data object, you can configure the method to:

- Save pointers in the conference call record to the specific record (source record), such as an incident or problem, that is the topic of discussion for the conference call.
- Allow/disallow multiple conference calls for a source record.
- Automatically log the participants that were in the conference call in the "Work Notes" field of the source record.
- Have a message read aloud when a participant answers an outgoing call from the conference.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>String</td>
<td>Defines the conference call action to perform.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The following are the available conference call actions:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• start: Starts the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• end: Terminates the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• join: Adds the participant specified in the <code>data.items</code> array to the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• multiJoin: Adds the participants specified in the <code>data.items</code> array to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the conference call identified in <code>data.confId</code></td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• selfJoin:</td>
<td></td>
<td>Adds the currently logged in user to the conference call (no entry in <code>data.items</code> required.)</td>
</tr>
<tr>
<td>• kick:</td>
<td></td>
<td>Removes the participant specified in the <code>data.items</code> array from the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>• multiKick:</td>
<td></td>
<td>Removes the participants specified in the <code>data.items</code> array from the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>• mute:</td>
<td></td>
<td>Mutes the participant specified in the <code>data.items</code> array on the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>• multiMute:</td>
<td></td>
<td>Mutes the participants specified in the <code>data.items</code> array on the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>• unmute:</td>
<td></td>
<td>Unmutes the participant specified in the <code>data.items</code> array from the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>• multiUnmute:</td>
<td></td>
<td>Unmutes the participants specified in the <code>data.items</code> array from the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>data</td>
<td>Object</td>
<td>Object that describes the conference call.</td>
</tr>
<tr>
<td>data.addToWorkNotes</td>
<td>Boolean</td>
<td>Flag that indicates whether to add information about the participants that were included in the conference call in the work notes field of the associated record. For this functionality to work, you must also specify values in the <code>data.table</code>.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>and <code>data.sysId</code> parameters. These parameters identify the record in which to add the work notes. Default: false Actions for which this parameter is valid:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• start</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• join</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• multiJoin</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• selfJoin</td>
</tr>
<tr>
<td>data.allowMulticonference</td>
<td>Boolean</td>
<td>Flag that indicates whether to allow multiple conference calls for a specific record at one time. For this functionality to work, you must also specify values in the <code>data.table</code> and <code>data.sysId</code> parameters. These parameters identify the record that is allowed to have multiple conference calls. Default: false Actions for which this parameter is valid:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• start</td>
</tr>
<tr>
<td>data.confId</td>
<td>String</td>
<td>Sys ID of the conference call. The conference Sys ID is located in the Notify Conference Call [notify_conference_call] table. Actions for which this parameter is required:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• end</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• join</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• multiJoin</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• selfJoin</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data.fromNumber</td>
<td>String</td>
<td>Service provider number to call into for the conference call. Locate this value in the Number or Phone number column of the Notify Phone Number [notify_number] table. Actions for which this parameter is required: • start</td>
</tr>
<tr>
<td>data.isNewConference</td>
<td>Boolean</td>
<td>Flag that indicates whether this is a new or an existing conference call. Valid values: • true: New conference call • false: Existing conference call Default: false Actions for which this parameter is valid: • start</td>
</tr>
<tr>
<td>data.items</td>
<td>Array</td>
<td>Information for each participant to include in the conference call. Valid array values: • id: Sys ID of user; located in the User [sys_User] table. Valid actions: join, multiJoin, start</td>
</tr>
</tbody>
</table>

**Note:** Participant actions such as mute, unmute, and kick do not require this parameter to be set as the method obtains this information from the Notify Conference Call Participant [notify_participant] table.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyParticipantId</td>
<td>Sys ID</td>
<td>Sys ID of the Notify participant; located in the Notify Participant [notify_participant] table. Valid actions: join, kick, multiJoin, mute, start, unmute.</td>
</tr>
<tr>
<td>phoneNumber</td>
<td></td>
<td>Phone number of the participant. If this value is passed in conjunction with either the id or notifyParticipantId, this value supersedes the phone numbers in the user/participant record and is used to place the call. Valid actions: join, multiJoin, start.</td>
</tr>
<tr>
<td>email</td>
<td></td>
<td>Email address of the participant. Valid actions: join, multiJoin, start.</td>
</tr>
<tr>
<td>data.message</td>
<td>String</td>
<td>Message that is read aloud when a user answers the call, such as, &quot;P1 incident has been created please login to instance.&quot; Actions for which this parameter is valid:</td>
</tr>
<tr>
<td>data.serviceProvider</td>
<td>String</td>
<td>Required. Name of conference service provider, such as Zoom or Webex. Actions for which this parameter is required: all</td>
</tr>
<tr>
<td>data.sysId</td>
<td>String</td>
<td>Sys ID of the source record to associate with the conference call.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>For example, if a conference call is held to discuss a specific incident or problem, put the Sys ID of the incident or problem record in this value. This Sys ID is stored in the Source column of the NotifyConference Call [notify_conference_call] table and can later be tracked. This parameter is used in conjunction with the data.Table, data.addToWorkNotes, and allowMulticonference parameters. Actions for which this parameter is valid:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• start</td>
</tr>
<tr>
<td>data.table</td>
<td>String</td>
<td>Table that contains the source record to associate with the conference call. A source record can be any record, such as an &quot;incident&quot; or &quot;problem&quot;, that is the topic of discussion in the conference call. This table name is stored in the Table column of the NotifyConference Call [notify_conference_call] table and can be tracked. This parameter is used in conjunction with the data.sysId, data.addToWorkNotes, and allowMulticonference parameters. Actions for which this parameter is valid:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• start</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Results of the conference action.</td>
</tr>
<tr>
<td></td>
<td>&lt;action&gt;.status: Status of the conference action.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Data type: Boolean</td>
<td>• Valid values:</td>
</tr>
<tr>
<td></td>
<td>◦ true: Conference action succeeded</td>
</tr>
<tr>
<td></td>
<td>◦ false: Conference action failed</td>
</tr>
<tr>
<td>&lt;action&gt;.successMessages: If status is true, success message(s), else empty.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
<tr>
<td>&lt;action&gt;.warnMessages: If status is false, any warning messages thrown during processing.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
<tr>
<td>&lt;action&gt;.errorMessages: If status is false, any error messages thrown during processing.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
</tbody>
</table>

Example

```javascript
(function (confId) {
  var confUtils = new NotifyConferenceUtils();
  var data = confUtils.getConferenceInputDataTemplate();
  data.table = 'incident';
  data.sysId = '1234';
  data.addToWorkNotes = false;
  data.confId = confId;
  data.message = 'p1 incident has been created';
  data.fromNumber = 'twilio/Telephony driver number';
  data.items.push({ id: 'user3SysId', phoneNumber: '+917799555332' });
  data.items.push({ id: 'user4SysId', email: 'yln99518@gmail.com' });

  var result = confUtils.doConferenceAction('start', data);
  if (result.status) {
    gs.info('Start conference action succeeded');
  } else
    gs.info('Start conference action failed');
});
```
result.errorMessages.forEach(function (msg) {
    gs.info(msg);
});
result.warnMessages.forEach(function (msg) {
    gs.info(msg);
});
result.successMessages.forEach(function (msg) {
    gs.info(msg);
});
})("activeConfSysId");

NotifyConferenceUtils - getConferenceInputDataTemplate()

Returns a JSON data template to use with the doConferenceAction() method. Using this template automatically structures the data object so that you don't have to manually create it.

Call this method prior to calling the doConferenceAction() method. For the desired conference call action, set the desired parameters within the template, and then pass the template in the doConferenceAction() call. For additional information on the valid parameters for each action, see doConferenceAction().

>Note: This is a helper method. You can also manually construct this object and pass that object into the doConferenceAction() method and have the same outcome.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Object that describes the conference call.</td>
</tr>
<tr>
<td>data.addToWorkNotes</td>
<td>Flag that indicates whether to add information about the participants that were included in the conference call in the work notes field of the associated record.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>For this functionality to work, you must also specify values in the <code>data.table</code> and <code>data.sysId</code> parameters. These parameters identify the record in which to add the work notes.</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td></td>
<td>Actions for which this parameter is valid:</td>
</tr>
<tr>
<td></td>
<td>• start</td>
</tr>
<tr>
<td></td>
<td>• join</td>
</tr>
<tr>
<td></td>
<td>• multiJoin</td>
</tr>
<tr>
<td></td>
<td>• selfJoin</td>
</tr>
<tr>
<td><code>data.allowMultconference</code></td>
<td>Flag that indicates whether to allow multiple conference calls for a specific record at one time.</td>
</tr>
<tr>
<td></td>
<td>For this functionality to work, you must also specify values in the <code>data.table</code> and <code>data.sysId</code> parameters. These parameters identify the record that is allowed to have multiple conference calls.</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td></td>
<td>Actions for which this parameter is valid:</td>
</tr>
<tr>
<td></td>
<td>• start</td>
</tr>
<tr>
<td><code>data.confId</code></td>
<td>Sys ID of the conference call.</td>
</tr>
<tr>
<td></td>
<td>The conference Sys ID is located in the Notify Conference Call <code>[notify_conference_call]</code> table.</td>
</tr>
<tr>
<td></td>
<td>Actions for which this parameter is required:</td>
</tr>
<tr>
<td></td>
<td>• end</td>
</tr>
<tr>
<td></td>
<td>• join</td>
</tr>
<tr>
<td></td>
<td>• multiJoin</td>
</tr>
<tr>
<td></td>
<td>• selfJoin</td>
</tr>
</tbody>
</table>
 Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data.fromNumber</td>
<td>Service provider number to call into for the conference call.</td>
</tr>
<tr>
<td></td>
<td>Locate this value in the Number or Phone number column of the Notify Phone Number [notify_number] table.</td>
</tr>
<tr>
<td></td>
<td>Actions for which this parameter is required:</td>
</tr>
<tr>
<td></td>
<td>• start</td>
</tr>
<tr>
<td>data.isNewConference</td>
<td>Flag that indicates whether this is a new or an existing conference call.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: New conference call</td>
</tr>
<tr>
<td></td>
<td>• false: Existing conference call</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td></td>
<td>Actions for which this parameter is valid:</td>
</tr>
<tr>
<td></td>
<td>• start</td>
</tr>
<tr>
<td>data.items</td>
<td>Information for each participant to include in the conference call.</td>
</tr>
<tr>
<td></td>
<td>Valid array values:</td>
</tr>
<tr>
<td></td>
<td>• id: Sys ID of user; located in the User [sys_User] table.</td>
</tr>
<tr>
<td></td>
<td>Valid actions: join, multiJoin, start</td>
</tr>
<tr>
<td></td>
<td>• notifyParticipantId: Sys ID of the Notify participant; located in the Notify Participant [notify_participant] table.</td>
</tr>
<tr>
<td></td>
<td>Valid actions: join, kick, multiJoin, mute, start, unmute</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>• phoneNumber: Phone number of the participant. If this value is passed in conjunction with either the <strong>id</strong> or <strong>notifyParticipantId</strong>, this value supersedes the phone numbers in the user/participant record and is used to place the call.</td>
</tr>
<tr>
<td></td>
<td>Valid actions: join, multiJoin, start</td>
</tr>
<tr>
<td>• email: Email address of the participant.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Valid actions: join, multiJoin, start</td>
</tr>
<tr>
<td>data.message</td>
<td>Message that is read aloud when a user answers the call, such as, &quot;P1 incident has been created please login to instance.&quot;</td>
</tr>
<tr>
<td></td>
<td>Actions for which this parameter is valid:</td>
</tr>
<tr>
<td></td>
<td>• start</td>
</tr>
<tr>
<td></td>
<td>• join</td>
</tr>
<tr>
<td></td>
<td>• multiJoin</td>
</tr>
<tr>
<td>data.serviceProvider</td>
<td>Required. Name of conference service provider, such as Zoom or Webex.</td>
</tr>
<tr>
<td></td>
<td>Actions for which this parameter is required:</td>
</tr>
<tr>
<td></td>
<td>• all</td>
</tr>
<tr>
<td>data.sysId</td>
<td>Sys ID of the source record to associate with the conference call.</td>
</tr>
<tr>
<td></td>
<td>For example, if a conference call is held to discuss a specific incident or problem, put the Sys ID of the incident or problem record in this value. This Sys ID is stored in the Source column of the NotifyConference Call [notify_conference_call] table and can later be tracked.</td>
</tr>
<tr>
<td></td>
<td>This parameter is used in conjunction with the <strong>data.Table</strong>, <strong>data.addToWorkNotes</strong>, and <strong>allowMulticonference</strong> parameters.</td>
</tr>
<tr>
<td></td>
<td>Actions for which this parameter is valid:</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data.table</td>
<td>Table that contains the source record to associate with the conference call. A source record can be any record, such as an &quot;incident&quot; or &quot;problem&quot;, that is the topic of discussion in the conference call. This table name is stored in the Table column of the NotifyConference Call [notify_conference_call] table and can be tracked. This parameter is used in conjunction with the data.sysId, data.addToWorkNotes, and allowMulticonference parameters. Actions for which this parameter is valid: start</td>
</tr>
</tbody>
</table>

#### Example

```javascript
(function (confId) {
    var confUtils = new NotifyConferenceUtils();
    var data = confUtils.getConferenceInputDataTemplate();
    data.confId = confId;
    var result = confUtils.doConferenceAction('end', data);
    if (result.status) {
        gs.info('Conference call has been ended');
    } else
        gs.info('End conference call action failed');

    result.errorMessages.forEach(function (msg) {
        gs.info(msg);
    });
    result.warnMessages.forEach(function (msg) {
        gs.info(msg);
    });
    result.successMessages.forEach(function (msg) {
        gs.info(msg);
    });
}
```
NotifyConferenceUtils - getServiceProvidersCapabilities()

Returns the capabilities of all telephony service provider drivers in the instance.

Possible capabilities include:

- **archive**: archives the conference after it ends
- **beepOnLeave**: plays a "beep" tone when a participant leaves the conference call
- **beepOnJoin**: plays a "beep" tone when a participant joins the conference call
- **end**: ends the identified conference call
- **filesharing**: allows file sharing between participants
- **join**: adds a participant to a conference call
- **kick**: removes a participant from a conference call
- **multiJoin**: adds multiple participants to a conference call
- **multiKick**: removes multiple participants from a conference call
- **muteOnJoin**: mutes a participant when they initially join a conference call
- **multiUnmute**: unmutes multiple participants for a conference call
- **record**: records conference calls
- **recording**: provides an on-screen indicator when the conference call is being recorded
- **screenSharing**: allows participant screens to be shared with the group
- **selfJoin**: adds the current logged in user to a conference call
- **speaking**: provides an on-screen message as to who is currently speaking
- **start**: starts the identified conference call
- **unmute**: unmutes a participant in a conference call

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Key-value pairs of the status of each driver capability.</td>
</tr>
</tbody>
</table>

Valid values:

- `isSupported = 0`: capability is not supported
- `isSupported = 1`: capability is supported

List driver capabilities for all telephony service provider drivers in the instance.

```javascript
(function () {
  var confUtils = new NotifyConferenceUtils();
  var providerToCapability = confUtils.getServiceProvidersCapabilities();
  for (var provider in providerToCapability) {
    gs.info('{0} supported capabilities 

', provider);
    var capabilities = providerToCapability[provider]
    for (var cap in capabilities)
      if(confUtils.isActionSupported(capabilities[cap].isSupported))
        gs.info('"{0}" action supported', cap);
      else
        gs.info("'{0}' action is not supported by this conference driver", cap);
  }
})();
```

Output: The method returns a node similar to the following for each active telephony service provider within the instance.

```json
{
  "Telephony": {
    "start": {
      "isSupported": 1,
      "meta": {}
    },
    "end": {
      "isSupported": 1,
      "meta": {}
    },
    "selfJoin": {
      "isSupported": 1,
      "meta": {}
    },
    "join": {
```
"isSupported": 1,
"meta": { }
},
"multiJoin": {
  "isSupported": 1,
  "meta": { }
},
"mute": {
  "isSupported": 1,
  "meta": { }
},
"multiMute": {
  "isSupported": 1,
  "meta": { }
},
"unmute": {
  "isSupported": 1,
  "meta": { }
},
"multiUnmute": {
  "isSupported": 1,
  "meta": { }
},
"kick": {
  "isSupported": 1,
  "meta": { }
},
"multiKick": {
  "isSupported": 1,
  "meta": { }
},
"record": {
  "isSupported": 0,
  "meta": { }
},
"speaking": {
  "isSupported": 0,
  "meta": { }
},
"recording": {
  "isSupported": 0,
  "meta": { }
},
"screenSharing": {
NotifyConferenceUtils - isActionSupported(Number action)

Determines whether a Notify conference action is supported by a telephony service provider.

To use this method, you must first call the getServiceProviderCapabilities() method. This method returns an object that contains information about the availability of each possible Notify conference action for each service provider configured in your instance.

For example:

```json
{
   "Telephony": {
      "start": {
         "isSupported": 1,
         "meta": {}
      },
      "end": {
         "isSupported": 0,
         "meta": {}
      }
   }
}
```
{"isSupported": 0,
  "meta": {}},
"recording": {
  "isSupported": 0,
  "meta": {}},
"screenSharing": {
  "isSupported": 0,
  "meta": {}},
"fileSharing": {
  "isSupported": 0,
  "meta": {}},
"archive": {
  "isSupported": 0,
  "meta": {}},
"muteOnJoin": {
  "isSupported": 0,
  "meta": {}},
"beepOnJoin": {
  "isSupported": 0,
  "meta": {}},
"beepOnLeave": {
  "isSupported": 0,
  "meta": {}}
}

## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>Number</td>
<td>Value of the <strong>isSupported</strong> parameter returned by the <code>getServiceProvidersCapabilities()</code> method for a specific action and service provider.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> Although the <code>isSupported</code> value may appear to be a Boolean, it is actually a Number. Do not try and evaluate the capabilities as Boolean values. Use this method as the associated values may be expanded in future versions.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean| Flag that indicates whether the telephony service provider supports the specified action. Valid values:  
• true: action is supported by the service provider  
• false: action is not supported by the service provider |

Example

```javascript
(function () {
  var confUtils = new NotifyConferenceUtils();
  var providerToCapability = confUtils.getServiceProvidersCapabilities();
  for (var provider in providerToCapability) {
    gs.info('{0} supported capabilities 

', provider);
    var capabilities = providerToCapability[provider]
    for (var cap in capabilities)
      if (confUtils.isActionSupported(capabilities[cap].isSupported))
        gs.info('"{0}" action supported', cap);
      else
        gs.info('"{0}" action is not supported by this conference driver", cap);
  }
})();
```

**NotifyConferenceUtils - kickByParticipantGR(GlideRecord notifyParticipantGR)**

Removes the participant associated with the passed in GlideRecord from the current conference call.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyParticipantGR</td>
<td>GlideRecord</td>
<td>GlideRecord object of the participant to remove from the conference call. These records are located in the Notify Participant [notify_participant] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Results of the conference action.</td>
</tr>
<tr>
<td></td>
<td>&lt;action&gt;.status: Status of the conference action.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>• Valid values:</td>
</tr>
<tr>
<td></td>
<td>○ true: Conference action succeeded</td>
</tr>
<tr>
<td></td>
<td>○ false: Conference action failed</td>
</tr>
<tr>
<td></td>
<td>&lt;action&gt;.successMessages: If status is true, success message(s), else empty.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
<tr>
<td></td>
<td>&lt;action&gt;.warnMessages: If status is false, any warning messages thrown during processing.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
<tr>
<td></td>
<td>&lt;action&gt;.errorMessages: If status is false, any error messages thrown during processing.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
</tbody>
</table>

Example

```javascript
(function () {
    var confUtils = new NotifyConferenceUtils();
    var participantGR = new GlideRecord('notify_participant'); // Participant record contains conf call ID
    participantGR.get('validParticipantSysId');
})();
```
var actionResult = confUtils.kickByParticipantGR(participantGR);
if (actionResult.status)
    gs.info('Participant has been kicked out of conference');
else {
    gs.info('kick operation failed');
    actionResult.warnMessages.forEach(function (msg) {
        gs.info(msg);
    });
    actionResult.errorMessages.forEach(function (msg) {
        gs.info(msg);
    })
}
})()
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| `<action>.warnMessages` | If `status` is false, any warning messages thrown during processing.  
  • Data type: Array of Strings |
| `<action>.errorMessages` | If `status` is false, any error messages thrown during processing.  
  • Data type: Array of Strings |

Example

```
(function () {
    var confUtils = new NotifyConferenceUtils();
    var participantGR = new GlideRecord('notify_participant');
    participantGR.get('validSysId');
    var actionResult = confUtils.muteByParticipantGR(participantGR);
    if (actionResult.status)
        gs.info('Participant has been muted');
    else {
        gs.info('mute operation failed');
        actionResult.warnMessages.forEach(function (msg) {
            gs.info(msg);
        });
        actionResult.errorMessages.forEach(function (msg) {
            gs.info(msg);
        });
    }
})();
```

NotifyConferenceUtils - NotifyConferenceUtils()

Instantiates a NotifyConferenceUtils object (constructor).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Example
function () {
  var confGR = new GlideRecord('notify_conference_call');
  confGR.get('76d3364d0b5133008e64aabc4673a6d');

  var confUtils = new NotifyConferenceUtils();
  var actionResult = confUtils.addToConferenceByPhoneNumber("+917799555331", confGR)
  if (actionResult.status)
    gs.info('Participant has been added to conference');
  else {
    gs.info('join operation failed');
    actionResult.warnMessages.forEach(function (msg) {
      gs.info(msg);
    });
    actionResult.errorMessages.forEach(function (msg) {
      gs.info(msg);
    });
  }
})();

NotifyConferenceUtils - unmuteByParticipantGR(GlideRecord notifyParticipantGR)

Unmutes the participant associated with the passed in GlideRecord on the current conference call.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyParticipantGR</td>
<td>GlideRecord</td>
<td>GlideRecord object of the participant to unmute. These records are located in the Notify Participant [notify_participant] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Results of the conference action.</td>
</tr>
<tr>
<td></td>
<td>&lt;action&gt;.status: Status of the conference action.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>• Valid values:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>◦ true: Conference action succeeded</td>
</tr>
<tr>
<td></td>
<td>◦ false: Conference action failed</td>
</tr>
<tr>
<td>&lt;action&gt;.successMessages:</td>
<td>If status is true, success message(s), else empty.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
<tr>
<td>&lt;action&gt;.warnMessages:</td>
<td>If status is false, any warning messages thrown during processing.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
<tr>
<td>&lt;action&gt;.errorMessages:</td>
<td>If status is false, any error messages thrown during processing.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
</tbody>
</table>

(function () {
  var confUtils = new NotifyConferenceUtils();
  var participantGR = new GlideRecord('notify_participant');
  participantGR.get('validSysId');
  var actionResult = confUtils.unmuteByParticipantGR(participantGR);
  if (actionResult.status)
    gs.info('Participant has been Unmuted');
  else {
    gs.info('Unmute operation failed');
    actionResult.warnMessages.forEach(function (msg) {
      gs.info(msg);
    });
    actionResult.errorMessages.forEach(function (msg) {
      gs.info(msg);
    });
  }
})();
NotifyJsTelephonyDriver - Global

The NotifyJsTelephonyDriver API provides methods that you can use to obtain information about the capabilities of the associated telephony driver.

It is a scripted extension point that can only be used when its object is returned by another method, such as NotifyUtil - getTelephonyDriverFromNotifyNumber(). You cannot call this API directly. The calling method associates a specific driver to the interface, abstracting the details of determining the driver. Each of the available drivers implement the same interface, which is defined by the NotifyJsTelephonyDriver API.

In the following code example, the getTelephonyDriverFromNotifyNumber() method returns an implementation of this API, with the driver interface being determined by the passed in telephone number, such as TwilioNotifyJsTelephonyDriver.

```javascript
function getDriverDetails(notifyNumber) {
  var nUtil = new NotifyUtil();
  if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
    return;
  var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
  if (driver) {
    return {
      driverName: driver.getName(),
      supportsCall: driver.supportsCall(),
      supportsSMS: driver.supportsSms()
    }
  }
}
```

To see which drivers are associated with the NotifyJsTelephonyDriver extension point, navigate to System Extension Points>global.NotifyJsTelephonyDriver. All associated driver implementations appear in the Implementations tab.

For additional information on scripted extension points, see Using extension points to extend application functionality point.

To use this API you must activate the Notify (com.snc.notify) plugin. To activate specific driver implementations, such as Twilio, you must activate their specific plugin.

**NotifyJsTelephonyDriver - call(Object notifyPhoneNumber, String phoneNumber)**

Calls a specified telephone number.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyPhoneNumber</td>
<td>NotifyPhoneNumber</td>
<td>Notify PhoneNumber record that contains the Notify phone number from which to make the call. Located in the Notify Phone Number [notify_number] table.</td>
</tr>
<tr>
<td>toPhoneNumber</td>
<td>String</td>
<td>Telephone number to call. Format: E.164</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
getDriverDetails('<notify_number>');

function getDriverDetails(notifyNumber) {
  var nUtil = new NotifyUtil();
  if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
    return;
  var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
  if (driver) {

  }
}
```

### NotifyJsTelephonyDriver - getCapabilities()

Returns a list of the capabilities of the telephony driver.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example returns the capabilities of the associated telephony driver.

```javascript
getDriverDetails('<notify_number>');
```

```javascript
function getDriverDetails(notifyNumber) {
    var nUtil = new NotifyUtil();
    if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
        return;
    var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
    if (driver) {
        return {
            capabilities: driver.getCapabilities()
        }
    }
}
```

### NotifyJsTelephonyDriver - getMaxSizeForBulkSms(String phoneNumber)

Returns the maximum number of telephone numbers to which the telephony driver associated with the passed in Notify telephone number can send an SMS message to at one time.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phoneNumber</td>
<td>String</td>
<td>Notify telephone number to check for the maximum number of telephone numbers that the associated driver is able to send to in one bulk SMS message.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Maximum number of telephone numbers that an SMS message can be sent to in a single bulk SMS delivery.</td>
</tr>
</tbody>
</table>

This example returns the maximum number of telephone numbers that can be sent to in a single bulk SMS message for the associated telephony driver.
function getDriverDetails(notifyNumber) {
    var nUtil = new NotifyUtil();
    if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
        return;
    var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
    if (driver) {
        return {
            maxSmsNumbers: driver.getMaxSizeForBulkSMS(notifyNumber)
        }
    }
}
NotifyJsTelephonyDriver - getPhoneNumbers()

Returns a list of all Notify telephone numbers associated with the current telephony driver.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Comma-separated list of all Notify telephone numbers associated with the current driver.</td>
</tr>
</tbody>
</table>

This example obtains the telephony driver and returns the list of Notify phone numbers associated with the current telephony driver.

```
getDriverDetails('<notify_number>');
```

```
function getDriverDetails(notifyNumber) {
    var nUtil = new NotifyUtil();
    if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
        return;
    var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
    if (driver) {
        return {
            phoneNumbers: driver.getPhoneNumbers()
        }
    }
}
```

NotifyJsTelephonyDriver - isActive()

Checks whether the current telephony driver is active.
This example obtains the telephony driver and returns whether the current telephony driver is active.

```javascript
getDriverDetails('<notify_number>');

function getDriverDetails(notifyNumber) {
  var nUtil = new NotifyUtil();
  if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
    return;
  var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
  if (driver) {
    return {
      active: driver.isActive()
    }
  }
}
```

**NotifyJsTelephonyDriver - kick(GlideRecord participantRecord)**

Removes the specified caller from the current Notify conference call.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>participantRecord</td>
<td>GlideRecord</td>
<td>GlideRecord object containing the Notify Participant [notify_participant] record of the caller to remove from the conference call.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Only returned if error. Error message that describes why the caller was not removed from the call.</td>
</tr>
</tbody>
</table>

This example mutes the associated caller in the current conference call.

```javascript
getDriverDetails('<notify_number>');
```

```javascript
function getDriverDetails(notifyNumber) {
    var nUtil = new NotifyUtil();
    if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
        return;
    var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
    if (driver) {
        var notifyParticipantGr = new GlideRecord('notify_participant');
        notifyParticipantGr.set('active participant sys id');

        if (notifyParticipantGr.isValid) {
            driver.kick(notifyParticipantGr)
        }
    }
}
```

**NotifyJsTelephonyDriver - mute(GlideRecord participantRecord)**

Mutes the specified caller in the current Notify conference call.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>participantRecord</td>
<td>GlideRecord</td>
<td>GlideRecord object containing the Notify Participant [notify_participant] record of the caller to mute in the conference call.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Only returned if error. Error message that describes why the caller was not muted.</td>
</tr>
</tbody>
</table>

This example mutes the associated caller in the current conference call.
getDriverDetails('<notify_number>');

function getDriverDetails(notifyNumber) {
    var nUtil = new NotifyUtil();
    if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
        return;
    var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
    if (driver) {
        var notifyParticipantGr = new GlideRecord('notify_participant');
        notifyParticipantGr.get('active participant sys id');

        if (notifyParticipantGr.isValid()) {
            driver.mute(notifyParticipantGr);
        }
    }
}

NotifyJsTelephonyDriver - sendAutonomousBulkSms(Object notifyPhoneNumber,
Array toPhoneNumber, String message, GlideRecord source)

Sends the specified Short Message Service (SMS) message to the specified list of telephone numbers.

In addition, you can optionally associate the Incident record that caused the SMS message to be generated with the SMS message.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Message to send.</td>
</tr>
<tr>
<td>notifyPhoneNumber</td>
<td>NotifyPhoneNumber - Global</td>
<td>Record that contains the phone number that is sending the bulk SMS message.</td>
</tr>
<tr>
<td>source</td>
<td>GlideRecord - Global</td>
<td>Optional. Incident GlideRecord to store in the Source field of the associated SMS message record in the Notify Message [notify message] table. This links the Incident record that caused the SMS message to be generated to that SMS message. Default: None. If this parameter is not passed in, this information is not tracked.</td>
</tr>
<tr>
<td>toPhoneNumber</td>
<td>Array</td>
<td>List of telephone numbers of the devices to receive the SMS message.</td>
</tr>
</tbody>
</table>
This example shows how to send an autonomous bulk SMS.

```
getDriverDetails('<notify_number>');
```

```
function getDriverDetails(notifyNumber) {
    var nUtil = new NotifyUtil();
    if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
        return;
    var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
    if (driver) {
    }
}
```

**NotifyJsTelephonyDriver - sendSMS(NotifyPhoneNumber notifyPhoneNumber, String toPhoneNumber, String messageBody)**

Sends a specified Short Message Service (SMS) message to a specified telephone number.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notifyPhoneNumber</td>
<td>NotifyPhoneNumber</td>
<td>Notify phone number record that contains the telephone number that is sending the SMS message. Located in the Notify Phone Number [notify_number] table.</td>
</tr>
<tr>
<td>toPhoneNumber</td>
<td>String</td>
<td>Phone number to send the SMS message to. Format: E.164 compliant</td>
</tr>
<tr>
<td>message</td>
<td>String</td>
<td>Text to send in the SMS message.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
This example sends an SMS message to a specified telephone number.

```javascript
getDriverDetails('<notify_number>');

function getDriverDetails(notifyNumber) {
    var nUtil = new NotifyUtil();
    if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
        return;
    var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
    if (driver) {
        return;
    }
}
```

**NotifyJsTelephonyDriver - supportsAutonomousBulkSms(String phoneNumber)**

Checks whether the specified Notify telephone number is capable of handling autonomous bulk Short Message Service (SMS) messages.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean| Flag that indicates whether the specified Notify telephone number supports autonomous bulk SMS. Valid values:  
• true: Supports autonomous bulk SMS.  
• false: Does not support autonomous bulk SMS. |

This example obtains the telephony driver and returns whether the driver supports autonomous bulk SMS.

```javascript
getDriverDetails('<notify_number>');

function getDriverDetails(notifyNumber) {
    var nUtil = new NotifyUtil();
    if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
        return;
    var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
    if (driver) {
        return;
    }
}
var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
if (driver) {
    return {
        supportsBulkSMS: driver.supportsAutonomousBulkSms(notifyNumber)
    }
}

NotifyJsTelephonyDriver - supportsCall()

Checks whether the current telephony driver is capable of handling telephone calls.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the current telephony driver supports telephone calls.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Driver supports telephone calls.</td>
</tr>
<tr>
<td></td>
<td>• false: Driver does not support telephone calls.</td>
</tr>
</tbody>
</table>

This example obtains the telephony driver and returns whether the driver supports telephone calls.

getDriverDetails('<notify_number>');

function getDriverDetails(notifyNumber) {
    var nUtil = new NotifyUtil();
    if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
        return;
    var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
    if (driver) {
        return {
            supportsCall: driver.supportsCall()
        }
    }
NotifyJsTelephonyDriver - supportsCallOverWebRtc()
Checks whether the specified Notify telephone number is capable of calls to a browser using WebRTC (Real-Time Communications.)

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the specified Notify telephone number supports browser calls using WebRTC.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Supports WebRTC.</td>
</tr>
<tr>
<td></td>
<td>• false: Does not support WebRTC.</td>
</tr>
</tbody>
</table>

This example obtains the telephony driver and returns whether the driver supports browser calls using WebRTC.

```javascript
getDriverDetails('<notify_number>');

function getDriverDetails(notifyNumber) {
  var nUtil = new NotifyUtil();
  if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
    return;
  var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
  if (driver) {
    return {
      supportsWebRTC: driver.supportsCallOverWebRtc(notifyNumber)
    }
  }
}
```

NotifyJsTelephonyDriver - supportsSMS()
Checks whether the current telephony driver is capable of handling Short Message Service (SMS) messages.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the current telephony driver supports SMS.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Driver supports SMS.</td>
</tr>
<tr>
<td></td>
<td>• false: Driver does not support SMS.</td>
</tr>
</tbody>
</table>

This example obtains the telephony driver and returns whether the driver supports SMS.

```javascript
getDriverDetails('<notify_number>');

function getDriverDetails(notifyNumber) {
    var nUtil = new NotifyUtil();
    if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
        return;
    var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
    if (driver) {
        return {
            supportsSMS: driver.supportsSMS()}
    }
}
```

**NotifyJsTelephonyDriver - unmute(GlideRecord participantRecord)**

Unmutes the specified caller in the current Notify conference call.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>participantRecord</td>
<td>GlideRecord</td>
<td>GlideRecord object containing the Notify Participant [notify_participant] record of the caller to mute in the conference call.</td>
</tr>
</tbody>
</table>
This example mutes the associated caller in the current conference call.

```
getDriverDetails('<notify_number>');
```

```
function getDriverDetails(notifyNumber) {
    var nUtil = new NotifyUtil();
    if (!notifyNumber || !nUtil.validateOutboundNotifyNumber(notifyNumber))
        return;
    var driver = nUtil.getTelephonyDriverFromNotifyNumber(notifyNumber);
    if (driver) {
        var notifyParticipantGr = new GlideRecord('notify_participant');
        notifyParticipantGr.get('active participant sys id');
        if (notifyParticipantGr.isValid) {
            driver.unmute(notifyParticipantGr)
        }
    }
}
```

**NotifyPhoneNumber - Global**

The *NotifyPhoneNumber* API allows you to query information about a Notify phone number.

Access the global *NotifyPhoneNumber* class and its associated methods in the *SNC* namespace.

**NotifyPhoneNumber - getDialCode()**

Returns the international dialing code for a Notify phone number.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>International phone code for a country.</td>
</tr>
</tbody>
</table>

This example shows how to obtain the dial code for a phone number.

```javascript
var numbers = SNC.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.getDialCode());
}
```

### Scoped equivalent

To use the `getDialCode()` method in a scoped application, use the corresponding scoped method: **Scoped NotifyPhoneNumber - getDialCode()**.

### NotifyPhoneNumber -getID()

Returns the ID of this phone number as defined by the telephony provider.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Identifier of the number within the telephony provider.</td>
</tr>
</tbody>
</table>

This example shows how to obtain the unique identifier for a phone number.

```javascript
var numbers = SNC.NotifyScoped.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);
```
// Here number is of type NotifyPhoneNumber
gs.info(number.getID());
}

**Scoped equivalent**

To use the `getID()` method in a scoped application, use the corresponding scoped method: **Scoped NotifyPhoneNumber - getID().**

**NotifyPhoneNumber - getNumber()**

Returns the numerical phone number for the current Notify caller.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>E.164-compliant phone number.</td>
</tr>
</tbody>
</table>

This example shows how to obtain a Notify caller's phone number.

```javascript
var numbers = SNC.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.getNumber());
}
```

**Scoped equivalent**

To use the `getNumber()` method in a scoped application, use the corresponding scoped method: **Scoped NotifyPhoneNumber - getNumber().**

**NotifyPhoneNumber - getOwner()**

Returns the telephony provider associated with this phone number.
This example shows how to obtain the telephony provider that owns the specified phone number.

```javascript
var numbers = SNC.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.getOwner());
}
```

**Scoped equivalent**

To use the `getOwner()` method in a scoped application, use the corresponding scoped method: `Scoped NotifyPhoneNumber - getOwner()`.

**NotifyPhoneNumber - getTerritory()**

Returns the country associated with the phone number.
This example shows how to obtain the country of a Notify caller.

```javascript
var numbers = SNC.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.getTerritory());
}
```

**Scoped equivalent**

To use the `getTerritory()` method in a scoped application, use the corresponding scoped method: **Scoped NotifyPhoneNumber - getTerritory()**.

**NotifyPhoneNumber - isShortCode()**

Determines whether the current Notify phone number is a short code.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Boolean</td>
</tr>
<tr>
<td>• true: phone number is a short code.</td>
</tr>
<tr>
<td>• false: phone number is not a short code.</td>
</tr>
</tbody>
</table>

This example shows how to check if the current Notify phone number is a short code.

```javascript
GlideRecord notifyNumber = new GlideRecord(TABLE_NOTIFY_NUMBER);
notifyNumber.query(COL_NUMBER, notifyPhoneNumber.getNumber());

if (!notifyNumber.next()) {
    notifyNumber.initialize();
    notifyNumber.setValue(COL_OWNER, notifyPhoneNumber.getOwner());
}
```
if (notifyPhoneNumber.isShortCode()) {
    notifyNumber.setValue(COL_SHORT_CODE, notifyPhoneNumber.getNumber());
} else {
    notifyNumber.setValue(COL_PHONE_NUMBER, notifyPhoneNumber.getNumber());
}

**NotifyPhoneNumber - supportsConferenceCall()**

Determines if the Notify phone number supports conference calling.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Value that indicates whether the Notify phone number supports conference calling.</td>
</tr>
<tr>
<td></td>
<td>• true: phone number does support conference calling</td>
</tr>
<tr>
<td></td>
<td>• false: phone number does not support conference calling</td>
</tr>
</tbody>
</table>

This example shows how to determine if a Notify caller supports conference calls.

```javascript
var numbers = SNC.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.supportsConferenceCall());
}
```

**Scoped equivalent**

To use the `supportsConferenceCall()` method in a scoped application, use the corresponding scoped method: `Scoped NotifyPhoneNumber - supportsConferenceCall()`.
**NotifyPhoneNumber - supportsIncomingPhoneCall()**

Determines if the Notify phone number supports receiving phone calls.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Value that indicates whether the Notify phone number supports incoming phone calls.</td>
</tr>
<tr>
<td></td>
<td>• true: phone number does support incoming phone calls</td>
</tr>
<tr>
<td></td>
<td>• false: phone number does not support incoming phone calls</td>
</tr>
</tbody>
</table>

This example shows how to determine if a Notify caller can receive incoming phone calls.

```javascript
var numbers = SNC.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.supportsIncomingPhoneCall());
}
```

**Scoped equivalent**

To use the `supportsIncomingPhoneCall()` method in a scoped application, use the corresponding scoped method: **Scoped NotifyPhoneNumber - supportsIncomingPhoneCall()**.

**NotifyPhoneNumber - supportsIncomingSMS()**

Determines if the Notify phone number supports receiving SMS messages.
This example shows how to determine if a Notify caller can receive incoming SMS messages.

```javascript
var numbers = SNC.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.supportsIncomingSMS());
}
```

**Scoped equivalent**

To use the `supportsIncomingSMS()` method in a scoped application, use the corresponding scoped method: `Scoped NotifyPhoneNumber - supportsIncomingSMS()`.

**NotifyPhoneNumber - supportsOutgoingPhoneCall()**

determines if the Notify phone number supports initiating phone calls.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean  | Value that indicates whether the Notify phone number supports initiating outgoing phone calls.  
- true: phone number does support initiating outgoing phone calls  
- false: phone number does not support initiating outgoing phone calls |

This example shows how to determine if a Notify caller can make outgoing phone calls.

```javascript
var numbers = SNC.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
gs.info(number.supportsOutgoingPhoneCall());
}
```

### Scoped equivalent

To use the `supportsOutgoingPhoneCall()` method in a scoped application, use the corresponding scoped method: `Scoped NotifyPhoneNumber - supportsOutgoingPhoneCall()`.

### NotifyPhoneNumber - supportsOutgoingSMS()

Determines if the Notify phone number supports sending SMS messages.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Value that indicates whether the Notify phone number supports sending SMS messages.</td>
</tr>
<tr>
<td></td>
<td>• true: phone number does support sending SMS messages</td>
</tr>
<tr>
<td></td>
<td>• false: phone number does not support sending SMS messages</td>
</tr>
</tbody>
</table>

This example shows how to determine if a Notify caller can initiate outgoing SMS messages.

```javascript
var numbers = SNC.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
  var number = numbers.get(0);

  // Here number is of type NotifyPhoneNumber
  gs.info(number.supportsOutgoingSMS());
}
```

Scoped equivalent

To use the `supportsOutgoingSMS()` method in a scoped application, use the corresponding scoped method: **Scoped NotifyPhoneNumber - supportsOutgoingSMS()**.

**NotifyPhoneNumber - supportsRecording()**

Determines if the Notify phone number supports recording phone calls.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Value that indicates whether the Notify phone number supports recording phone calls.</td>
</tr>
</tbody>
</table>
This example shows how to determine if a Notify caller can record calls.

```javascript
var numbers = SNC.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.supportsRecording());
}
```

**Scoped equivalent**

To use the `supportsRecording()` method in a scoped application, use the corresponding scoped method: `Scoped NotifyPhoneNumber - supportsRecording()`.

**NotifyPhoneNumber - supportsWebRTC()**

Determines if the Notify phone number supports calls to a browser, such as in a WebRTC implementation.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Boolean</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
This example shows how to determine if a Notify caller can initiate/accept browser-to-browser calls.

```javascript
var numbers = SNC.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.supportsWebRTC());
}
```

**Scoped equivalent**

To use the `supportsWebRTC()` method in a scoped application, use the corresponding scoped method: `Scoped NotifyPhoneNumber - supportsWebRTC()`.

**NotifyPhoneNumber - Scoped**

The `NotifyPhoneNumber` API allows you to query information about a Notify phone number.

Access the scoped `NotifyPhoneNumber` API and its associated methods in the `sn_notify` namespace.

**Scoped NotifyPhoneNumber - getDialCode()**

Returns the international dialing code for a Notify phone number.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

This example shows how to obtain the dial code for a phone number.

```javascript
var numbers = sn_notify.NotifyScoped.getPhoneNumbers();
```
// Here numbers is of type List
if (numbers.size() > 0) { 
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.getDialCode());
}

Scoped NotifyPhoneNumber - getID()
Returns the ID of this phone number as defined by the telephony provider.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Identifier of the number within the telephony provider.</td>
</tr>
</tbody>
</table>

This example shows how to obtain the unique identifier for a phone number.

var numbers = sn_notify.NotifyScoped.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) { 
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.getID());
}

Scoped NotifyPhoneNumber - getNumber()
Returns the numerical phone number for the current Notify caller.
This example shows how to obtain a Notify caller's phone number.

```javascript
var numbers = sn_notify.NotifyScoped.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.getNumber());
}
```

**Scoped NotifyPhoneNumber - getOwner()**

Returns the telephony provider associated with this phone number.

This example shows how to obtain the telephony provider that owns the specified phone number.

```javascript
var numbers = sn_notify.NotifyScoped.getPhoneNumbers();
```
// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.getOwner());
}

Scoped NotifyPhoneNumber - getTerritory()
Returns the country associated with the phone number.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the country to which the phone number belongs.</td>
</tr>
</tbody>
</table>

This example shows how to obtain the country of a Notify caller.

```java
var numbers = sn_notify.NotifyScoped.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.getTerritory());
}
```

Scoped NotifyPhoneNumber - supportsConferenceCall()
Determines if the Notify phone number supports conference calls.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Value that indicates whether the Notify phone number supports conference calling.</td>
</tr>
<tr>
<td></td>
<td>• true: phone number does support conference calling</td>
</tr>
<tr>
<td></td>
<td>• false: phone number does not support conference calling</td>
</tr>
</tbody>
</table>

This example shows how to determine if a Notify caller supports conference calls.

```javascript
var numbers = sn_notify.NotifyScoped.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.supportsConferenceCall());
}
```

Scoped NotifyPhoneNumber - supportsIncomingPhoneCall()

Determines if the Notify phone number supports receiving phone calls.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Value that indicates whether the Notify phone number supports incoming phone calls.</td>
</tr>
</tbody>
</table>

Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• true: phone number does support incoming phone calls</td>
</tr>
<tr>
<td></td>
<td>• false: phone number does not support incoming phone calls</td>
</tr>
</tbody>
</table>

This example shows how to determine if a Notify caller can receive incoming phone calls.

```java
var numbers = sn_notify.NotifyScoped.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.supportsIncomingPhoneCall());
}
```

Scoped NotifyPhoneNumber - supportsIncomingSMS()

Determines if the Notify phone number supports receiving SMS messages.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Value that indicates whether the Notify phone number supports incoming SMS messages.</td>
</tr>
<tr>
<td></td>
<td>• true: phone number does support incoming SMS messages</td>
</tr>
<tr>
<td></td>
<td>• false: phone number does not support incoming SMS messages</td>
</tr>
</tbody>
</table>

This example shows how to determine if a Notify caller can receive incoming SMS messages.

```java
var numbers = sn_notify.NotifyScoped.getPhoneNumbers();
```
// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.supportsIncomingSMS());
}

Scoped NotifyPhoneNumber - supportsOutgoingPhoneCall()
Determines if the Notify phone number supports initiating phone calls.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Value that indicates whether the Notify phone number supports initiating outgoing phone calls.</td>
</tr>
<tr>
<td></td>
<td>• true: phone number does support initiating outgoing phone calls</td>
</tr>
<tr>
<td></td>
<td>• false: phone number does not support initiating outgoing phone calls</td>
</tr>
</tbody>
</table>

This example shows how to determine if a Notify caller can make outgoing phone calls.

```javascript
var numbers = sn_notfy.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.supportsOutgoingPhoneCall());
}
```
Scoped NotifyPhoneNumber - supportsOutgoingSMS()

Determines if the Notify phone number supports sending SMS messages.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Value that indicates whether the Notify phone number supports sending SMS messages.</td>
</tr>
<tr>
<td></td>
<td>• true: phone number does support sending SMS messages</td>
</tr>
<tr>
<td></td>
<td>• false: phone number does not support sending SMS messages</td>
</tr>
</tbody>
</table>

This example shows how to determine if a Notify caller can initiate outgoing SMS messages.

```java
var numbers = sn_notify.NotifyScoped.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.supportsOutgoingSMS());
}
```

Scoped NotifyPhoneNumber - supportsRecording()

Determines if the Notify phone number supports recording phone calls.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Value that indicates whether the Notify phone number supports recording phone calls.</td>
</tr>
<tr>
<td></td>
<td>• true: phone number does support recording phone calls</td>
</tr>
<tr>
<td></td>
<td>• false: phone number does not support recording phone calls</td>
</tr>
</tbody>
</table>

This example shows how to determine if a Notify caller can record calls.

```javascript
var numbers = sn_notify.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
  var number = numbers.get(0);

  // Here number is of type NotifyPhoneNumber
  gs.info(number.supportsRecording());
}
```

### Scoped NotifyPhoneNumber - supportsWebRTC()

Determines if the Notify phone number supports calls to a browser, such as in a WebRTC implementation.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Value that indicates whether the Notify phone number supports browser-to-browser (WebRTC) calls.</td>
</tr>
<tr>
<td></td>
<td>• true: phone number does support browser-to-browser (WebRTC) calls</td>
</tr>
<tr>
<td></td>
<td>• false: phone number does not support browser-to-browser (WebRTC) calls</td>
</tr>
</tbody>
</table>
This example shows how to determine if a Notify caller can initiate/accept browser-to-browser calls.

```javascript
var numbers = sn_notify.Notify.getPhoneNumbers();

// Here numbers is of type List
if (numbers.size() > 0) {
    var number = numbers.get(0);

    // Here number is of type NotifyPhoneNumber
    gs.info(number.supportsWebRTC());
}
```

**NotifyOnTaskClient - Client**

The `NotifyOnTaskClient` API provides methods for sending SMS messages or starting/managing a conference call for various telephony service providers, such as Zoom and WebEx.

Any UI can consume the `NotifyOnTaskClient` API by explicitly including the `NotifyOnTaskClient UI script`.

Using the `NotifyOnTaskClient` API you can:

- Start a conference call
- End a conference call
- Add participants
- Perform actions that are available through the telephony driver such as:
  - mute/unmute participants
  - remove participants from a conference call
  - add participants to a conference call
  - start a conference call
  - end a conference call

The Notify (com.snc.notify) plugin requires a separate subscription. For additional information on activating the Notify plugin, see [Activate Notify](#).

**NotifyOnTaskClient - addParticipants(Object data)**

Adds the specified participants to a specified conference call.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Object</td>
<td>Object that describes the conference call.</td>
</tr>
<tr>
<td>data.addToWorkNotes</td>
<td>Boolean</td>
<td>Flag that indicates whether to add information about the participants that were included in the conference call in the work notes field of the associated record. For this functionality to work, you must also specify values in the <code>data.table</code> and <code>data.sysId</code> parameters. These parameters identify the record in which to add the work notes. Default: false</td>
</tr>
<tr>
<td>data.confId</td>
<td>String</td>
<td>Required. Sys ID of the conference call. The conference Sys ID is located in the Notify Conference Call [notify_conference_call] table.</td>
</tr>
<tr>
<td>data.items</td>
<td>Array</td>
<td>Required. Information for each participant to include in the conference call. Valid array values: • <strong>id</strong>: User Sys ID; located in the User [sys_User] table. • <strong>notifyParticipantId</strong>: Participant Sys ID; located in the Notify Participant [notify_participant] table. • <strong>phoneNumber</strong>: Phone number of the participant. If this value is passed in conjunction with either the <code>id</code> or <code>notifyParticipantId</code>, this value supersedes the phone numbers in the user/participant record and is used to place the call. • <strong>email</strong>: email address of the participant.</td>
</tr>
<tr>
<td>data.message</td>
<td>String</td>
<td>Message that is read aloud when a user answers the call, such as, &quot;P1 incident has been created please login to instance.&quot;</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data.serviceProvider</td>
<td>String</td>
<td>Required. Name of conference service provider, such as Zoom or Webex.</td>
</tr>
<tr>
<td>data.sysId</td>
<td>String</td>
<td>Sys ID of the source record to associate with the conference call.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, if a conference call is held to discuss a specific incident or problem, put the Sys ID of the incident or problem record in this value. This Sys ID is stored in the Source column of the NotifyConference Call [notify_conference_call] table and can later be tracked.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This parameter is used in conjunction with the data.Table, data.addToWorkNotes, and allowMulticonference parameters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You should configure this value when the conference call is initially created through a &quot;start&quot; action. If required, you can also set this value through this method.</td>
</tr>
<tr>
<td>data.table</td>
<td>String</td>
<td>Table that contains the source record to associate with the conference call. A source record can be any record, such as an &quot;incident&quot; or &quot;problem&quot;, that is the topic of discussion in the conference call.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This table name is stored in the Table column of the NotifyConference Call [notify_conference_call] table and can be tracked.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This parameter is used in conjunction with the data.sysId, data.addToWorkNotes, and allowMulticonference parameters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You should configure this value when the conference call is initially created through a &quot;start&quot; action. If required, you can also set this value through this method.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Results of the conference action.</td>
</tr>
</tbody>
</table>

- `<action>.status`: Status of the conference action.
  - Data type: Boolean
  - Valid values:
    - true: Conference action succeeded
    - false: Conference action failed

- `<action>.successMessages`: If `status` is true, success message(s), else empty.
  - Data type: Array of Strings

- `<action>.warnMessages`: If `status` is false, any warning messages thrown during processing.
  - Data type: Array of Strings

- `<action>.errorMessages`: If `status` is false, any error messages thrown during processing.
  - Data type: Array of Strings

### Example

```javascript
function addToConferenceCall() {
    var data = NotifyOnTaskClient.getNotifyActionTemplate();
    data.serviceProvider = 'Telephony'; // e.g 'Zoom', 'WebEx'
    data.confId = 'Active conference sysId';
    data.items.push({ id: 'userSysId' });
    data.items.push({ phoneNumber: '+917799555331' });
    data.items.push({ email: 'yln99518@gmail.com' });

    NotifyOnTaskClient.addParticipants(data).then(function (result) {
        var joinActionResult = result[0];
        if(joinActionResult.status) {
            joinActionResult.successMessages.forEach(function(msg) {
                console.log(msg);
            });
        }
    });
}
return;
}

joinActionResult.warnMessages.forEach(function(msg) {
    console.warn(msg);
});
joinActionResult.errorMessages.forEach(function(msg) {
    console.error(msg);
});
}, function (errMsg) {
    console.log(errMsg);
});

NotifyOnTaskClient - doConferenceAction(String action, Object data)

Performs the specified conference call action, such as starting/ending a conference call or joining, removing, muting, or unmuting participants from a conference call.

You can start a new conference call and add participants within a single call to this method or call the method multiple times to start the call and then manage participants separately. In addition, through the passed in data object, you can configure the method to:

- Save pointers in the conference call record to the specific record (source record), such as an incident or problem, that is the topic of discussion for the conference call.
- Allow/disallow multiple conference calls for a source record.
- Automatically log the participants that were in the conference call in the **Work notes** field of the source record.
- Have a message read aloud when a participant answers an outgoing call from the conference.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>String</td>
<td>Defines the conference call action to perform. The following are the available conference call actions:</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>start</td>
<td></td>
<td>• start: Starts the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>end</td>
<td></td>
<td>• end: Terminates the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>join</td>
<td></td>
<td>• join: Adds the participant specified in the <code>data.items</code> array to the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>multiJoin</td>
<td></td>
<td>• multiJoin: Adds the participants specified in the <code>data.items</code> array to the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>selfJoin</td>
<td></td>
<td>• selfJoin: Adds the currently logged in user to the conference call (no entry in <code>data.items</code> required.)</td>
</tr>
<tr>
<td>kick</td>
<td></td>
<td>• kick: Removes the participant specified in the <code>data.items</code> array from the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>multiKick</td>
<td></td>
<td>• multiKick: Removes the participants specified in the <code>data.items</code> array from the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>mute</td>
<td></td>
<td>• mute: Mutes the participant specified in the <code>data.items</code> array on the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>multiMute</td>
<td></td>
<td>• multiMute: Mutes the participants specified in the <code>data.items</code> array on the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>unmute</td>
<td></td>
<td>• unmute: Unmutes the participant specified in the <code>data.items</code> array from the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>multiUnmute: multiUnmute</td>
<td>multiUnmute</td>
<td>Unmutes the participants specified in the <code>data.items</code> array from the conference call identified in <code>data.confId</code></td>
</tr>
<tr>
<td>data</td>
<td>Object</td>
<td>Object that describes the conference call.</td>
</tr>
<tr>
<td>data.addToWorkNotes</td>
<td>Boolean</td>
<td>Flag that indicates whether to add information about the participants that were included in the conference call in the work notes field of the associated record.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For this functionality to work, you must also specify values in the <code>data.table</code> and <code>data.sysId</code> parameters. These parameters identify the record in which to add the work notes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Actions for which this parameter is valid:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• start  Adam</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• join  Adam</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• multiJoin  Adam  Miranda</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• selfJoin  Adam  Lauren</td>
</tr>
<tr>
<td>data.allowMulticonference</td>
<td>Boolean</td>
<td>Flag that indicates whether to allow multiple conference calls for a specific record at one time.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For this functionality to work, you must also specify values in the <code>data.table</code> and <code>data.sysId</code> parameters. These parameters identify the record that is allowed to have multiple conference calls.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Actions for which this parameter is valid:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• start  Adam</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• join  Adam</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• multiJoin  Adam  Miranda</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• selfJoin  Adam  Lauren</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| data.confId         | String | Sys ID of the conference call. The conference Sys ID is located in the Notify Conference Call [notify_conference_call] table. Actions for which this parameter is required:  
  - end  
  - join  
  - multiJoin  
  - selfJoin  
  
  **Note:** Participant actions such as mute, unmute, and kick do not require this parameter to be set as the method obtains this information from the Notify Conference Call Participant [notify_participant] table. |
| data.fromNumber     | String | Service provider number to call into for the conference call.  
  Locate this value in the Number or Phone number column of the Notify Phone Number [notify_number] table.  
  Actions for which this parameter is required:  
  - start |
| data.isNewConference| Boolean| Flag that indicates whether this is a new or an existing conference call.  
  Valid values:  
  - true: New conference call  
  - false: Existing conference call |
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Default</strong>: false</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Actions for which this parameter is valid:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>start</strong></td>
</tr>
<tr>
<td>data.message</td>
<td>String</td>
<td>Message that is read aloud when a user answers the call, such as, &quot;P1 incident has been created please login to instance.&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Actions for which this parameter is valid:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>start</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>join</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>multiJoin</strong></td>
</tr>
<tr>
<td>data.items</td>
<td>Array</td>
<td>Information for each participant to include in the conference call.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Valid array values:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>id</strong>: Sys ID of user; located in the User [sys_User] table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid actions: join, multiJoin, start</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>notifyParticipantId</strong>: Sys ID of the Notify participant; located in the Notify Participant [notify_participant] table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid actions: join, kick, multiJoin, mute, start, unmute</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>phoneNumber</strong>: Phone number of the participant. If this value is passed in conjunction with either the <strong>id</strong> or <strong>notifyParticipantId</strong>, this value supersedes the phone numbers in the user/participant record and is used to place the call.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid actions: join, multiJoin, start</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>email</td>
<td></td>
<td>• Email address of the participant. Valid actions: join, multiJoin, start.</td>
</tr>
<tr>
<td>data.serviceProvider</td>
<td>String</td>
<td>Required. Name of conference service provider, such as Zoom or Webex. Actions for which this parameter is required: • all.</td>
</tr>
<tr>
<td>data.sysId</td>
<td>String</td>
<td>Sys ID of the source record to associate with the conference call. For example, if a conference call is held to discuss a specific incident or problem, put the Sys ID of the incident or problem record in this value. This Sys ID is stored in the Source column of the NotifyConference Call [notify_conference_call] table and can later be tracked. This parameter is used in conjunction with the data.Table, data.addToWorkNotes, and allowMulticonference parameters. Actions for which this parameter is valid: • start.</td>
</tr>
<tr>
<td>data.table</td>
<td>String</td>
<td>Table that contains the source record to associate with the conference call. A source record can be any record, such as an &quot;incident&quot; or &quot;problem&quot;, that is the topic of discussion in the conference call. This table name is stored in the Table column of the NotifyConference Call [notify_conference_call] table and can be tracked.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>This parameter is used in conjunction with the <code>data.sysId</code>, <code>data.addtoworkNotes</code>, and <code>allowMulticonference</code> parameters. Actions for which this parameter is valid:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• start</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Results of the conference action.</td>
</tr>
<tr>
<td></td>
<td><code>&lt;action&gt;.status</code>: Status of the conference action.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>• Valid values:</td>
</tr>
<tr>
<td></td>
<td>◦ true: Conference action succeeded</td>
</tr>
<tr>
<td></td>
<td>◦ false: Conference action failed</td>
</tr>
<tr>
<td></td>
<td><code>&lt;action&gt;.successMessages</code>: If <code>status</code> is true, success message(s), else empty.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
<tr>
<td></td>
<td><code>&lt;action&gt;.warnMessages</code>: If <code>status</code> is false, any warning messages thrown during processing.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
<tr>
<td></td>
<td><code>&lt;action&gt;.errorMessages</code>: If <code>status</code> is false, any error messages thrown during processing.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
</tbody>
</table>

The following example shows how to create a function to call `doConferenceAction()` to manipulate the participants in a conference call by passing in the action and the participants.
/**
 * @param {string} action - action to perform on the conference object or participant object
 * @param {Array} participants;
 */
function doConferenceAction(action, participants) {
  var data = NotifyOnTaskClient.getNotifyActionTemplate();
  data.serviceProvider = 'Telephony'; // e.g 'Zoom', 'WebEx'
  data.confId = 'Active conference sysId';
  data.items = participants;

  NotifyOnTaskClient.doConferenceAction(action, data).then(function (result) {
    var kickActionResult = result[0];
    if (kickActionResult.status)
      console.log(action + ' succeeded');
    else {
      kickActionResult.warnMessages.forEach(function (msg) {
        console.warn(msg);
      });
      kickActionResult.errorMessages.forEach(function (msg) {
        console.error(msg);
      });
    }
  }, function (errMsg) {
    console.log(errMsg)
  });
}

// kick participants
doConferenceAction('kick', [{notifyParticipantId: 'notifyParticipantSysId'}]);

// kick multiple participants
doConferenceAction('multiKick', [{notifyParticipantId: 'notifyParticipantSysId'},
  {notifyParticipantId: 'notifyParticipantSysId'}]);

// Mute participants
doConferenceAction('mute', [{notifyParticipantId: 'notifyParticipantSysId'}]);
doConferenceAction('mute', [{notifyParticipantId: 'notifyParticipantSysId'}]);
doConferenceAction('multiMute',

// self join to any conference.
doConferenceAction('selfJoin', [{id: 'logged in userId'}]);

NotifyOnTaskClient - endConference(Object data)
Terminates the specified conference call.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Object</td>
<td>Object that describes the conference call.</td>
</tr>
<tr>
<td>data.confId</td>
<td>String</td>
<td>Sys ID of the conference call.</td>
</tr>
<tr>
<td>data.serviceProvider</td>
<td>String</td>
<td>Required. Name of conference service provider, such as Zoom or Webex.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Results of the conference action.</td>
</tr>
<tr>
<td>&lt;action&gt;.status</td>
<td>Status of the conference action.</td>
</tr>
<tr>
<td>• Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>• Valid values:</td>
<td></td>
</tr>
<tr>
<td>◦ true: Conference action succeeded</td>
<td></td>
</tr>
<tr>
<td>◦ false: Conference action failed</td>
<td></td>
</tr>
<tr>
<td>&lt;action&gt;.successMessages</td>
<td>If status is true, success message(s), else empty.</td>
</tr>
<tr>
<td>• Data type: Array of Strings</td>
<td></td>
</tr>
<tr>
<td>&lt;action&gt;.warnMessages</td>
<td>If status is false, any warning messages thrown during processing.</td>
</tr>
<tr>
<td>• Data type: Array of Strings</td>
<td></td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;action&gt;.errorMessages</code></td>
<td>If status is false, any error messages thrown during processing.</td>
</tr>
<tr>
<td>• Data type: Array of Strings</td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
function endConferenceCall() {
  var data = NotifyOnTaskClient.getNotifyActionTemplate();
  data.serviceProvider = 'Telephony'; // e.g 'Zoom', 'WebEx'
  data.confId = 'Active conference sysId';

  NotifyOnTaskClient.endConference(data).then(function (result) {
    var endActionResult = result[0];
    if (endActionResult.status)
      console.log('Conference has been ended');
    else {
      endActionResult.warnMessages.forEach(function (msg) {
        console.warn(msg);
      });
      endActionResult.errorMessages.forEach(function (msg) {
        console.error(msg);
      });
    }
  }, function (errMsg) {
    console.log(errMsg);
  });
}
```

**NotifyOnTaskClient - getNotifyActionTemplate()**

Returns a JSON data template to use with the `doConferenceAction()` method. Using this template automatically structures the data object so that you don't have to manually create it.

Call this method prior to calling the `doConferenceAction()` method. For the desired conference call action, set the desired parameters within the template, and then pass the template in the `doConferenceAction()` call. For additional information on the valid parameters for each action, see `doConferenceAction()`. 
Note: This is a helper method. You can also manually construct this object and pass it into the `doConferenceAction()` method and have the same outcome.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Object that describes the conference call.</td>
</tr>
<tr>
<td>data.addToWorkNotes</td>
<td>Flag that indicates whether to add information about the participants that were included in the conference call in the work notes field of the associated record. For this functionality to work, you must also specify values in the <code>data.table</code> and <code>data.sysId</code> parameters. These parameters identify the record in which to add the work notes. Default: false. Actions for which this parameter is valid: • start • join • multiJoin • selfJoin</td>
</tr>
<tr>
<td>data.allowMultconference</td>
<td>Flag that indicates whether to allow multiple conference calls for a specific record at one time. For this functionality to work, you must also specify values in the <code>data.table</code> and <code>data.sysId</code> parameters. These parameters identify the record that is allowed to have multiple conference calls. Default: false. Actions for which this parameter is valid: • start</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>data.confId</td>
<td>Sys ID of the conference call. &lt;br&gt;The conference Sys ID is located in the Notify Conference Call [notify_conference_call] table. &lt;br&gt;Actions for which this parameter is required:  &lt;br&gt;• end  &lt;br&gt;• join  &lt;br&gt;• multiJoin  &lt;br&gt;• selfJoin  &lt;br&gt;<strong>Note:</strong> Participant actions such as mute, unmute, and kick do not require this parameter to be set as the method obtains this information from the Notify Conference Call Participant [notify_participant] table.</td>
</tr>
<tr>
<td>data.fromNumber</td>
<td>Service provider number to call into for the conference call. &lt;br&gt;Locate this value in the Number or Phone number column of the Notify Phone Number [notify_number] table. &lt;br&gt;Actions for which this parameter is required:  &lt;br&gt;• start</td>
</tr>
<tr>
<td>data.isNewConference</td>
<td>Flag that indicates whether this is a new or an existing conference call. &lt;br&gt;Valid values:  &lt;br&gt;• true: New conference call  &lt;br&gt;• false: Existing conference call  &lt;br&gt;Default: false &lt;br&gt;Actions for which this parameter is valid:  &lt;br&gt;• start</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data.items</td>
<td>Information for each participant to include in the conference call. Valid array values:</td>
</tr>
<tr>
<td></td>
<td>• id: Sys ID of user; located in the User [sys_User] table. Valid actions: join, multiJoin, start</td>
</tr>
<tr>
<td></td>
<td>• notifyParticipantId: Sys ID of the Notify participant; located in the Notify Participant [notify_participant] table. Valid actions: join, kick, multiJoin, mute, start, unmute</td>
</tr>
<tr>
<td></td>
<td>• phoneNumber: Phone number of the participant. If this value is passed in conjunction with either the id or notifyParticipantId, this value supersedes the phone numbers in the user/participant record and is used to place the call. Valid actions: join, multiJoin, start</td>
</tr>
<tr>
<td></td>
<td>• email: Email address of the participant. Valid actions: join, multiJoin, start</td>
</tr>
<tr>
<td>data.message</td>
<td>Message that is read aloud when a user answers the call, such as, &quot;P1 incident has been created please login to instance.&quot;</td>
</tr>
<tr>
<td></td>
<td>Actions for which this parameter is valid: start, join, multiJoin</td>
</tr>
<tr>
<td>data.serviceProvider</td>
<td>Required. Name of conference service provider, such as Zoom or Webex. Actions for which this parameter is required:</td>
</tr>
<tr>
<td></td>
<td>• all</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data.sysId</td>
<td>Sys ID of the source record to associate with the conference call. For example, if a conference call is held to discuss a specific incident or problem, put the Sys ID of the incident or problem record in this value. This Sys ID is stored in the Source column of the NotifyConference Call [notify_conference_call] table and can later be tracked. This parameter is used in conjunction with the data.Table, data.addToWorkNotes, and allowMulticonference parameters. Actions for which this parameter is valid: • start</td>
</tr>
<tr>
<td>data.table</td>
<td>Table that contains the source record to associate with the conference call. A source record can be any record, such as an &quot;incident&quot; or &quot;problem&quot;, that is the topic of discussion in the conference call. This table name is stored in the Table column of the NotifyConference Call [notify_conference_call] table and can be tracked. This parameter is used in conjunction with the data.sysId, data.addToWorkNotes, and allowMulticonference parameters. Actions for which this parameter is valid: • start</td>
</tr>
</tbody>
</table>

The following example shows how to call getNotifyActionTemplate() to obtain the data template necessary to define the actions for doConferenceAction().

```javascript
/**
 * @param {string} action - action to perform on the conference object or participant object
```
* @param {Array} participants;
*/

function doConferenceAction(action, participants) {
    var data = NotifyOnTaskClient.getNotifyActionTemplate();
    data.serviceProvider = 'Telephony'; // e.g 'Zoom', 'WebEx'
    data.confId = 'Active conference sysId';
    data.items = participants;

    NotifyOnTaskClient.doConferenceAction(action, data).then(function (result) {
        var kickActionResult = result[0];
        if (kickActionResult.status)
            console.log(action + ' succeeded');
        else {
            kickActionResult.warnMessages.forEach(function (msg) {
                console.warn(msg);
            });
            kickActionResult.errorMessages.forEach(function (msg) {
                console.error(msg);
            });
        }
    }, function (errMsg) {
        console.log(errMsg)
    });
}

// kick participants
doConferenceAction('kick', [{notifyParticipantId: 'notifyParticipantSysId'}]);

// kick multiple participants
doConferenceAction('multiKick', [
    {notifyParticipantId: 'notifyParticipantSysId'},
    {notifyParticipantId: 'notifyParticipantSysId'}]);

// Mute participants
doConferenceAction('mute', [{notifyParticipantId: 'notifyParticipantSysId'}]);
doConferenceAction('mute', [{notifyParticipantId: 'notifyParticipantSysId'}]);
doConferenceAction('multiMute', [
    {notifyParticipantId: 'notifyParticipantSysId'},
    {notifyParticipantId: 'notifyParticipantSysId'}]);

// self join to any conference.
doConferenceAction('selfJoin', [{id: 'logged in userId'}]);

NotifyOnTaskClient - start(Object data)

Starts a new conference call.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Object</td>
<td>Object that describes the conference call.</td>
</tr>
<tr>
<td>data.addToWorkNotes</td>
<td>Boolean</td>
<td>Optional. Flag that indicates whether to add information about the participants that were included in the conference call in the work notes field of the associated record. For this functionality to work, you must also specify values for the <code>data.table</code> and <code>data.sysId</code> parameters to identify the record in which to add the work notes. Default: false</td>
</tr>
<tr>
<td>data.allowMulticonference</td>
<td>Boolean</td>
<td>Optional. Flag that indicates whether to allow multiple conference calls for a specific record at one time. For this functionality to work, you must also specify values in the <code>data.table</code> and <code>data.sysId</code> parameters. These parameters identify the record that is allowed to have multiple conference calls. Default: false</td>
</tr>
<tr>
<td>data.fromNumber</td>
<td>String</td>
<td>Required. Service provider number to call into for the conference call. Locate this value in the Number or Phone number column of the Notify Phone Number [notify_number] table.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| data.items       | Array   | Optional. Information for each participant to include in the conference call. Valid array values:  
  • **id**: User Sys ID; located in the User [sys_User] table.  
  • **notifyParticipantId**: Participant Sys ID; located in the Notify Participant [notify_participant] table.  
  • **phoneNumber**: Phone number of the participant. If this value is passed in conjunction with either the **id** or **notifyParticipantId**, this value supersedes the phone numbers in the user/participant record and is used to place the call.  
  • **email**: Email address of the participant. |
| data.message     | String  | Optional. Message that is read aloud when a user answers the call, such as, "P1 incident has been created please login to instance." |
| data.serviceProvider | String | Required. Name of conference service provider, such as Zoom or Webex. |
| data.sysId       | String  | Optional. Sys ID of the source record to associate with the conference call. For example, if a conference call is held to discuss a specific incident or problem, put the Sys ID of the incident or problem record in this value. This Sys ID is stored in the Source column of the NotifyConference Call [notify_conference_call] table and can later be tracked. |
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data.table</td>
<td>String</td>
<td>Optional. Table that contains the source record to associate with the conference call. A source record can be any record, such as an &quot;incident&quot; or &quot;problem&quot;, that is the topic of discussion in the conference call. This table name is stored in the Table column of the NotifyConference Call [notify_conference_call] table and can be tracked. This parameter is used in conjunction with the <code>data.sysId</code>, <code>data.addToWorkNotes</code>, and <code>allowMulticonference</code> parameters.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Results of the conference action.</td>
</tr>
<tr>
<td></td>
<td><code>&lt;action&gt;.status</code>: Status of the conference action.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>• Valid values:</td>
</tr>
<tr>
<td></td>
<td>◦ true: Conference action succeeded</td>
</tr>
<tr>
<td></td>
<td>◦ false: Conference action failed</td>
</tr>
<tr>
<td></td>
<td><code>&lt;action&gt;.successMessages</code>: If <code>status</code> is true, success message(s), else empty.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
<tr>
<td></td>
<td><code>&lt;action&gt;.warnMessages</code>: If <code>status</code> is false, any warning messages thrown during processing.</td>
</tr>
<tr>
<td></td>
<td>• Data type: Array of Strings</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;action&gt;.errorMessages:</td>
<td>If status is false, any error messages thrown during processing.</td>
</tr>
<tr>
<td>• Data type: Array of Strings</td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
function startConferenceCall() {
    var data = NotifyOnTaskClient.getNotifyActionTemplate();
    data.table = 'incident';
    data.sysId = '1234';
    data.serviceProvider = serviceProvider;
    data.addToWorkNotes = true;
    data.fromNumber = 'Telephony Number';
    data.items.push({ id: 'userSysId' });
    data.items.push({ phoneNumber: '+917799555332' });
    data.items.push({ email: 'yln99517@gmail.com' });

    NotifyOnTaskClient.start(data).then(function (result) {
        var startActionResult = result[0];
        if(startActionResult.status) {
            startActionResult.successMessages.forEach(function(msg) {
                console.log(msg);
            });
            return;
        }
        startActionResult.warnMessages.forEach(function(msg) {
            console.warn(msg);
        });
        startActionResult.errorMessages.forEach(function(msg) {
            console.error(msg);
        });
        }, function (errMsg) {
            console.log(errMsg);
        });
    }
```
**NotifySMS - Global**

The NotifySMS API provides methods that enable you to send Short Message Service (SMS) messages from Notify telephone numbers to devices using either their telephone numbers or a GlideRecord that has a field that contains telephone number information.

To use this API you must activate the Notify (com.snc.notify) plugin.

**NotifySMS - NotifySMS()**

Creates an instance of a NotifySMS object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

This example shows how to instantiate a NotifySMS object.

```javascript
var sourceRecord = new GlideRecord('incident');
sourceRecord.query();
if (sourceRecord.next()) {
  var fromNumber = getFromNumber();
  var groupId = sourceRecord.assignment_group + ' ';
  var toGr = getRecipientGRs(groupId);
  var message = 'An Incident ' + sourceRecord.getDisplayValue() + ' has been assigned to your group';
  if (fromNumber && toGr) {
    var notifySMS = new NotifySMS();
    notifySMS.sendToUsers(fromNumber, toGr, message, sourceRecord);
  }
}
```

**NotifySMS - sendToNumber(String fromNumber, String toNumber, String message, Object sourceRecord)**

Sends a Short Message Service (SMS) message from a specified Notify telephone number to a specified telephone number.

In addition, you can optionally associate the Incident record that caused the SMS message to be generated to the SMS message.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromNumber</td>
<td>String</td>
<td>Notify telephone number sending the SMS message.</td>
</tr>
<tr>
<td>message</td>
<td>String</td>
<td>Message to send.</td>
</tr>
<tr>
<td>sourceRecord</td>
<td>GlideRecord  - Global</td>
<td>Optional. Incident GlideRecord to store in the Source field of the associated SMS message record in the Notify Message [notify_message] table. This links the Incident record that caused the SMS message to be generated to that SMS message. Default: None. If this parameter is not passed in, this information is not tracked.</td>
</tr>
<tr>
<td>toNumber</td>
<td>String</td>
<td>Telephone number of the device to receive the SMS message.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example illustrates how to send an SMS message to a specified phone number and stamp the associated Incident record with the message information.

```javascript
var fromNumber = getFromNumber();
var toNumber = '+123456789';
var message = 'This is an example SMS';
var sourceRecord = new GlideRecord('incident');
sourceRecord.query();
if (sourceRecord.next()) {
    var notifySMS = new NotifySMS();
    notifySMS.sendToNumber(fromNumber, toNumber, message, sourceRecord);
}

function getFromNumber() {
    var prop = gs.getProperty('custom_property_name', '');
    if (!prop) {
        return getFallbackFromNumber();
    }
    return prop;
}
```
NotifySMS - sendToNumbers(String fromNumber, Array toNumber, String message, Object sourceRecord)

Sends a Short Message Service (SMS) message from a specified Notify phone number to a list of phone numbers.

In addition, you can optionally associate the Incident record that caused the SMS message to be generated with the SMS message.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromNumber</td>
<td>String</td>
<td>Notify telephone number sending the SMS message.</td>
</tr>
<tr>
<td>message</td>
<td>String</td>
<td>Message to send.</td>
</tr>
<tr>
<td>sourceRecord</td>
<td>GlideRecord - Global</td>
<td>Optional. Incident GlideRecord to store in the Source field of the associated SMS message record in the Notify Message [notify_message] table. This links the Incident record that caused the SMS message to be generated to that SMS message. Default: None. If this parameter is not passed in, this information is not tracked.</td>
</tr>
<tr>
<td>toNumber</td>
<td>Array</td>
<td>List of telephone numbers to receive the SMS message.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
The following example illustrates how to send an SMS message to multiple telephone numbers and store the information in the associated Incident record.

```javascript
var fromNumber = getFromNumber();
var toNumbers = getRecipientNumbers();
var message = 'This is an example SMS';
var sourceRecord = new GlideRecord('incident');
sourceRecord.query();
if (sourceRecord.next()) {
    var notifySMS = new NotifySMS();
    notifySMS.sendToNumbers(fromNumber, toNumbers, message, sourceRecord);
}

function getRecipientNumbers() {
    var userGr = new GlideRecord('sys_user');
    userGr.addActiveQuery();
    userGr.addQuery('first_name', 'STARTSWITH', 'A');
    userGr.setLimit(5);
    userGr.query();
    var userIds = [];
    while (userGr.next()) {
        userIds.push(userGr.getUniqueValue());
    }
    if (userIds.length > 0) {
        var nUtil = new NotifyUtil();
        return nUtil.getUniquePhoneNumbersForUsersAndGroups(null, userIds, null, 'sms', false);
    }
}

function getFromNumber() {
    var prop = gs.getProperty('custom_property_name', '');
    if (!prop) {
        return getFallbackFromNumber();
    }
    return prop;
}

function getFallbackFromNumber() {
    var notifyNumGr = new GlideRecord('notify_number');
    notifyNumGr.addActiveQuery();
    notifyNumGr.addQuery('has_sms_out', 'yes');
    notifyNumGr.query();
    if (notifyNumGr.next()) {
```
NotifySMS - sendToUser(String fromNumber, Object toGr, String message, Object sourceRecord)

Sends a Short Message Service (SMS) message from a specified phone number to the user identified in a specified GlideRecord.

This method extracts the telephone number from the specified GlideRecord. In addition, you can optionally associate the Incident record that caused the SMS message to be generated with the SMS message.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromNumber</td>
<td>String</td>
<td>Notify telephone number sending the SMS message.</td>
</tr>
<tr>
<td>message</td>
<td>String</td>
<td>Message to send.</td>
</tr>
<tr>
<td>sourceRecord</td>
<td>GlideRecord</td>
<td>Optional. Incident GlideRecord to store in the Source field of the associated SMS message record in the Notify Message [notify_message] table. This links the Incident record that caused the SMS message to be generated to that SMS message. Default: None. If this parameter is not passed in, this information is not tracked.</td>
</tr>
</tbody>
</table>
| toGr            | GlideRecord      | GlideRecord of a record type that contains a field that resolves to a user's telephone number. The record type must correlate with the record type used by the phone number resolver that is currently implemented. A phone number resolver is simply a method that obtains the user telephone number from a specific type of GlideRecord, such as a User record or a Case record. When using the default resolver, the records correspond to the User [sys_user] table. To change the type of record that the resolver uses to obtain the telephone number, use the NotifySMS - setRecordToNumberResolver(Object

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The following example illustrates how to send an SMS message to a user identified in the associated Incident record and stamp the associated Incident record with the message information.

```javascript
var sourceRecord = new GlideRecord('incident');
sourceRecord.query();
if (sourceRecord.next()) {
    var fromNumber = getFromNumber();
    var toGr = sourceRecord.assigned_to.getRefRecord();
    var message = 'An Incident ' + sourceRecord.getDisplayValue() + ' has been assigned to you';
    if (fromNumber && toGr.isValidRecord()) {
        var notifySMS = new NotifySMS();
        notifySMS.sendToUser(fromNumber, toGr, message, sourceRecord);
    }
}

function getFromNumber() {
    var prop = gs.getProperty('custom_property_name', '');
    if (!prop) {
        return getFallbackFromNumber();
    }
    return prop;
}

function getFallbackFromNumber() {
    var notifyNumGr = new GlideRecord('notify_number');
    notifyNumGr.addActiveQuery();
    notifyNumGr.addQuery('has_sms_out', 'yes');
    notifyNumGr.query();
    if (notifyNumGr.next()) {
        return notifyNumGr.number + '';```
NotifySMS - sendToUsers(String fromNumber, Object toGr, String message, Object sourceRecord)

Sends a Short Message Service (SMS) message from a specified Notify phone number to users found in a specified GlideRecord.

This method extracts the destination telephone numbers from the passed-in GlideRecord. In addition, you can optionally associate the Incident record that caused the SMS message to be generated with the SMS message.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromNumber</td>
<td>String</td>
<td>Notify telephone number sending the SMS message.</td>
</tr>
<tr>
<td>message</td>
<td>String</td>
<td>Message to send.</td>
</tr>
<tr>
<td>sourceRecord</td>
<td>GlideRecord</td>
<td>Optional. Incident GlideRecord to store in the Source field of the associated SMS message record in the Notify Message [notify_message] table. This links the Incident record that caused the SMS message to be generated to that SMS message. Default: None. If this parameter is not passed in, this information is not tracked.</td>
</tr>
<tr>
<td>toGr</td>
<td>GlideRecord</td>
<td>GlideRecord of a record type that contains a field that resolves to a user's telephone number. The record type must correlate with the record type used by the phone number resolver that is currently implemented. A phone number resolver is simply a method that obtains the user telephone number from a specific type of GlideRecord, such as a User record or a Case record. When using the default resolver, the records correspond to the User [sys_user] table. To change the type of record that the resolver uses to obtain the telephone number, use the NotifySMS - setRecordToNumberResolver(Object scriptIncludeInstance, String methodName) method.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
This example shows how to send an SMS message to users within a specified incident record.

```
var sourceRecord = new GlideRecord('incident');
sourceRecord.query();
if (sourceRecord.next()) {
    var fromNumber = getFromNumber();
    var groupId = sourceRecord.assignment_group + '';
    var toGr = getRecipientGRs(groupId);
    var message = 'An Incident ' + sourceRecord.getDisplayValue() + ' has been assigned to your group';
    if (fromNumber && toGr) {
        var notifySMS = new NotifySMS();
        notifySMS.sendToUsers(fromNumber, toGr, message, sourceRecord);
    }
}
```

```
function getFromNumber() {
    var prop = gs.getProperty('custom_property_name', '');
    if (!prop){
        return getFallbackFromNumber();
    }
    return prop;
}
```

```
function getFallbackFromNumber() {
    var notifyNumGr = new GlideRecord("notify_number");
    notifyNumGr.addActiveQuery();
    notifyNumGr.addQuery('has_sms_out', 'yes');
    notifyNumGr.query();
    if (notifyNumGr.next()) {
        return notifyNumGr.number + '';
    }
    return ''; 
}
```

```
function getRecipientGRs(groupId) {
    if (!groupId)
```
```
return;
var userMemberGr = new GlideRecord('sys_user_grmember');
userMemberGr.addQuery('group', groupId);
userMemberGr.query();
var userIds = [];
while (userMemberGr.next()) {
    userIds.push(userMemberGr.user + '');
}
var userGr = new GlideRecord('sys_user');
userGr.addActiveQuery();
userGr.addQuery('sys_id', 'IN', userIds.join(','));
userGr.query();
return userGr;
}
```

**NotifySMS - setRecordToNumberResolver(Object scriptIncludeInstance, String methodName)**

Sets the method within a script include to use to obtain the target telephone number in the GlideRecord that is passed in the `NotifySMS.sendToUser()` and `NotifySMS.sendToUsers()` methods.

By default, the `NotifyUtil.getSMSNumberForUser()` method is used to obtain the telephone number from a User [sys_user] based GlideRecord. Use the `setRecordToNumberResolver()` method if you need to create a custom method to obtain this information from a different record type, such as a Case record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>methodName</td>
<td>String</td>
<td>Name of the method to use to obtain the target telephone number in a GlideRecord.</td>
</tr>
<tr>
<td>scriptIncludeInstance</td>
<td>Object</td>
<td>Instance of the script include that contains the specified number resolver method.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to set the phone number resolver to the `getSMSNumberForConsumer()` method within the `csmUtils` script include.
// Script include that contains the logic to retrieve a phone number for a consumer.
var csmUtils = new global.CSMExampleUtils();

// Example Case record used as source for sending SMS
var caseGr = new GlideRecord('sn_customerservice_case');
caseGr.get('3af265b3d1cb5010f877cb055160faxx'); // Case sys_id

var notifySms = new global.NotifySMS();

// Use the getSMSNumberForConsumer method of the csmUtils script include (CSMExampleUtils)
to obtain the phone number of the consumer.
notifySms.setRecordToNumberResolver(csmUtils, 'getSMSNumberForConsumer');

// Send an SMS message to a user whose telephone number is in the specified Case GlideRecord.
var smsText = 'This is a test message for ' + caseGr.getValue('number');
notifySms.sendToUser('+1xxxxxxxxxx', caseGr.getValue('consumer'), smsText, caseGr); // First param is the Notify phone number

This example shows the code needed for the telephone resolver method getSMSNumberForConsumer().

getSMSNumberForConsumer: function(consumerGrOrId) {
    if (!consumerGrOrId) {
        return;
    }

    var consumerId, consumerGr;
    if (typeof consumerGrOrId === "string") {
        consumerId = consumerGrOrId;
    } else if (typeof consumerGrOrId === "object" && consumerGrOrId["isValidRecord"]) {
        consumerGr = consumerGrOrId;
        consumerId = consumerGr.getUniqueValue();
    }

    if (!consumerId) {
        return;
    }

    if (!consumerGr) {
        consumerGr = new GlideRecord('csm_consumer');
        consumerGr.get(consumerId);
    }

    if (consumerGr.isValidRecord()) {
        return consumerGr.getValue('mobile_phone');
    }
}
NotifyNow API (Legacy) - Global

The legacy NotifyNow API provides functionality for sending emails, sending SMS messages, and setting up conference calls.

Use this when you want to use Notify functionality with applications on your system.

⚠️ Note: This API is included with the legacy Notify functionality. For APIs included in the current Notify feature, see the Notify, NotifyAction, NotifyPhoneNumber, and NotifyClient APIs.

NotifyNow - addConferenceCallParticipant(String conferenceCall, String participant)

Adds ad-hoc users to an ongoing conference call.

When the method is called with a phone number for the participant parameter and there is exactly one sys_user record that matches the phone number, that sys_user record will be related to the participant. The participant's phone number field will be left blank because the phone number is in the sys_user record. If there are several sys_user records that match the phone number, or if there are no results, the participant's phone number field will be filled in, and there will be no stored reference to sys_user because the user is not known.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>conferenceCall</td>
<td>The sys_id or GlideRecord of an active conference call.</td>
</tr>
<tr>
<td>participant</td>
<td>The sys_id or GlideRecord of a user with an E.164-compliant phone number, or an E.164-compliant phone number.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The participant record of the new participant that was added to the conference call.</td>
</tr>
</tbody>
</table>
// add a new participant by conference call sys_id (string) and phone number (string)
var nn = new SNC.NotifyNow();
gs.log(nn.addConferenceCallParticipant('d193b242eb020100a04d4910f206fe39', '+31612345678'));

// add a new participant by conference call sys_id (string) and user record (GlideRecord)
var user = new GlideRecord('sys_user');
user.query('user_name', 'myUserName');
if (user.hasNext() && user.next()) {
  var nn = new SNC.NotifyNow();
  gs.log(nn.addConferenceCallParticipant('d193b242eb020100a04d4910f206fe39', user));

  // you could have added the user by sys_id as well:
  // nn.addConferenceCallParticipant('d193b242eb020100a04d4910f206fe39',
  // user.getValue('sys_id'));
} else {
  gs.log('no such user');
}

// add a new participant by conference call record (GlideRecord) and phone number (string)
var conferenceCall = new GlideRecord('notifynow_conference_call');
conferenceCall.query('title', 'IA0001001');
if (conferenceCall.hasNext() && conferenceCall.next()) {
  var nn = new SNC.NotifyNow();
  gs.log(nn.addConferenceCallParticipant(conferenceCall, '+31612345678'));
} else {
  gs.log('no such conference call');
}

**NotifyNow - getReadyState()**

Indicates whether Notify is set up correctly or not.

This method can only be accessed by administrators or users with the notifynow_admin role. Users with all other roles get the message False when trying to run the function in a script.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if Notify is set up correctly, otherwise false.</td>
</tr>
</tbody>
</table>

```java
var nn = new SNC.NotifyNow();
gs.log(((nn.getReadyState()) ? "OK" : "NOT OK"));
```

**NotifyNow - getStatus()**

Returns the current status of Notify configuration.

This method can only be accessed by administrators or users with the notifynow_admin role. Users with all other roles get the message Unauthorized when trying to run the function in a script.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>One of the possible status messages.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO_NUMBER_MESSAGE</td>
<td>The account does not have a telephone number set up.</td>
</tr>
<tr>
<td>NO_ENDPOINTS_MESSAGE</td>
<td>The account does not have its endpoints set up correctly.</td>
</tr>
<tr>
<td>ACCOUNT_OK_MESSAGE</td>
<td>The account is active and ready for use.</td>
</tr>
<tr>
<td>ACCOUNT_NO_AUTH</td>
<td>The Twilio AuthToken is not valid.</td>
</tr>
<tr>
<td>ACCOUNT_NOT_CONFIGURED</td>
<td>The Twilio AccountSID or AuthToken is not valid.</td>
</tr>
</tbody>
</table>

```java
var nn = new SNC.NotifyNow();
gs.log(nn.getStatus());
```
NotifyNow - initiateConferenceCall(String[] conferenceCallParticipants, String conferenceCallTitle)

Initiate a new conference call.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>conferenceCallParticipants</td>
<td>String</td>
<td>One or more users, conference call participants, identified by the sys_ids from the sys_user table or E.164-compliant phone numbers.</td>
</tr>
<tr>
<td>conferenceCallTitle</td>
<td>String</td>
<td>Title of the conference call. This parameter has a maximum length of 40 characters.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The conference call record, or null if there was an error.</td>
</tr>
</tbody>
</table>

This initiates a conference call with E.164-compliant phone numbers for participants, without the optional source record parameter and and does not send any conference call details via SMS or email.

```javascript
var participants = ['+31205655548', '+31205655552', '+31652825393'];
// set up conference call
var nn = new SNC.NotifyNow();
var conferenceCall = nn.initiateConferenceCall(participants, "testing12");
gs.log('started conference call: ' + conferenceCall.getUniqueValue());
```

NotifyNow - initiateConferenceCall(String[] conferenceCallParticipants, String conferenceCallTitle, GlideRecord sourceRecord, Boolean private)

Initiate a new conference call.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>conferenceCallParticipants</td>
<td>String</td>
<td>One or more users, conference call participants, identified by the sys_ids from the sys_user table or E.164-compliant phone numbers.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>conferenceCallTitle</td>
<td>String</td>
<td>Title of the conference call. This parameter has a maximum length of 40 characters.</td>
</tr>
<tr>
<td>sourceRecord</td>
<td>GlideRecord</td>
<td>Source record to associate to the conference call such as an incident or problem number.</td>
</tr>
<tr>
<td>private</td>
<td>Boolean</td>
<td>Value to control if a conference call is private. This value defaults to false.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The conference call record, or null if there was an error.</td>
</tr>
</tbody>
</table>

This initiates a conference call with participants that have a E.164-compliant phone number and participants from the sys_user table and sends conference call details via SMS and email to all participants.

```javascript
// define phone number participants
var participants = ['+31205655548', '+31205655552', '+31652825393'];

// we also want to add two Dutch sys_user participants
var user = new GlideRecord('sys_user');
user.addNotNullQuery('mobile_phone');
user.addQuery('mobile_phone', 'STARTSWITH', '+316');
user.setLimit(2);
user.query();

// add users to the participant array
while (user.hasNext() && user.next()) {
    gs.log('adding user ' + user.getValue('name') + ' with phone number ' +
            user.getValue('mobile_phone') + ' to the participant array');
    participants.push(user.getUniqueValue());
}

// define a source record to associate with the conference call
var source = new GlideRecord("cmdb_ci");
```
source.query("asset_tag", "P1000167");
if (source.hasNext() && source.next()) {
    // set up conference call
    var nn = new SNC.NotifyNow();
    var conferenceCall = nn.initiateConferenceCall(participants, "testing 1 2", source);
    // check if the conference call was successfully created
    if (conferenceCall != null) {
        gs.log('started conference call: ' + conferenceCall.getUniqueValue());
    } else {
        gs.log('could not start the conference call :(');
    }
}

### NotifyNow - isCallable(String participant)

Determines whether a user is callable or not.

A user must have a valid phone number to be callable. A user who is already in an active session is not callable.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>participant</td>
<td>String or GlideRecord</td>
<td>A sys_user or notifynow_participant record, or an E.164-compliant phone number.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>Whether this participant can be called or not.</td>
</tr>
</tbody>
</table>

var nn = new SNC.NotifyNow();
gs.log('by number: ' + nn.isCallable('+31612345678'));

var user = GlideRecord('sys_user');
user.query('sys_id', '13d39544eb5201003cf587b9d106fea9');
if (user.hasNext() && user.next())
    gs.log('by user: ' + nn.isCallable(user));

var participant = GlideRecord('notifynow_participant');
participant.query('sys_id', '33b11430eb1201003cf587b9d106feb9');
if (participant.hasNext() && participant.next())
gs.log('by participant: ' + nn.isCallable(participant));

**NotifyNow - isSMSCapable()**
Checks if the telephone number associated with the Twilio account is capable of sending SMS messages.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Whether the telephone number associated with the Twilio account is capable of sending SMS messages.</td>
</tr>
</tbody>
</table>

```
gs.log('The twilio number is SMS capable: ' + ((new SNC.NotifyNow().isSMSCapable()) ? 'yes' : 'no'));
```

**NotifyNow - isSMSCapable(String userID)**
Checks if a user is able to send SMS messages.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userID</td>
<td>String</td>
<td>The sys_id of the user you want to check for an SMS-capable phone number.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>If the user can send SMS messages.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
gs.log('the user is able to send SMS messages (e.g. has a SMS device): ' + (new SNC.NotifyNow().isSMSCapable('<user sys_id>')) ? 'yes' : 'no'));

NotifyNow - isVoiceCapable()

Checks if the telephone number associated with the Twilio account is capable of setting up phone calls.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Whether the telephone number associated with the Twilio account is capable of setting up phone calls.</td>
</tr>
</tbody>
</table>

gs.log('the Twilio number is Voice capable: ' + ((new SNC.NotifyNow().isVoiceCapable()) ? 'yes' : 'no'));

NotifyNow - isVoiceCapable(String userID)

Checks if a user is able to make voice calls.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userID</td>
<td>String</td>
<td>The sys_id of the user you want to check for a voice-call capable phone number.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>Whether the user has a voice-call capable phone number.</td>
</tr>
</tbody>
</table>
gs.log('the user is able to send SMS messages (e.g. has a SMS device): ' +
        ((new SNC.NotifyNow().isVoiceCapable('someuserid')) ? 'yes' : 'no'));

**NotifyNow - kick(GlideRecord participant)**
Removes a participant from a conference call.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>participant</td>
<td>GlideRecord</td>
<td>The conference call participant to remove from the call.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the participant was removed, otherwise false.</td>
</tr>
</tbody>
</table>

```javascript
var participantId = "<participant sys_id>";
var participant = new GlideRecord('notifynow_participant');
participant.get(participantId);
if (participant.isValid()) {
    // kick participant
    result = new SNC.NotifyNow().kick(participant);
    gs.log('participant kicked: ' + result);
}
```

**NotifyNow - mute(GlideRecord participant)**
Mutes a participant on a conference call.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>participant</td>
<td>GlideRecord</td>
<td>The conference call participant to mute.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the participant was muted, otherwise false.</td>
</tr>
</tbody>
</table>

```javascript
var participantId = "<participant sys_id>";
var participant = new GlideRecord('notifynow_participant');
participant.get(participantId);
if (participant.isValid()) {
    // mute participant
    result = new SNC.NotifyNow().mute(participant);
    gs.log('participant muted: ' + result);
}
```

**NotifyNow - umute(GlideRecord participant)**

Unmutes a participant on a conference call.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>participant</td>
<td>GlideRecord</td>
<td>The muted conference call participant to unmute.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the participant was unmuted, otherwise false.</td>
</tr>
</tbody>
</table>

```javascript
var participantId = "<participant sys_id>";
var participant = new GlideRecord('notifynow_participant');
participant.get(participantId);
if (participant.isValid()) {
    // unmute participant
    result = new SNC.NotifyNow().unmute(participant);
    gs.log('participant unmuted: ' + result);
}
```
**NotifyNow - sendEmailQuestion(String emailAddress, String question, GlideRecord sourceRecord, String emailSubject)**

Send an email question to an email address.

The `sendEmailQuestion` method produces a question body and requires users to click a link to indicate their choice.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>emailAddress</td>
<td>String</td>
<td>Email address to send the question to.</td>
</tr>
<tr>
<td>question</td>
<td>String or GlideRecord</td>
<td>The question record to send or the sys_id of a question record.</td>
</tr>
<tr>
<td>sourceRecord</td>
<td>GlideRecord</td>
<td>An optional source record to associate to the SMS question, such as an incident.</td>
</tr>
<tr>
<td>emailSubject</td>
<td>String</td>
<td>Optional text to override the default email subject.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The conversation sys_id.</td>
</tr>
</tbody>
</table>

This example demonstrates using the default email subject.

```javascript
var user = GlideRecord("sys_user");
user.get("email", "someone@somedomain.com");

new SNC.NotifyNow().sendEmailQuestion(user.getValue('email'),
    "b6b34500bf3111003cf585ce2c0739ce", user);
```

This example uses dot-walking and specifies a source record and email subject.

```javascript
new SNC.NotifyNow().sendEmailQuestion("someone@somedomain.com",
    "b6071733bf1111003cf585ce2c07390f", current,
    "Please answer this question");
```

This example uses dot-walking and specifies an email subject but no source record.
new SNC.NotifyNow().sendEmailQuestion("someone@somedomain.com", "b6071733bf1111003cf585ce2c07390f", "Please answer this question");

**NotifyNow - sendSMS(String phoneNumber, String smsBody)**

Sends an SMS message to an E.164-compliant mobile phone number.

Notify supports international numbers. Using this method with a number that does not support sending SMS messages results in an error being logged.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phoneNumber</td>
<td>String</td>
<td>The E.164-compliant phone number to send the message to.</td>
</tr>
<tr>
<td>smsBody</td>
<td>String</td>
<td>The message to send, maximum 1600 characters.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

new SNC.NotifyNow().sendSMS("+31612345678", "This is a message without source record");

**NotifyNow - sendSMS(String phoneNumber, String smsBody, GlideRecord source)**

Sends an SMS message to an E.164-compliant mobile phone number.

Notify supports international numbers. Using this method with a number that does not support sending SMS messages results in an error being logged.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phoneNumber</td>
<td>String</td>
<td>The E.164-compliant phone number to send the message to.</td>
</tr>
<tr>
<td>smsBody</td>
<td>String</td>
<td>The message to send, maximum 1600 characters.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>GlideRecord</td>
<td>The source record to associate with this SMS message.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var source = new GlideRecord("my_table");
source.query("my_field", "my_value");

if (source.hasNext() && source.next()) {
    // send a text message
    var nn = new SNC.NotifyNow();
    var message = "this is just a test";
    var number = "+31612345678";
    nn.sendSMS(number, message, source);
}
```

This example uses dot-walking and the current record as the source record.

```javascript
new SNC.NotifyNow().sendSMS("+31612345678", "this is a test", current);
```

**NotifyNow - sendSMSQuestion(String phoneNumber, String question, GlideRecord sourceRecord)**

Sends an SMS question.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phoneNumber</td>
<td>String</td>
<td>An E.164-compliant phone number to send the message to.</td>
</tr>
<tr>
<td>question</td>
<td>String or GlideRecord</td>
<td>The question record to send or the sys_id of a question record.</td>
</tr>
<tr>
<td>sourceRecord</td>
<td>GlideRecord</td>
<td>An optional source record to associate to the SMS question, such as an incident.</td>
</tr>
</tbody>
</table>
var question = new GlideRecord("notifynow_question");
question.query();

// get the first question
if (question.hasNext() && question.next()) {
    // send the sms question
    var number = "+31612345678";
    var nn = new SNC.NotifyNow();
    nn.sendSMSQuestion(number, question.getUniqueValue(), current);
}

NotifyNow - convertLocalPhoneNumberToE164(String userID, String phoneNumber)

Converts a local phone number to an E.164-compliant phone number based on a user's location.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userID</td>
<td>String</td>
<td>The sys_id of a sys_user record to get location information from.</td>
</tr>
<tr>
<td>phoneNumber</td>
<td>String</td>
<td>The phone number.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The E.164-compliant phone number.</td>
</tr>
</tbody>
</table>

var localPhoneNumber = '01784 221600';
var userName = 'Heath Vanalphen';

var user = new GlideRecord('sys_user');
user.get('name', userName);
var E164Number = new SNC.NotifyNow().convertLocalPhoneNumberToE164(user.getUniqueValue(), localPhoneNumber);

gs.log('converted: ' + localPhoneNumber + ' to ' + E164Number + ' based on ' + user.getValue('name') + '\''s location (' + user.getValue('location') + ') ');

### NotifyNow - getConferenceCallParticipants(String conferenceCallId, Boolean isCallable)

Returns all participants for a conference call.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>conferenceCallId</td>
<td>String</td>
<td>The ID of the conference call.</td>
</tr>
<tr>
<td>isCallable</td>
<td>Boolean</td>
<td>An optional flag to return either only the users you can call (true) or those you cannot call (false).</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The participants</td>
</tr>
</tbody>
</table>

var nn = new SNC.NotifyNow();
var user = nn.getConferenceCallParticipants('c2e91710eb120100f34087b9d106fe37');

while (user.hasNext() && user.next()) {
    if (user.getValue('participant')) {
        gs.log('user: ' + user.getValue('sys_id'));
    } else {
        gs.log('phone number: ' + user.getValue('phone_number'));
    }
}

var nn = new SNC.NotifyNow();
var user = nn.getConferenceCallParticipants('c2e91710eb120100f34087b9d106fe37', true);

while (user.hasNext() && user.next()) {
    if (user.getValue('participant')) {
        gs.log('user: ' + user.getValue('sys_id'));
    }
}
var conferenceCallId = '32b11430eb1201003cf587b9d106feb8';

// get all participants
gs.log('all conference call participants:');
var nn = new SNC.NotifyNow();
var user = nn.getConferenceCallParticipants(conferenceCallId);
gs.log(user);

// get all callable participants
gs.log('all conference call participants we can call:');
user = nn.getConferenceCallParticipants(conferenceCallId, true);
gs.log(user);

// get all un callable participants
gs.log('all conference call participants that are already in an active session and whom we cannot call:');
user = nn.getConferenceCallParticipants(conferenceCallId, false);
gs.log(user);

NotifyNow - getFrequentlyCalledUsers(Number limit)

Returns a number of frequently-called users, up to the limit parameter, in alphabetical order.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>limit</td>
<td>Number</td>
<td>The maximum number of results.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The frequently called users in alphabetical order.</td>
</tr>
</tbody>
</table>
while (fc.hasNext() && fc.next()) {
    gs.log("got user " + fc.getValue('name') + ' - ' + fc.getValue('sys_id'));
}

### NotifyNow - getPreferredE164SMSNumber(GlideRecord user)

Returns a user's preferred E.164-compliant phone number for SMS messages.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>GlideRecord or String</td>
<td>The user record or the sys_id of a user to get the E.164-compliant phone number from.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The E.164-compliant phone number or null.</td>
</tr>
</tbody>
</table>

```javascript
var userID = "<user sys_id>";
var E164Number = new SNC.NotifyNow().getPreferredE164SMSNumber(userID);
gs.log('the preferred phone number for sending SMS notifications is ' + E164Number + ' for user with id: ' + userID);
```

### NotifyNow - getPreferredE164VoiceNumber(GlideRecord user)

Returns a user's preferred E.164-compliant phone number for voice calls.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>GlideRecord or String</td>
<td>The user record or the sys_id of a user to get the E.164-compliant phone number from.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The E.164-compliant phone number or null.</td>
</tr>
</tbody>
</table>
var userID = "<user sys_id>";
var E164Number = new SNC.NotifyNow().getPreferredE164VoiceNumber(userID);
gs.log('the preferred phone number for setting up voice calls is ' + E164Number + ' for user with id: ' + userID);

NotifyNow - getPreferredEmailAddress(GlideRecord user)
Returns a user's preferred email address

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>GlideRecord or String</td>
<td>The user record or the sys_id of a user to get the email address from.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The email address or null.</td>
</tr>
</tbody>
</table>

var userID = "some user sys id";
var email = new SNC.NotifyNow().getPreferredEmailAddress(userID);
gs.log('the preferred email address for sending email notifications is ' + email + ' for user with id: ' + userID);

NotifyUtil - Global
The NotifyUtil API provides utility methods to use when interacting with Notify calls and SMS messages using server-side scripts.

To use this API you must activate the Notify (com.snc.notify) plugin.

Using the NotifyUtil API you can:

- Obtain all of the Notify telephone numbers and related Notify information from a specified source record.
- Obtain a list of unique Notify telephone numbers.
- Determine whether there are any active conference calls for the specified source record.
- Obtain the SMS-capable number associated with the specified Notify user.
- Validate a specified Notify telephone number.
NotifyUtil - NotifyUtil()

Instantiates a NotifyUtil class object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This example instantiates a NotifyUtil object.

```javascript
var notifyUtil = new NotifyUtil();
notifyUtil.getTelephonyProviders();
```

NotifyUtil - getListOfNotifyNumbersAndProviders(String sourceTable, String sourceSysId, String notifyGroupSelectorSysId, Boolean filterSMSCapableNums)

Returns all of the Notify telephone numbers and related Notify information from a specified source record, such as an incident.

You can use this information to initiate a call or send an SMS message on a particular source record. The information that is returned is based on the configuration of the Notify Provider Selector framework. For additional information, see Notify

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| filterSMSCapableNums        | Boolean    | Optional. Flag that indicates if only numbers that are SMS-capable should be returned. Valid values:  
  • true: Only return SMS-capable telephone numbers and information.  
  • false: Return all notify telephone numbers and information.  
  Default: false |
| notifyGroupSelectorSysId    | String     | Optional. Sys_id of a Notify group for which to return the notify numbers and information.  
  Default: All groups |
| sourceSysId                 | String     | Sys_id of the source record for which to return the Notify numbers and information. For example, this |
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>could be the sys_id of a record in the Incident [incident] table.</td>
</tr>
<tr>
<td>sourceTable</td>
<td>String</td>
<td>Name of the table that contains the source record that contains the desired Notify numbers and information.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>confProviders</td>
<td>List of available conference providers.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>numbers</td>
<td>List of objects, each describing a single Notify number.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;numbers&quot;: [{</td>
</tr>
<tr>
<td></td>
<td>&quot;defaultFlag&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;number&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;shortCode&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;sysId&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}]</td>
</tr>
</tbody>
</table>

- **numbers.defaultFlag**: Flag that indicates whether the associated Notify number is the default number. Possible values:
  - `true`: Default number
  - `false`: Not the default number
  - Data type: Boolean

- **numbers.name**: Name or label of the number.
  - Data type: String

- **numbers.number**: Notify number.
  - Data type: String

- **numbers.shortCode**: Flag that indicates whether the associated Notify number is a short code.
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Possible values:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>• true: Short code</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Not a short code</td>
</tr>
<tr>
<td>numbers.sysId</td>
<td>Sys_id of the Notify number.</td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

This example shows how to obtain the Notify telephone numbers and related Notify information from a specified source record.

```
function updateConferenceBridges(sourceTable, sourceId) {
    var notifyUtil = new global.NotifyUtil();
    var numbersAndProviders = notifyUtil.getListOfNotifyNumbersAndProviders(sourceTable, sourceId);
    var confBridges = [];
    if (numbersAndProviders.confProviders) {
        numbersAndProviders.confProviders.forEach(function(provider){
            confBridges.push(provider);
        });
    }
    if (numbersAndProviders.numbers) {
        numbersAndProviders.numbers.forEach(function(number){
            confBridges.push(number.name);
        });
    }
}
```

**NotifyUtil - getSMSNumberForUser(String userGROrId)**

Returns the SMS-capable number associated with the specified Notify user.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>user for whom to return the SMS-capable telephone number.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>User SMS-capable telephone number. Returns null if the specified user is not found.</td>
</tr>
</tbody>
</table>

This example shows how to obtain an SMS-capable telephone number using the associated GlideRecord.

```javascript
var sourceRecord = new GlideRecord('incident');
sourceRecord.query();
if (sourceRecord.next()) {
    var fromNumber = getFromNumber();
    var nUtil = new NotifyUtil();
    var toNumber = nUtil.getSMSNumberForUser(sourceRecord.assigned_to.getRefRecord());
    var message = 'Incident ' + sourceRecord.getDisplayValue() + ' has been assigned to you.';
    if (fromNumber && nUtil.validateOutboundNotifyPhoneNumber(fromNumber) && toNumber && 
        nUtil.validatePhoneNumber(toNumber)) {
        var notifySMS = new NotifySMS();
        notifySMS.sendToNumber(fromNumber, toNumber, message, sourceRecord);
    }
}

function getFromNumber() {
    var prop = gs.getProperty('custom_property_name', '');
    if (!prop) {
        return getFallbackFromNumber();
    }
    return prop;
}

function getFallbackFromNumber() {
    var notifyNumGr = new GlideRecord('notify_number');
    notifyNumGr.addActiveQuery();
    notifyNumGr.addQuery('has_sms_out', 'yes');
    notifyNumGr.query();
}
```
if (notifyNumGr.next()) {
    return notifyNumGr.number + '';
}
return '';

NotifyUtil - getUniquePhoneNumbersForUsersAndGroups(Array numbers, Array users, Array groups, String type, Boolean getData)

Returns a list of unique Notify telephone numbers.

If you don't pass any parameters in the call, all Notify numbers within the Notify Phone Number [notify_number] table are checked for duplicates, with each available phone number only appearing once in the returned list. You can refine the return results by specifying a list of users or groups to check, or by specifying a set of numbers or number types (SMS or voice.) You can also request that the metadata associated with each number be returned along with the unique numbers. If you do not want to use a parameter, simply pass `null` as a placeholder. For example: `return nUtil.getUniquePhoneNumbersForUsersAndGroups(null, userIds, null, 'sms', false);`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getData</td>
<td>Boolean</td>
<td>Optional. Flag that indicates whether to return metadata along with the list of unique phone numbers. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Return metadata.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not return metadata.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>groups</td>
<td>Array</td>
<td>Optional. List of sys_id groups to check. Located in the Group [sys_user_group] table. Default: Check all groups.</td>
</tr>
<tr>
<td>numbers</td>
<td>Array</td>
<td>Optional. List of specific Notify telephone numbers to check.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Check all phone numbers.</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>Optional. Type of telephone numbers to check.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values (case-sensitive):</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>voice</td>
<td>• voice • sms</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Default: Check all phone number types</td>
<td></td>
</tr>
<tr>
<td>users</td>
<td>Array</td>
<td>Optional. List of sys_ids of specific users to check. Located in the User [sys_user] table. Default: Check all users</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numbers</td>
<td>Unique Notify telephone numbers. Data type: Array</td>
</tr>
<tr>
<td>result</td>
<td>Only returned if <code>getData</code> is set to true. Metadata associated with each unique number. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;number&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sysId&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;valid&quot;: Boolean</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.number</td>
<td>Unique Notify telephone number. Data type: String</td>
</tr>
<tr>
<td>result.sysId</td>
<td>Sys_id of the record that contains the Notify telephone number. Located in the Notify Phone Number [notify_number] table. Data type: String</td>
</tr>
<tr>
<td>result.type</td>
<td>Always contains &quot;u&quot; for user. Data type: String</td>
</tr>
<tr>
<td>result.valid</td>
<td>Flag that indicates whether the Notify telephone number is in valid E.164 format.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Valid E.164 format.</td>
</tr>
<tr>
<td></td>
<td>• false: Not in E.164 format.</td>
</tr>
</tbody>
</table>

Data type: Boolean

This example shows how to request a specific set of unique Notify telephone numbers that have SMS capabilities.

```javascript
var fromNumber = getFromNumber();
var toNumbers = getRecipientNumbers();
var message = 'This is an example SMS';
var sourceRecord = new GlideRecord('incident');
sourceRecord.query();
if (sourceRecord.next()) {
    var notifySMS = new NotifySMS();
    notifySMS.sendToNumber(fromNumber, toNumbers, message, sourceRecord);
}

function getRecipientNumbers() {
    var userGr = new GlideRecord('sys_user');
    userGr.addActiveQuery();
    userGr.addQuery('first_name', 'STARTSWITH', 'A');
    userGr.setLimit(5);
    userGr.query();
    var userIds = [];
    while (userGr.next()) {
        userIds.push(userGr.getUniqueValue());
    }
    if (userIds.length > 0) {
        var nUtil = new NotifyUtil();
        return nUtil.getUniquePhoneNumbersForUsersAndGroups(null, userIds, null, 'sms', false);
    }
}

function getFromNumber() {
    var prop = gs.getProperty('custom_property_name', '');
    if (!prop){
        return getFallbackFromNumber();
    }
}
function getFallbackFromNumber() {
    var notifyNumGr = new GlideRecord("notify_number");
    notifyNumGr.addActiveQuery();
    notifyNumGr.addQuery('has_sms_out', 'yes');
    notifyNumGr.query();
    if (notifyNumGr.next()) {
        return notifyNumGr.number + ' ';
    }
    return '';
}

NotifyUtil - hasActiveConferenceCalls(String sourceRecSysId)
Determines whether there are any active conference calls for the specified source record.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceRecSysId</td>
<td>String</td>
<td>Sys_id of the record to check for active conference calls. For example the sys_id of a record in the Incident table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean   | Flag that indicates whether the specified record has associated active conference calls. Possible values:  
• true: Active conference calls are available for the specified record.  
• false: No active conference calls. |

This example displays an information message if there are any active conference calls associated with an incident record

(function executeRule(current, previous /*null when async*/) {
    var nUtil = new NotifyUtil();
    if (nUtil.hasActiveConferenceCalls(current.getUniqueValue())) {

return prop;
}
gs.addInfoMessage("There are active conference calls related to this Incident.");
}
})(current, previous);

**NotifyUtil - validateOutboundNotifyPhoneNumber(String number)**

Validates a specified Notify telephone number.

The method performs three types of validation:

1. Whether the Notify number exists in the Notify Phone Number [notify_number] table.
2. Whether the Notify number has a Notify group associated with it.
3. Whether the Notify number is active.

If any one of these validations fail, the method throws an exception.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>number</td>
<td>String</td>
<td>Notify number to validate.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example illustrates how to validate a notify number.

```javascript
var sourceRecord = new GlideRecord('incident');
sourceRecord.query();
if (sourceRecord.next()) {
    var fromNumber = getFromNumber();
    var nUtil = new NotifyUtil();
    var toNumber = nUtil.getSMSNumberForUser(sourceRecord.assigned_to.getRefRecord());
    var message = 'Incident ' + sourceRecord.getDisplayValue() + ' has been assigned to you.';
    if (fromNumber && nUtil.validateOutboundNotifyPhoneNumber(fromNumber) && toNumber &&
        nUtil.validatePhoneNumber(toNumber)) {
        var notifySMS = new NotifySMS();
        notifySMS.sendToNumber(fromNumber, toNumber, message, sourceRecord);
    }
}
```
function getFromNumber() {
    var prop = gs.getProperty('custom_property_name', '');
    if (!prop){
        return getFallbackFromNumber();
    }
    return prop;
}

function getFallbackFromNumber() {
    var notifyNumGr = new GlideRecord("notify_number");
    notifyNumGr.addActiveQuery();
    notifyNumGr.addQuery('has_sms_out', 'yes');
    notifyNumGr.query();
    if (notifyNumGr.next()) {
        return notifyNumGr.number + '';
    }
    return '';
}

NotifyUtil - validatePhoneNumber(String number)
Verifies that the specified number is a valid E.164 telephone number.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>number</td>
<td>String</td>
<td>Telephone number to validate.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the specified number is a valid telephone number.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Valid E.164 telephone number.</td>
</tr>
<tr>
<td></td>
<td>• false: Invalid telephone number.</td>
</tr>
</tbody>
</table>

This example illustrates how to validate a telephone number.

```javascript
var sourceRecord = new GlideRecord('incident');
sourceRecord.query();
```
if (sourceRecord.next()) {
    var fromNumber = getFromNumber();
    var nUtil = new NotifyUtil();
    var toNumber = nUtil.getSMSNumberForUser(sourceRecord.assigned_to.getRefRecord());
    var message = 'Incident ' + sourceRecord.getDisplayValue() + ' has been assigned to you.';
    if (fromNumber && nUtil.validateOutboundNotifyPhoneNumber(fromNumber) && toNumber &&
        nUtil.validatePhoneNumber(toNumber)) {
        var notifySMS = new NotifySMS();
        notifySMS.sendToNumber(fromNumber, toNumber, message, sourceRecord);
    }
}

function getFromNumber() {
    var prop = gs.getProperty('custom_property_name', '');
    if (!prop)
        return getFallbackFromNumber();
    return prop;
}

function getFallbackFromNumber() {
    var notifyNumGr = new GlideRecord("notify_number");
    notifyNumGr.addActiveQuery();
    notifyNumGr.addQuery('has_sms_out', 'yes');
    notifyNumGr.query();
    if (notifyNumGr.next()) {
        return notifyNumGr.number + '';
    }
    return '';}

openFrameAPI - Client

OpenFrame is an omni-present frame that communication partners can use to integrate their systems into the ServiceNow platform.

One of the core requirements is the ability to connect and serve code from different domains that can connect seamlessly with partner subsystems. This cross domain connection is required to keep connections and callbacks registered into communication systems without any cross domain issues.

OpenFrame has two significant parts: one that lives in the ServiceNow application (referred to as TopFrame) and this API that is sourced from the
partner application. This API has the necessary methods to communicate with TopFrame and control the visual features of the OpenFrame.

**Note:** To stay current with reference to the OpenFrame library, use the following resource URI: https://[servicenow instance]/scripts/openframe/latest/openFrameAPI.min.js.

**APIs not supported on Agent Workspace**
The following functionalities are not supported on Agent Workspace:

- openServiceNowList
- OpenCustomURL

**openFrameAPI - hide()**
Hides the OpenFrame in the TopFrame.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

**openFrameAPI - init(Object config, function successCallback, function failureCallback)**
Initialize OpenFrame, must be the first method called.

This method initializes communication to TopFrame and initializes any visual elements passed in the config parameter.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>Object</td>
<td>An object of key value pairs. The possible keys are height, width, title, subTitle, and titleIcon. All keys are optional.</td>
</tr>
<tr>
<td>successCallback</td>
<td>function</td>
<td>The callback function used if the init method succeeds. The openframe configuration stored in the system is passed as a parameter to the callback function.</td>
</tr>
<tr>
<td>failureCallback</td>
<td>function</td>
<td>The callback function used if the init method fails.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var config = { 
  height: 300, 
  width: 200 
};

function handleCommunicationEvent(context) {
  console.log("Communication from Topframe", context);
}

function initSuccess(snConfig) {
  console.log("openframe configuration", snConfig);
  // register for communication event from TopFrame
  openFrameAPI.subscribe(openFrameAPI.EVENTS.COMMUNICATION_EVENT, handleCommunicationEvent);
}

function initFailure(error) {
  console.log("OpenFrame init failed..", error);
}

openFrameAPI.init(config, initSuccess, initFailure);
```

**openFrameAPI - isVisible(function callback)**

Checks to see if the OpenFrame is visible in the TopFrame.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callback</td>
<td>function</td>
<td>The callback function receives a parameter with a value of true or false. True if OpenFrame is visible and false if not visible.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
function callback(isVisible) {
  console.log(isVisible)
}
openFrameAPI.isVisible(callback)
```

**openFrameAPI - openCustomURL(String details)**

Opens a custom URL in TopFrame.

ℹ️ **Note:** This API is not supported on Agent Workspace.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Url</td>
<td>String</td>
<td>A string of 2083 or fewer characters.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
openFrameAPI.openCustomURL('10_cool_things.do');
```

**openFrameAPI - openServiceNowForm(Object details)**

Opens a form URL.
When an agent receives an incoming call, the OpenFrame window displays information such as the account, contact, or consumer. Clicking a link in the OpenFrame window displays the corresponding record.

- In the platform interface, this API opens a form URL in TopFrame.
- For Agent Workspace, this API supports interaction tab management. In Agent Workspace, an interaction record opens in a parent tab and the specified entity record opens in a child tab under the interaction tab.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>details</td>
<td>Object</td>
<td>Key-value pairs that identify the form URL to open.</td>
</tr>
<tr>
<td>&quot;details&quot;: {</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;entity&quot;: &quot;String&quot;;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;interaction_sys_id&quot;: &quot;String&quot;;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;query&quot;: &quot;String&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>details.entity</td>
<td>String</td>
<td>Table or entity name.</td>
</tr>
<tr>
<td>details.interaction_sys_id</td>
<td>String</td>
<td>Optional. Sys_id of the interaction record to open as parent tab in Agent Workspace.</td>
</tr>
<tr>
<td>details.query</td>
<td>String</td>
<td>Query to identify the record to open, such as: query:'sys_id=&lt;record_sys_id&gt;'.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows basic usage in platform:

```javascript
openFrameAPI.openServiceNowForm({entity:'customer_account',
query:'sys_id=447832786f0331003b3c498f5d3ee452',
'interaction_sys_id':'3be092313b711300758ce9b534efc4dd'});
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The following example shows how to use the `query` parameter to create a new record with data provided in the form by using `sysparm_query` and an encoded query to populate the first and last name fields in Workspace:

```javascript
openFrameAPI.openServiceNowForm({
    entity: 'sys_user',
    query: 'sys_id=-1&sysparm_query=first_name=Ivan^last_name=Greggor'
});
```

**openFrameAPI - openServiceNowList(Object details)**

Opens a list URL in TopFrame.

ℹ️ **Note:** This API is not supported on Agent Workspace.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>details</td>
<td>Object</td>
<td>An object of key value pairs. The possible keys are</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• entity, the table name</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• query, an encoded query string</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
openFrameAPI.openServiceNowList({entity:'case', query:'active=true'});
```

**openFrameAPI - setFrameMode(mode)**

Sets the OpenFrame mode.

The mode passed in this API:

- Sets the appropriate icon in the header: collapse or expand
- Raises the relevant event for CTI:
  - `openFrameAPI.EVENTS.COLLAPSE`
  - `openFrameAPI.EVENTS.EXPAND`
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode</td>
<td>String</td>
<td>Set OpenFrame Mode. Enumerated options:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1. openFrameAPI.FRAME_MODE.COLLAPSE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. openFrameAPI.FRAME_MODE.EXPAND</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
openFrameAPI.setFrameMode(openFrameAPI.FRAME_MODE.COLLAPSE);
```

**openFrameAPI - setFrameMode(openFrameAPI.FRAME_MODE.COLLAPSE)**

Sets the OpenFrame height.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Height</td>
<td>Number</td>
<td>Height in pixels</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
openFrameAPI.setHeight(100);
```

**openFrameAPI - setHeight(height)**

Sets the OpenFrame height.

**openFrameAPI - setIcons(Array icons)**

The OpenFrame header can include icons that are placed next to the close icon.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>icons</td>
<td>Array</td>
<td>A list of icon configurations, where each icon configuration is an object with key values <code>imageURL</code>, <code>imageTitle</code>, and any other needed context.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
```

**openFrameAPI - setPresenceIndicator(presence)**

Sets the presence indicator to display agent availability in a workspace.

For more information on configuring OpenFrame, refer to Create an OpenFrame configuration

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| state | String | Presence state of the agent. Default states:
- Available
- Away
- Offline
You can also specify custom states. |
| color | String | Presence indicator color on workspace. Supported colors:
- red
- orange |
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>grey</td>
<td></td>
</tr>
<tr>
<td></td>
<td>green</td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
openframeAPI.setPresenceIndicator('Available', 'green');
```

**openFrameAPI - setSize(Number width, Number height)**

Sets the OpenFrame size.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Should be greater than zero.</td>
</tr>
<tr>
<td>height</td>
<td>Number</td>
<td>Should be greater than zero.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
openFrameAPI.setSize(300, 370);
```

**openFrameAPI - setSubtitle(String subTitle)**

Sets the OpenFrame subtitle.

```javascript
openFrameAPI.setSubtitle('Subtitle here');
```
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subTitle</td>
<td>String</td>
<td>A string of 256 or fewer characters.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
openFrameAPI.setSubtitle('+18888888888');
```

#### openFrameAPI - setTitle(String title)

Sets the OpenFrame title.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>String</td>
<td>A string of 256 or fewer characters.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
openFrameAPI.setTitle('Incoming Call');
```

#### openFrameAPI - setTitleIcon(Object icon)

Sets the OpenFrame's title icon.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>icon</td>
<td>Object</td>
<td>Object of key value pairs. Keys include imageUrl, imageTitle, and any other context needed.</td>
</tr>
</tbody>
</table>

```java
openFrameAPI.setTitleIcon('Incoming Call');
```
openFrameAPI.setTitleIcon({imageURL:'/my/image/path.png', imageTitle:'mute', id:101});

openFrameAPI.setTitleIcon({imageURL:'https://mydomain.com/image/path.png', imageTitle:'mute', id:101});

openFrameAPI.setTitleIcon({imageURL:'https://mydomain.com/image/path.png', imageTitle:'mute', id:101});

**openFrameAPI - setWidth(width)**

Sets the OpenFrame width.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Width</td>
<td>Number</td>
<td>Width in pixels</td>
</tr>
</tbody>
</table>

openFrameAPI.setWidth(100);

**openFrameAPI - show()**

Makes the OpenFrame visible in the TopFrame.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>void</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

`openFrameAPI.show()`

`openFrameAPI - subscribe(openFrameAPIEVENT event, function eventCallback)`
Subscribes to the event.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
<td>openFrameAPIEVENT</td>
<td>One of the following events:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• openframe_agent_off_interaction: PLACEHOLDER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• openframe_awa_agent_presence: In Advanced Work Assignment (AWA), this event occurs when there is any change in the agent presence state. Computer Telephony Integration (CTI) developers can subscribe to the this event to receive presence state changes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• openframe_awa_workitem_accepted: PLACEHOLDER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• openframe_awa_workitem_offered: PLACEHOLDER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• openframe_awa_workitem_rejected: PLACEHOLDER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• openframe_before_destroy: Occurs before the TopFrame is unloaded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• openframeCollapse: Occurs when the collapse icon is clicked on the OpenFrame header.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>openframe_communication</td>
<td>Application-specific and can be customized.</td>
<td></td>
</tr>
<tr>
<td>openframe_communication_failure</td>
<td>Occurs when communication to TopFrame fails.</td>
<td></td>
</tr>
<tr>
<td>openframe_expand</td>
<td>Occurs when the expand icon is clicked on the OpenFrame header.</td>
<td></td>
</tr>
<tr>
<td>openframe_header_icon_clicked</td>
<td>Deprecated. Use openframe_icon_clicked or openframe_title_icon_clicked instead.</td>
<td></td>
</tr>
<tr>
<td>openframe_hidden</td>
<td>Occurs when the OpenFrame is hidden.</td>
<td></td>
</tr>
<tr>
<td>openframe_icon_clicked</td>
<td>Occurs when any icon other than the close icon is clicked on the OpenFrame footer. The callback receives the icon object as a parameter.</td>
<td></td>
</tr>
<tr>
<td>openframe_shown</td>
<td>Occurs when the OpenFrame is shown.</td>
<td></td>
</tr>
<tr>
<td>openframe_title_icon_clicked</td>
<td>Occurs when the title icon is clicked on the OpenFrame. The callback receives the titleIcon object as a parameter.</td>
<td></td>
</tr>
</tbody>
</table>

**eventCallback**
Function called when the specified event occurs.
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| None unless described otherwise.          | Most event subscriptions have no return values, with the following exception(s):

In AWA, the openframe_awa_agent_presence event returns the presence object:

- presence: Information about an agent's current presence state and channel. Output example below.
- presence.name: Name of the agent's presence state.
- presence.available: Flag that indicates whether the agent is available.
- presence.channels: List of objects that describe the available channels of communication with the agent.
- presence.channels.name: Channel name, such as Chat or Phone.
- presence.channels.available: Flag that indicates whether the channel is available.
- presence.channels.restrict_update: Flag that indicates whether the user can restrict updates. |

Example

```javascript
function handleIconClick(context) {
  console.log("Icon was clicked", context);
}
openFrameAPI.subscribe(openFrameAPI.events.openframe_awa_agent_presence, handleIconClick);
```

Output

```javascript
// Sample presence object output
// openframe_awa_agent_presence event only

{
  "result": {
    "presence": {
      "name": "Available",
      "..."  
  }
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
openFrameAPI - version()

Returns the OpenFrame API version.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The OpenFrame API version</td>
</tr>
</tbody>
</table>

```javascript
var version = openFrameAPI.version();
console.log("API version " + version);
```
OAuthUtil - Global

This script include modifies request parameters and parses the token response during runtime.

If the external OAuth provider returns a response other than an application/JSON type response, you can customize your own version of this script include to parse responses that are in different format. Extend or copy this script include, and then reference your version from the OAuth API Script field on the Application Registry form for third-party OAuth providers. The custom script include name must start with OAuth.

For example, if the OAuth provider requires a resource parameter with value https://outlook.office365.com, the code would look like:

```javascript
```

OAuthUtil - interceptRequestParameters(requestParamMap)

Adds a name:value pair to the request parameters.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestParamMap</td>
<td>String</td>
<td>The name:value pair you want to add.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

OAuthUtil - parseTokenResponse(accessTokenResponse)

Parses the token received into a parameter map.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessTokenResponse</td>
<td>String</td>
<td>The access token response that you want to parse.</td>
</tr>
</tbody>
</table>
**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**OCRosterSpanApprovalUtil - Global**

The OCRosterSpanApprovalUtil API for the PTO approval feature.

**OCRosterSpanApprovalUtil - approvePTOSpan (GlideRecord rosterSpanGr)**

Changes the type of the roster_schedule_span to time-off from approval.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rosterSpanGr</td>
<td>GlideRecord</td>
<td>A roster_schedule_span_proposal record.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**OCRosterSpanApprovalUtil - getContextualCalUrlPerSpanProposal (GlideRecord rosterSpanProposalGr)**

Returns a URL to the on-call calendar based on the rosterSpanProposalGr.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rosterSpanProposalGr</td>
<td>GlideRecord</td>
<td>A roster_schedule_span_proposal record.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A URL to the on-call calendar based on the rosterSpanProposalGr.</td>
</tr>
</tbody>
</table>
OCRosterSpanApprovalUtil - getFromDateDisplayValuePerSpanProposal (GlideRecord rosterSpanProposalGr)

Returns a formatted GlideScheduleDateTime string based on the start date of the rosterSpanProposalGr.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rosterSpanProposalGr</td>
<td>GlideRecord</td>
<td>A roster_schedule_span_proposal record.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A formatted GlideScheduleDateTime string based on the start date of the rosterSpanProposalGr.</td>
</tr>
</tbody>
</table>

OCRosterSpanApprovalUtil - getPTOApproversList (GlideRecord rosterSpanProposalGr)

Returns an array of group managers based on the user's rota groups.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rosterSpanProposalGr</td>
<td>GlideRecord</td>
<td>A roster_schedule_span_proposal record.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of group managers based on the user's rota groups.</td>
</tr>
</tbody>
</table>

OCRosterSpanApprovalUtil - getToDateDisplayValuePerSpanProposal (rosterSpanProposalGr GlideRecord)

Returns a formatted GlideScheduleDateTime string based on the end date of the rosterSpanProposalGr.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rosterSpanProposalGr</td>
<td>GlideRecord</td>
<td>A roster_schedule_span_proposal record.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A formatted GlideScheduleDateTime string based on the end date of the rosterSpanProposalGr.</td>
</tr>
</tbody>
</table>

#### OCRosterSpanApprovalUtil - getUserNamePerSpanProposal (GlideRecord rosterSpanProposalGr)

Returns the user's name based on rosterSpanProposalGr.

| Parameters
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rosterSpanProposalGr</td>
<td>GlideRecord</td>
<td>A roster_schedule_span_proposal record.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The user's name based on rosterSpanProposalGr.</td>
</tr>
</tbody>
</table>

#### OCRosterSpanApprovalUtil - isPTOApprovalRequired ()

Checks whether the system property `com.snc.on_call_rotation.pto.approval.required` is true.

| Parameters
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the system property \texttt{com.snc.on_call_rotation.pto.approval.required} is true.</td>
</tr>
</tbody>
</table>

\textbf{OCRosterSpanApprovalUtil - rejectPTOSpan (rosterSpanGr GlideRecord)}

Changes the type of the roster\_schedule\_span to rejected from approval.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rosterSpanGr</td>
<td>GlideRecord</td>
<td>A roster_schedule_span_proposal record.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

\textbf{OCRotation - Global}

The core \texttt{ORotation} builds a data structure that is used to display the calendar.

The code builds up the GlideAJAXSchedulePage object which stores a list of span items. You can use this type for the Gwt (legacy) calendar. See the \texttt{OCRotationV2} API for other calendar types. This is part of On-Call Scheduling.

\textbf{Related information}

\texttt{OCRotationV2 - Global}

\texttt{OCRotation - buildRotas()}

Builds the on-call coverage based on your groups, rotas, and rosters. This is used by the \texttt{OCRotationV2 - getSpans} method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Void</td>
<td></td>
</tr>
</tbody>
</table>

Related reference

**OCRotationV2 - getSpans()**

**OCRotation - getEndDate()**

Returns the end date for the time period for which you want to retrieve on-call coverage.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>End date for the time period for which you want to retrieve on-call coverage.</td>
</tr>
</tbody>
</table>

**OCRotation - getGroupIds()**

Returns a comma separated list of group sys_id (sys_user_group) values for an on-call schedule.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Comma separated list of group sys_id (sys_user_group) values for an on-call schedule.</td>
</tr>
</tbody>
</table>
**OCRotation - getRosterIds()**

Returns the rosters according to the selected rotas.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Rosters according to the selected rotas.</td>
</tr>
</tbody>
</table>

**OCRotation - getRotaGr(String rotaIds, String groupIds, String rosterIds, String userIds)**

Returns a GlideRecord for the cmn_rota table filtered by groups, rota, users, and roster.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaIds</td>
<td>String</td>
<td>Comma separated list of rota sys_id (cmn_rota) values for an on-call schedule.</td>
</tr>
<tr>
<td>groupIds</td>
<td>String</td>
<td>Comma separated list of group sys_id (sys_user_group) values for an on-call schedule.</td>
</tr>
<tr>
<td>rosterIds</td>
<td>String</td>
<td>Comma separated list of roster sys_id (cmn_rota_roster) values for an on-call schedule.</td>
</tr>
<tr>
<td>userIds</td>
<td>String</td>
<td>Comma separated list of user sys_id (sys_user) values for an on-call schedule.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>A GlideRecord for the cmn_rota table filtered by groups, rota, users, and roster.</td>
</tr>
</tbody>
</table>
**OCRotation - getRotaIds()**
Returns the rotas for the groups on your calendar.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Rotas for the groups on your calendar.</td>
</tr>
</tbody>
</table>

**OCRotation - getStartDate()**
Returns the start date for the time period for which you want to retrieve on-call coverage.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Start date for the time period for which you want to retrieve on-call coverage.</td>
</tr>
</tbody>
</table>

**OCRotation - getTimezone()**
Returns the timezone that your on-call calendar will use.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Timezone that your on-call calendar will use. For example, <strong>Europe/Madrid</strong> and <strong>US/Pacific</strong>.</td>
</tr>
</tbody>
</table>

**OCRotation - getUserIDs()**

This will filter the schedules return by users. Get comma separated list of user `sys_id` (sys_user) values used to filter on-call schedules.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Comma separated list of user <code>sys_id</code> (sys_user) values used to filter on-call schedules.</td>
</tr>
</tbody>
</table>

**OCRotation - setEndDate(String endDate, Boolean inclusive)**

Set the end date for the time period for which you want to retrieve on-call coverage. If the end date is not set, then the last day of the next month is applied by default. Use in conjunction with **OCRotationV2 - getSpans**.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td><code>endDate</code></td>
</tr>
<tr>
<td><code>inclusive</code></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Void</td>
<td></td>
</tr>
</tbody>
</table>
Related reference

OCRotationV2 - getSpans()

OCRotation - setGroupIds(String groupIds)
Filters the schedules return by groups. Set comma separated list of group sys_id (sys_user_group) values for an on-call schedule. Use in conjunction with OCRotationV2 - getSpans.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupIds</td>
<td>String</td>
<td>Comma separated list of group sys_id (sys_user_group) values for an on-call schedule.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Void</td>
<td></td>
</tr>
</tbody>
</table>

Related reference

OCRotationV2 - getSpans()

OCRotation - setRosterIds(String rosterIds)
Sets the rosters according to the selected rotas. Use in conjunction with OCRotationV2 - getSpans.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RosterIds</td>
<td>String</td>
<td>Rosters according to the selected rotas.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
**Related reference**

OCRotationV2 - getSpans()

**OCRotation - setRotaIds(String rotaIds)**
Sets the rotas for the groups on your calendar. Use in conjunction with OCRotationV2 - getSpans.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>RotaIds</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Void</td>
</tr>
</tbody>
</table>

**Related reference**

OCRotationV2 - getSpans()

**OCRotation - setStartDate(String startDate)**
Sets the start date for the time period for which you want to retrieve on-call coverage. If the start date is not set, then the first day of the previous month is applied by default. Use in conjunction with OCRotationV2 - getSpans.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>startDate</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>
Related reference

OCRotationV2 - getSpans()

OCRotation - setTimezone(String timezone)
Sets the timezone that your on-call calendar will use. Use in conjunction with OCRotationV2 - getSpans.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timezone</td>
<td>String</td>
<td>Timezone that your on-call calendar will use. For example, Europe/Madrid and US/Pacific.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Void</td>
<td></td>
</tr>
</tbody>
</table>

Related reference

OCRotationV2 - getSpans()

OCRotation - setUserIds(String userIds)
Sets comma separated list of user sys_id (sys_user) values used to filter on-call schedules. Use in conjunction with OCRotationV2 - getSpans.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserIds</td>
<td>String</td>
<td>Comma separated list of user sys_id (sys_user) values used to filter on-call schedules.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Void</td>
<td></td>
</tr>
</tbody>
</table>
Related reference

**OCRotationV2 - getSpans()**

---

**OCRotationV2 - Global**

**OCRotationV2** is an extension of **OCRotation**, which is used by Fullcalendar.io and the DHTMLX On-Call Calendar.

**OCRotationV2** is used to get the schedule time span between two dates that are further filtered by groups, rotas, rosters, and users. You can also extend **OCRotationV2** to add or modify behavior. This is part of On-Call Scheduling.

---

**OCRotationV2 - getGroups()**

Returns an array of groups that have active rotas.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of group objects. Each contains the following attributes:</td>
</tr>
<tr>
<td></td>
<td>• sys_id of the sys_user_group record</td>
</tr>
<tr>
<td></td>
<td>• name of the sys_user_group record</td>
</tr>
</tbody>
</table>

---

**OCRotationV2 - getRostersByRotas(String rotaSysIds)**

Returns an array of active rosters for a given rota ID.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaSysIds</td>
<td>String</td>
<td>Comma separated list of rota sys_id (cmn_rota) values.</td>
</tr>
</tbody>
</table>
returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of rota objects. Each contains the following attributes:</td>
</tr>
<tr>
<td></td>
<td>• sys_id of the cmn_rota_roster record</td>
</tr>
<tr>
<td></td>
<td>• name of the cmn_rota_roster record</td>
</tr>
<tr>
<td></td>
<td>• rota sys_id of the cmn_rota record</td>
</tr>
</tbody>
</table>

**OCRotationV2 - getRotasByGroup(String groupSysIds)**

Returns an array of active rotas for a given group ID.

**Params**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupSysIds</td>
<td>String</td>
<td>Comma separated list of group sys_id (sys_user_group) values.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of rota objects. Each contains the following attributes:</td>
</tr>
<tr>
<td></td>
<td>• sys_id of the cmn_rota record</td>
</tr>
<tr>
<td></td>
<td>• name of the cmn_rota record</td>
</tr>
<tr>
<td></td>
<td>• group sys_id of the sys_user_group record</td>
</tr>
</tbody>
</table>

**OCRotationV2 - getSpans()**

Get the spans from the specified start date to the specified end date. If no start and end dates are provided, the start date defaults to a month before and the end date defaults to a month after the current time. You can also use the groups IDs, rota IDs, roster IDs, user IDs to further filter the spans.

**Params**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of spans. Each contains the following attributes: id, sys_id,</td>
</tr>
<tr>
<td></td>
<td>table, rota_id, roster_id, user_id, text, description, color, textColor,</td>
</tr>
<tr>
<td></td>
<td>start_date, end_date.</td>
</tr>
</tbody>
</table>

Get all spans for the default time period

```javascript
var spans = new OCRotationV2().getSpans();
var firstSpanStartDate = spans[0].start_date; // get the first span's start date
```

Get all spans between 1st April 2014 and 5th June 2014

```javascript
var spans = new OCRotationV2()
  .setStartDate("2014-04-01")
  .setEndDate("2014-06-05")
  .getSpans();
```

Get the Network group's spans for the default time period

```javascript
var spans = new OCRotationV2()
  .setGroupIds("287ebd7da9fe198100f92cc8d1d2154e")
  .getSpans();
```

Get ITIL User's spans between 1st January 2014 and 31st January 2014

```javascript
var spans = new OCRotationV2()
  .setStartDate("2014-01-01")
  .setEndDate("2014-01-31")
  .setUserIds("681b365ec0a80164000fb0b05854a0cd")
  .getSpans();
```

**OCRotaICalendarSNC - Global**

The OCRotaICalendarSNC API provides iCal formatted events for the a specific user's rotation.

**OCRotaICalendarSNC - cleanExpiredCache ()**

Records in the cmn_rota_resp_cache table that have a From date prior to today.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

OCRotaICalendarSNC - createCalendarEvents (String groupId, String rotaId, String userId, Object dateRangeObj)

Returns list of AJAXScheduleItems for the user's on-call.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupId</td>
<td>String</td>
<td>The group sys id.</td>
</tr>
<tr>
<td>rotaId</td>
<td>String</td>
<td>The rota sys id.</td>
</tr>
<tr>
<td>userId</td>
<td>String</td>
<td>The user sys id.</td>
</tr>
<tr>
<td>dateRangeObj</td>
<td>Object</td>
<td>Contains the from and to dates for the data set.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AJAXScheduleItem</td>
<td>Java list of AJAXScheduleItems.</td>
</tr>
</tbody>
</table>

OCRotaICalendarSNC - createCustomEvent (String groupName, Object scheduleItemSpan, String calendarLink)

Creates a VEVENT based on the scheduleItemSpan provided.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupName</td>
<td>String</td>
<td>Name of the on-call group.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheduleItemSpan</td>
<td>Object</td>
<td>Span of time for the on-call period.</td>
</tr>
<tr>
<td>calendarLink</td>
<td>String</td>
<td>A link back to the on-call calendar in the instance.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A VEVENT based on the scheduleItemSpan provided.</td>
</tr>
</tbody>
</table>

OCRotaICalendarSNC - createPlaceholderCalendar (GlideRecord rotaGR, Object dateRangeObj, String calendarLink)

Creates a formatted iCalendar if the user has no upcoming on-call.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaGR</td>
<td>GlideRecord</td>
<td>The cmn_rota glide record.</td>
</tr>
<tr>
<td>dateRangeObj</td>
<td>Object</td>
<td>Contains the from and to dates for the data set.</td>
</tr>
<tr>
<td>calendarLink</td>
<td>String</td>
<td>A link to the on-call calendar for this user.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Formatted iCalendar with a placeholder VEVENT.</td>
</tr>
</tbody>
</table>

OCRotaICalendarSNC - getCalendarEvents (String groupId, String rotaId, String userId, Object dateRangeObj, Boolean useCache)

Returns the formatted iCalendar for the user's rotation.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupId</td>
<td>String</td>
<td>The group sys id.</td>
</tr>
<tr>
<td>rotaId</td>
<td>String</td>
<td>The rota sys id.</td>
</tr>
<tr>
<td>userId</td>
<td>String</td>
<td>The user sys id.</td>
</tr>
<tr>
<td>dateRangeObj</td>
<td>Object</td>
<td>Contains the from and to dates for the data set.</td>
</tr>
<tr>
<td>useCache</td>
<td>Boolean</td>
<td>If true, makes use of the cache table cmn_rota_resp_cache.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Formatted iCalendar for the user's rotation.</td>
</tr>
</tbody>
</table>

**OCRotaICalendarSNC - getEventsFromTable (String groupId, String rotaId, String userId, Object dateRangeObj)**

Returns the formatted iCalendar for the user's rotation if found in the cmn_rota_resp_cache table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupId</td>
<td>String</td>
<td>The group sys id.</td>
</tr>
<tr>
<td>rotaId</td>
<td>String</td>
<td>The rota sys id.</td>
</tr>
<tr>
<td>userId</td>
<td>String</td>
<td>The user sys id.</td>
</tr>
<tr>
<td>dateRangeObj</td>
<td>Object</td>
<td>Contains the from and to dates for the data set.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Formatted iCalendar for the user's rotation if found in the cmn_rota_resp_cache table.</td>
</tr>
</tbody>
</table>
OCRotaCalendarSNC - getIntersectRotaSpanItem (ScheduleTimeSpan timeSpan, Array rotaSpanItems)

Checks whether the provided timeSpan intersects with one of the rota span items.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeSpan</td>
<td>ScheduleTimeSpan</td>
<td>A single span for the user’s on-call.</td>
</tr>
<tr>
<td>rotaSpanItems</td>
<td>Array</td>
<td>Contains the user’s standard on-call spans.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>If the timeSpan intersects with one of the rota span items, returns an object containing the rota span item and the span that it intersects with.</td>
</tr>
</tbody>
</table>

OCRotaCalendarSNC - getMemberCalendarURL (String groupId, Object rotaId, String userid)

Creates the subscribable URL for the user’s iCalendar.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupId</td>
<td>String</td>
<td>The group sys id.</td>
</tr>
<tr>
<td>rotaId</td>
<td>Object</td>
<td>The rota sys id.</td>
</tr>
<tr>
<td>userid</td>
<td>String</td>
<td>The user sys id.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The subscribable URL for the user’s iCalendar.</td>
</tr>
</tbody>
</table>

OCRotaCalendarSNC - getOnCallCalendarURL (GlideRecord rotaGR)

Returns a URL to the on-call calendar based on the rota record.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaGR</td>
<td>GlideRecord</td>
<td>A cmn_rota record.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>URL to the on-call calendar based on the rota record.</td>
</tr>
</tbody>
</table>

**OCRotaICalendarSNC - handleOverrideMember (AJAXScheduleItem scheduleItem)**

Returns an array of the user's overrides (extra coverage).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheduleItem</td>
<td>AJAXScheduleItem</td>
<td>A list of AJAXScheduleItems.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of the user's overrides (extra coverage).</td>
</tr>
</tbody>
</table>

**OCRotaICalendarSNC - handleRotaMember (AjaxScheduleItem scheduleItem, Object rotaSpanItems, Object definitionItems, Array repeatRotaSpanIdArr)**

Returns an object containing the user's override (extra coverage) and excluded spans.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheduleItem</td>
<td>AJAXScheduleItem</td>
<td>A list of AJAXScheduleItems.</td>
</tr>
<tr>
<td>rotaSpanItems</td>
<td>Object</td>
<td>Contains the user's standard on-call spans.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>definitionItems</td>
<td>Object</td>
<td>Contains the rota's standard on-call spans.</td>
</tr>
<tr>
<td>repeatRotaSpanIdArr</td>
<td>Array</td>
<td>An array of rota span sys ids.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Contains the user's override (extra coverage) and excluded spans.</td>
</tr>
</tbody>
</table>

**OCRotaICalendarSNC - invalRotaRespCache** (GlideRecord rotaRespCacheGR)

Used by business rules on the cmn_rota_member, cmn_schedule_span, and roster_schedule_span tables to update the cmn_rota Resp_cache table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaRespCacheGR</td>
<td>GlideRecord</td>
<td>Record that has been updated.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**OCRotaICalendarSNC - matchRotaSpanRule** (ScheduleTimeSpan timeSpan, Array rotaSpanItems, Array repeatRotaSpanIdArr, Object seriesStartTimes)

Checks whether the provided timeSpan matches one of the rotaSpanItems.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeSpan</td>
<td>ScheduleTimeSpan</td>
<td>A single span for the user's on-call.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaSpanItems</td>
<td>Array</td>
<td>Contains the user's standard on-call spans.</td>
</tr>
<tr>
<td>repeatRotaSpanIdArr</td>
<td>Array</td>
<td>An array of rota span sys ids.</td>
</tr>
<tr>
<td>seriesStartTimes</td>
<td>Object</td>
<td>Start time of the rota spans.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the timeSpan matches one of the rotaSpanItems.</td>
</tr>
</tbody>
</table>

**OCRotaCalendarSNC - populateCalendarSubscriptionSettings ()**

Used by fixed job to populate the calendar subscription fields on the cmn_rota form

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**OCRotaCalendarSNC - processSeriesEvent (List scheduleItems, Array repeatRotaSpanIdArr, String userId)**

Returns an object containing the user's rotations.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheduleItems</td>
<td>List</td>
<td>The user's on-call rotation as an AJAXScheduleItem list.</td>
</tr>
<tr>
<td>repeatRotaSpanIdArr</td>
<td>Array</td>
<td>An array of the cmn_rota schedule span record sys_ids.</td>
</tr>
<tr>
<td>userId</td>
<td>String</td>
<td>The user sys id.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Contains the user's rotations.</td>
</tr>
</tbody>
</table>

**OCRotaICalendarSNC - saveCalendarEvents (String groupId, String rotaId, String userId, Object dateRangeObj, String result)**

Inserts the user's iCalendar into the cmn_rota_resp_cache table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupId</td>
<td>String</td>
<td>The group sys id.</td>
</tr>
<tr>
<td>rotaId</td>
<td>String</td>
<td>The rota sys id.</td>
</tr>
<tr>
<td>userId</td>
<td>String</td>
<td>The user sys id.</td>
</tr>
<tr>
<td>dateRangeObj</td>
<td>Object</td>
<td>Contains the from and to dates for the data set.</td>
</tr>
<tr>
<td>result</td>
<td>String</td>
<td>The user's iCalendar.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**OCRotaICalendarSNC - sendCalendarURL (GlideRecord rotaGR)**

Queues events to send an email to all members of the rotation.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaGR</td>
<td>GlideRecord</td>
<td>A cmn_rota record.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

OCRotaICalendarSNC - updateExceptionList (AJAXScheduleItem scheduleItem, ScheduleTimeSpan timeSpan, Array rotaSpanItems)

Returns an object that contains all of the timeSpans that need to be excluded from the user's iCalendar.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheduleItem</td>
<td>AJAXScheduleItem</td>
<td>One AJAXScheduleItem.</td>
</tr>
<tr>
<td>timeSpan</td>
<td>ScheduleTimeSpan</td>
<td>A single user's standard on-call span.</td>
</tr>
<tr>
<td>rotaSpanItems</td>
<td>Array</td>
<td>Contains the user's standard on-call spans.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object that contains all of the timeSpans that need to be excluded from the user's iCalendar.</td>
</tr>
</tbody>
</table>

### OCRotaICalendar - Global

The OCRotaICalendar API is an extension of the OCRotaICalendarSNC API.

### OCRotaMember - Global

The OCRotaMember API performs maintenance operations on the cmn_rota_member table.
**OCRotaMember - deactivateUser (String userId, GlideDateTime deactivateDate)**

Deactivates the user's rota member records according to the deactivate date.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userId</td>
<td>String</td>
<td>The sys id of the user.</td>
</tr>
<tr>
<td>deactivateDate</td>
<td>GlideDateTime</td>
<td>The date from which the user's rotation is deactivated.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**OCRotaMember - hasChanged (GlideRecord current, GlideRecord previous)**

Checks whether the cmn_rota_member record has changed. Checks the Member, From, To, and Order fields have changed.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>current</td>
<td>GlideRecord</td>
<td>The latest changes made to the cmn_rota_member record.</td>
</tr>
<tr>
<td>previous</td>
<td>GlideRecord</td>
<td>The original state of the cmn_rota_member record.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**OCRotaMember - hasOrderChanged (GlideRecord current, GlideRecord previous)**

On update of a cmn_rota_member record, checks whether the order of the record has changed.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>current</td>
<td>GlideRecord</td>
<td>The latest changes made to the cmn_rota_member record.</td>
</tr>
<tr>
<td>previous</td>
<td>GlideRecord</td>
<td>The original state of the cmn_rota_member record.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the order has changed.</td>
</tr>
</tbody>
</table>

### OCRotaMember - recalculate (GlideRecord current, GlideRecord previous)

Recalculates the schedule for the cmn_rota_member record.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>current</td>
<td>GlideRecord</td>
<td>The latest changes made to the cmn_rota_member record.</td>
</tr>
<tr>
<td>previous</td>
<td>GlideRecord</td>
<td>The original state of the cmn_rota_member record.</td>
</tr>
</tbody>
</table>

### Returns

| Type       | Description
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### OCRotaMember - validateDates (String from, String to)

Ensures that the From date occurs before or on the same date as the To date.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>from</td>
<td>String</td>
<td>Formatted date/</td>
</tr>
<tr>
<td>to</td>
<td>String</td>
<td>Formatted date/</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the From date is before or on same date as To date.</td>
</tr>
</tbody>
</table>

**OnCallContactPreferenceUtil - Global**

Utility for handling contact preferences.

For contact preference configuration, see [Configure contact preferences](#).

This utility API supports the [OnCallRotation API](#).

**OnCallContactPreferenceUtil - OnCallContactPreferenceUtil()**

Creates an instance of the OnCallContactPreferenceUtil class.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**OnCallContactPreferenceUtil - getContactsAtByEscalatee(Object escalatee, Number contactAttempt, GlideDateTime gdt, String rotaSysIds)**

Gets a list of contacts with the escalatee object. Each returned contact contains a list of user contact preferences.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>escalatee</td>
<td>Object</td>
<td>Mandatory. Escalatee object returned by the getEscalationPlan() method.</td>
</tr>
<tr>
<td>contactAttempt</td>
<td>Number</td>
<td>Optional. The attempt you are trying to contact [ default value = 1 ].</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1: First attempt</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2: First reminder</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 3 or more: Subsequent reminders</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gdt</td>
<td>GlideDateTime</td>
<td>Optional. yyyy-MM-dd HH:mm:ss format in the UTC timezone. Default is now.admin</td>
</tr>
<tr>
<td>rotaSysIds</td>
<td>String</td>
<td>Optional. Comma-separated string of Sys IDs of shifts from the [cmn_rota] table. Default value returns Sys IDs of the shifts at current time.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array object</td>
<td>Array of escalatees (either a user, group, or device records depending on the notification rules) at a specific time and position.</td>
</tr>
<tr>
<td>type: String</td>
<td>Type of contact, e.g., user.</td>
</tr>
<tr>
<td>sys_id: String</td>
<td>Sys ID of the user from the [sys_user] table.</td>
</tr>
<tr>
<td>rota_id: String</td>
<td>Sys ID of the shift from the [cmn_rota] table.</td>
</tr>
<tr>
<td>roster_id: String</td>
<td>Sys ID of the roster from the [cmn_rota_roster] table.</td>
</tr>
<tr>
<td>escalation_step_id: String</td>
<td>If an escalation path has been defined, Sys ID of the escalation step from the [cmn_rota_esc_step_def] table.</td>
</tr>
<tr>
<td>forced_communication_channel: String</td>
<td>String. If specified in the preferences, mandatory communication channel. Supported values include SMS, voice or email.</td>
</tr>
<tr>
<td>override_user_contact_preference: Boolean</td>
<td>Boolean. True if there are contact overrides, false otherwise.</td>
</tr>
<tr>
<td>preference_source: String</td>
<td>String. Source of preference, such as contact or group.</td>
</tr>
<tr>
<td>contact_attempt: Number</td>
<td>Contact attempts: 1 = first attempt, 2 = first reminder, and so on.</td>
</tr>
<tr>
<td>user_preferences: Array Object</td>
<td>Array Object.</td>
</tr>
</tbody>
</table>
**Returns (continued)**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>◦</td>
<td>type: String. User’s preferred contact format, e.g. sms, email, or number.</td>
</tr>
<tr>
<td>◦</td>
<td>&lt;name of preferred format, e.g., email or number&gt;: String. Content of preferred contact type, e.g., email address or phone number.</td>
</tr>
</tbody>
</table>

• contact_preferences, Array Object.
  ◦ type: String. User’s preferred contact format, e.g., email or number.
  ◦ <name of preferred format, e.g., email or number>: String. Content of preferred contact type, e.g., email address or phone number.

**Example:**

```javascript
var serviceDeskSysId = '<Sys_ID>';  
var ocRotation = new OnCallRotation();  
var escalationPlan = ocRotation.getEscalationPlan(serviceDeskSysId);  
var escalatee = escalationPlan[0];  
  
var onContactPrefUtil = new OnCallContactPreferenceUtil();  
var contactAttempt = 1;  
var gdt = new GlideDateTime();  
var contacts = onContactPrefUtil.getContactsAtByEscalatee(escalatee, contactAttempt, gdt);  
gs.log(JSON.stringify(contacts));
```

**Output:**

```json
[
  {
    "type": "user",
    "sys_id": "<Sys_ID>",
    "rota_id": "<Rotation_ID>",
    "roster_id": "",
    "escalation_step_id": "<Escalation_Step_ID>",
    "forced_communication_channel": "",
    "override_user_contact_preference": true,
    ".preference_source": "contact",
    "contact_attempt": 1,
    "user_preferences": [
```
OnCallContactPreferenceUtil - getUserPreferences(String userSysId, Number contactAttempt, GlideDateTime gdt, String communicationTypes)

Gets the list of contact preferences at the time at which the user has opted for communication. If no time is specified, then current time is used. This method is applicable to single and overlapping shifts.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userSysId</td>
<td>String</td>
<td>Sys ID of the user from the [sys_user] table.</td>
</tr>
<tr>
<td>contactAttempt</td>
<td>Number</td>
<td>Optional. The attempt you are trying to contact [ default value = 1 ].</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1: First attempt</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2: First reminder</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 3 or more: Subsequent reminders</td>
</tr>
<tr>
<td>gdt</td>
<td>GlideDateTime</td>
<td>Optional. yyyy-MM-dd HH:mm:ss format in the UTC timezone. Default is now.</td>
</tr>
<tr>
<td>communicationTypes</td>
<td>String</td>
<td>Optional. Comma-separated string containing one or more of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• email</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• sms</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• voice</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default includes all available communication types.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Array Object</td>
<td>On-call user contact preference.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• type: String. User's preferred contact format, e.g. sms, email, or number.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• &lt;name of preferred format, e.g., email or number&gt;: String. Content of</td>
<td></td>
</tr>
<tr>
<td></td>
<td>preferred contact type, e.g., email address or phone number.</td>
<td></td>
</tr>
</tbody>
</table>

Example:

```javascript
var onContactPrefUtil = new OnCallContactPreferenceUtil();
var contactAttempt = 1;
var gdt = new GlideDateTime();
var itilUserSysId = '<itilUserSysId>;';
var userPreferences = onContactPrefUtil.getUserPreferences(itilUserSysId, contactAttempt, gdt);
gs.log(JSON.stringify(userPreferences));
```

Output:

```json
[{  
    "type": "sms",
    "number": "5555555555"
}, {
    "type": "voice",
    "number": "5555555555"
}]
```

**OnCallContactPreferenceUtil - getCatchAllContacts(String rotaId, GlideDateTime gdt)**

Gets a list of catch-all contacts and preferences.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaId</td>
<td>String</td>
<td>Sys ID of the shift from the [cmn_rota] table.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gdt</td>
<td>GlideDateTime</td>
<td>Optional. yyyy-MM-dd HH:mm:ss format in the UTC timezone. Default is now.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array object</td>
<td>List of catch-all contacts with preferences.</td>
</tr>
<tr>
<td></td>
<td>• type: String. Type of contact, e.g., user.</td>
</tr>
<tr>
<td></td>
<td>• sys_id: String. Sys ID of the user from the [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>• rota_id: String. Sys ID of the shift from the [cmn_rota] table.</td>
</tr>
<tr>
<td></td>
<td>• roster_id: String. Sys ID of the roster from the [cmn_rota_roster] table.</td>
</tr>
<tr>
<td></td>
<td>• escalation_step_id: String. If an escalation path has been defined, Sys ID of the escalation step from the [cmn_rota_esc_step_def] table.</td>
</tr>
<tr>
<td></td>
<td>• forced_communication_channel: String. If specified in the preferences, mandatory communication channel. Supported values include SMS, voice or email.</td>
</tr>
<tr>
<td></td>
<td>• override_user_contact_preference: Boolean. True if there are contact overrides, false otherwise.</td>
</tr>
<tr>
<td></td>
<td>• preference_source: String. Source of preference, such as contact or group.</td>
</tr>
<tr>
<td></td>
<td>• contact_attempt: Number. Contact attempts: 1 = first attempt, 2 = first reminder, and so on.</td>
</tr>
<tr>
<td></td>
<td>• user_preferences: Array Object.</td>
</tr>
<tr>
<td></td>
<td>• type: String. User's preferred contact format, e.g. sms, email, or number.</td>
</tr>
<tr>
<td></td>
<td>• &lt;name of preferred format, e.g., email or number&gt;: String. Content of preferred contact type, e.g., email address or phone number.</td>
</tr>
<tr>
<td></td>
<td>• contact_preferences, Array Object.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>◦ type: String. User's preferred contact format, e.g., email or number.</td>
<td></td>
</tr>
<tr>
<td>◦ &lt;name of preferred format, e.g., email or number&gt;: String. Content of preferred contact type, e.g., email address or phone number.</td>
<td></td>
</tr>
</tbody>
</table>

Example:

```javascript
var onContactPrefUtil = new OnCallContactPreferenceUtil();
var contactAttempt = 1;
var gdt = new GlideDateTime();
var rotaSysId = '<Rotation_Sys_ID>'; 
var catchAllContacts = onContactPrefUtil.getCatchAllContacts(rotaSysId, gdt);
gs.log(JSON.stringify(catchAllContacts));
```

Output:

```json
[
  {
    "type": "user",
    "sys_id": "<Sys_ID>",
    "rota_id": "<Rotation_ID>",
    "roster_id": "",
    "escalation_step_id": "<Escalation_Step_ID>",
    "forced_communication_channel": "",
    "override_user_contact_preference": true,
    "preference_source": "contact",
    "contact_attempt": 1,
    "user_preferences": [
      {
        "type": "sms",
        "number": "8587200477"
      }
    ],
    "contact_preferences": [
      {
        "type": "voice",
        "number": "5555555555"
      }
    ]
  }
]
```

OnCallRotation - Global

Use to manage on-call schedules.
For information, see On-call scheduling

**OnCallRotation - getCatchAll(String rotaID)**

Gets the sys_id of the catch all person. Use together with the `getCatchAllType()` method to determine the source of the returned id. This API is applicable for a single shift, overlapping shifts, and custom escalation.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaID</td>
<td>String</td>
<td>The sys_id of the rota.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_id of a sys_user, a cmn_rota_roster, or null.</td>
</tr>
</tbody>
</table>

```javascript
var rotaSysId = '349f399173062300e7af6238edf6a75e';
var onCallRotation = new OnCallRotation();
var catchAll = onCallRotation.getCatchAll(rotaSysId);
gs.log(catchAll);
```

**Output:**

46d44a23a9fe19810012d100cca80666

**OnCallRotation - getCatchAllType(String rotaID)**

Gets the type of _catch all_ that lives at the end of the escalation chain. When the escalation chain finishes without assigning an on-call person, if configured, the incident is assigned to the _catch all_. This API is applicable for a single shift, overlapping shifts, and custom escalation.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaID</td>
<td>String</td>
<td>The sys_id of the rota.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String | • Null: invalid rota id or catch all is not available.  
        | • Group_manager: the manager of the group linked to the rota is the catch all person.  
        | • Individual: a configured user is the catch all person.  
        | • All: all members of the configured roster are the catch all person. |

```javascript
var rotaSysId = '349f399173062300e7af6238edf6a75e';
var onCallRotation = new OnCallRotation();
var catchAllType = onCallRotation.getCatchAllType(rotaSysId);
gs.log(catchAllType);
```

Output:

`group_manager`

---

**OnCallRotation - getEscalatees(String groupSysIds, String rotaSysIds, String rosterSysIds, GlideDateTime gdt, GlideRecord taskGr)**

Gets a list of escalatees that are on-call for a given time. If no time is specified, then the current time is used. If no groupSysIds are provided, empty list is returned. This API is applicable for a single shift, overlapping shifts, and custom escalation.

**Note:** Set the taskGR parameter to enable escalation plan evaluation.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupSysIds</td>
<td>String</td>
<td>Comma-separated list of sys_ids of the sys_user_group, to filter the records by.</td>
</tr>
<tr>
<td>rotaSysIds</td>
<td>String</td>
<td>Comma-separated list of cmn_rota sys_ids to filter the records by.</td>
</tr>
<tr>
<td>rosterSysIds</td>
<td>String</td>
<td>Comma-separated list of cmn_rota_roster sys_ids to filter the records by.</td>
</tr>
<tr>
<td>gdt</td>
<td>GlideDate</td>
<td>Date in the UTC timezone</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>taskGr</td>
<td>GlideRecord</td>
<td>Optional. Current task record. If taskGr is empty, this method returns the default escalation plan. If taskGr is passed, escalation policies are evaluated in order and this method returns the corresponding escalation plan.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Escalatee information such as userLds, deviceLds, and escalationGroups.</td>
</tr>
</tbody>
</table>

```javascript
var groupSysId = '096fb59173062300e7af6238edf6a783';
var currentTime = new GlideDateTime();
var onCallRotation = new OnCallRotation();
var result = onCallRotation.getEscalatees(groupSysId, '', '', currentTime);
gs.log(JSON.stringify(result));
```

Output:

For single shift:

```json
[
{
  "memberId": "119f399173062300e7af6238edf6a7e7",
  "memberIds": [],
  "userId": "62826bf03710200044e0bfc8bcbe5df1",
  "userIds": [],
  "roster": "9d9f399173062300e7af6238edf6a7e5",
  "rota": "349f399173062300e7af6238edf6a75e",
  "group": "096fb59173062300e7af6238edf6a783",
  "escalationGroups": [],
  "deviceId": ",",
  "deviceIds": [],
  "isDevice": false,
  "order": 1,
  "isOverride": false,
  "rotationScheduleId": "119f399173062300e7af6238edf6a7df",
}
```
"memberScheduleId": "9d9f399173062300e7af6238e79e6c"
],
{
"memberId": "1d9f399173062300e7af6238e79e68",
"memberIds": [],
"userId": "a8f98bb0eb32010045e1a5115206fe3a",
"userIds": [],
"roster": "9d9f399173062300e7af6238e79e65",
"rota": "349f399173062300e7af6238e79e75",
"group": "096fb59173062300e7af6238e7983",
"escalationGroups": [],
"deviceId": "",
"deviceIds": [],
"isDevice": false,
"order": 2,
"isOverride": false,
"rotationScheduleId": "1d9f399173062300e7af6238e797d",
"memberScheduleId": "d99f399173062300e7af6238e79e6f"
],
{
"memberId": "dd9f399173062300e7af6238e79e69",
"memberIds": [],
"userId": "0a826bf03710200044e0bfc8bcbe5d7a",
"userIds": [],
"roster": "9d9f399173062300e7af6238e79e65",
"rota": "349f399173062300e7af6238e79e75",
"group": "096fb59173062300e7af6238e7983",
"escalationGroups": [],
"deviceId": "",
"deviceIds": [],
"isDevice": false,
"order": 3,
"isOverride": false,
"rotationScheduleId": "1d9f399173062300e7af6238e797d",
"memberScheduleId": "d99f399173062300e7af6238e79e6f"
}
]

For overlapping shift:

[
{
"memberId": "119f399173062300e7af6238e79e79",
"memberIds": []
}
"memberId": "7daf799173062300e7af6238edf6a7a2",
"memberIds": [],
"userId": "d2826bf03710200044e0bfc8bcbe5dc9",
"userIds": [],
"roster": "b5af799173062300e7af6238edf6a7a2",
"rota": "35af799173062300e7af6238edf6a79e",
"group": "096fb59173062300e7af6238edf6a783",
"escalationGroups": [],
"deviceId": ",
"deviceIds": [],
"isDevice": false,
"order": 1,
"isOverride": false,
"rotationScheduleId": "3daf799173062300e7af6238edf6a79e",
"memberScheduleId": "06af799173062300e7af6238edf6a7cd"
},
{  "memberId": "1d9f399173062300e7af6238edf6a7e8",
"memberIds": [],
"userId": "a8f98bb0eb32010045e1a5115206fe3a",
"userIds": [],
"roster": "9d9f399173062300e7af6238edf6a7e5",
"rota": "349f399173062300e7af6238edf6a75e",
"group": "096fb59173062300e7af6238edf6a783",
"escalationGroups": [],
"deviceId": ",
"deviceIds": [],
"isDevice": false,
"order": 2,
"isOverride": false,
"rotationScheduleId": "119f399173062300e7af6238edf6a7df",
"memberScheduleId": "d99f399173062300e7af6238edf6a7ef"
},
{
"memberId": "f5af799173062300e7af6238edf6a7c9",
"memberIds": [],
"userId": "ee826bf03710200044e0bfc8bcbe5def",
"userIds": [],
"roster": "b5af799173062300e7af6238edf6a7a2",
"rota": "35af799173062300e7af6238edf6a79e",
"group": "096fb59173062300e7af6238edf6a783",
"escalationGroups": [],
"deviceId": "",
"deviceIds": [],
"isDevice": false,
"order": 2,
"isOverride": false,
"rotationScheduleId": "3daf799173062300e7af6238edf6a79e",
"memberScheduleId": "8aaf799173062300e7af6238edf6a7cf"
},
{
"memberId": "dd9f399173062300e7af6238edf6a7e9",
"memberIds": [],
"userId": "0a826bf03710200044e0bfc8bcbe5d7a",
"userIds": [],
"roster": "9d9f399173062300e7af6238edf6a7e5",
"rota": "349f399173062300e7af6238edf6a75e",
"group": "096fb59173062300e7af6238edf6a783",
"escalationGroups": [],
"deviceId": "",
"deviceIds": [],
"isDevice": false,
"order": 3,
"isOverride": false,
"rotationScheduleId": "119f399173062300e7af6238edf6a7df",
"memberScheduleId": "519f399173062300e7af6238edf6a7f2"
},
{
"memberId": "8aaf799173062300e7af6238edf6a7ca",
"memberIds": [],
"userId": "e2826bf03710200044e0bfc8bcbe5de0",
"userIds": [],
"roster": "b5af799173062300e7af6238edf6a7a2",
"rotationScheduleId": "119f399173062300e7af6238edf6a7df",
"memberScheduleId": "d99f399173062300e7af6238edf6a7ef"}
"rota": "35af799173062300e7af6238edf6a79e",
"group": "096fb59173062300e7af6238edf6a783",
"escalationGroups": [],
"deviceId": "",
"deviceIds": [],
"isDevice": false,
"order": 3,
"isOverride": false,
"rotationScheduleId": "3daf799173062300e7af6238edf6a79e",
"memberScheduleId": "02af799173062300e7af6238edf6a7d2"
}

For custom escalation

[
{
   "memberId": "",
   "memberIds": [],
   "userId": "",
   "userIds": ["62826bf03710200044e0bfc8bcbe5df1", "46d4a23a9fe19810000d100c8a0666"],
   "roster": "",
   "rota": "349f399173062300e7af6238edf6a75e",
   "group": "096fb59173062300e7af6238edf6a783",
   "escalationGroups": ["a715cd759f2002002920bde8132e7018"],
   "deviceId": "",
   "deviceIds": [],
   "isDevice": false,
   "order": 1,
   "isOverride": false,
   "rotationScheduleId": "119f399173062300e7af6238edf6a7df",
   "memberScheduleId": ""
},
{
   "memberId": "",
   "memberIds": ["119f399173062300e7af6238edf6a7e7"],
   "userId": "",
   "userIds": ["119f399173062300e7af6238edf6a7e7"],
   "roster": "",
   "rota": "349f399173062300e7af6238edf6a75e",
   "group": "096fb59173062300e7af6238edf6a783",
   "escalationGroups": [],
   "deviceId": "",
   "deviceIds": [],
   "isDevice": false,
   "order": 1,
   "isOverride": false,
   "rotationScheduleId": "119f399173062300e7af6238edf6a7df",
   "memberScheduleId": ""
}
]
{
    "userIds": [
        "62826bf03710200044e0bfc8bcbe5df1"
    ],
    "roster": "9d9f399173062300e7af6238edf6a7e5",
    "rota": "349f399173062300e7af6238edf6a75e",
    "group": "096fb59173062300e7af6238edf6a783",
    "escalationGroups": [],
    "deviceId": "",
    "deviceIds": [],
    "isDevice": false,
    "order": 2,
    "isOverride": false,
    "rotationScheduleId": "119f399173062300e7af6238edf6a7df",
    "memberScheduleId": ""
}

/**
 * create a custom escalation for service desk shift with condition as P1 for Incident
 * taskGr is optional parameter
 * if taskGr is empty then it returns list of escalatees in default escalation plan
 * if taskGr is passed then escalation policies are executed evaluated in order and list of escalatees in corresponding escalation plan is returned
 **/
var serviceDeskSysId = 'd625dccec0a8016700a222a0f7900d06';
var taskGr = new GlideRecord('incident');
taskGr.get('ed92e8d173d023002728660c4cf6a7bc');

var ocRotation = new OnCallRotation();
var escalatees = ocRotation.getEscalatees(serviceDeskSysId, null, null, null, null, taskGr);
gs.log(JSON.stringify(escalatees));
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupID</td>
<td>String</td>
<td>The sys_id for the group from which to get the item to escalate.</td>
</tr>
<tr>
<td>dateTime</td>
<td>GlideDateTime</td>
<td>The date and time when the escalation lineup should begin.</td>
</tr>
<tr>
<td>position</td>
<td>Number</td>
<td>The position in the lineup to determine the item to escalate to (1-based, which means the count starts with 1).</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The item to escalate to, that is, escalatees (either a user, group, or device records depending on the notification rules) at a specific time and position. Returns null if nothing is found.</td>
</tr>
</tbody>
</table>

```javascript
var groupSysId = '096fb59173062300e7af6238edf6a783';
var currentTime = new GlideDateTime();
var onCallRotation = new OnCallRotation();
var escalatee = onCallRotation.getEscalateeAt(groupSysId, currentTime, 1);
gs.log(escalatee.name);
```

Output:

Abel Tuter

**OnCallRotation - getEscalationPlan(String groupSysId, GlideDateTime gdt, String rotaSysIds, GlideRecord taskGr)**

Gets a list of objects to escalate to, escalation plans that consist of either a user [sys_user] or notification device [cmn_notif_device], for a group’s shift at the specified date and time. This API is applicable for a single shift, overlapping shifts, and custom escalation.

The Escalation object contains the following fields:

- **Number order**: the escalation order within the lineup
- **String userId**: sys_id of the sys_user record or null
• String deviceId: sys_id of the cmn_notif_device record or null
• Boolean isDevice: true or false depending on whether the item to escalate to is a device or user
• GlideDuration timeBetweenReminders: the time between reminders being sent
• Number reminderNum: number of reminders to be sent for each item to escalate to

⚠️ Note: Set the taskGR parameter to enable escalation plan evaluation.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupID</td>
<td>String</td>
<td>sys_id for the group for which you need the escalation plan.</td>
</tr>
<tr>
<td>gdt</td>
<td>GlideDateTime</td>
<td>Date and time when the escalation lineup should begin. Default is now. yyyy-MM-dd HH:mm:ss is the format in UTC timezone.</td>
</tr>
<tr>
<td>rotaSysIds</td>
<td>String</td>
<td>Comma-separated list of cmn_rota sys_ids to filter the records by.</td>
</tr>
<tr>
<td>taskGr</td>
<td>GlideRecord</td>
<td>Optional. Current task record. If taskGr is empty, this method returns the default escalation plan. If taskGr is passed, escalation policies are evaluated in order and this method returns the corresponding escalation plan.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>A list of escalation objects to escalate to.</td>
</tr>
</tbody>
</table>

```javascript
var groupSysId = '096fb59173062300e7af6238edf6a783';
var currentTime = new GlideDateTime();
var onCallRotation = new OnCallRotation();
var result = onCallRotation.getEscalationPlan(groupSysId, currentTime);
gs.log(JSON.stringify(result));
```

Output:
For single shift:

```json

[  
  
  {  
    "order": 1,  
    "userId": "62826bf03710200044e0bfc8bcbe5df1",  
    "userIds": [],  
    "deviceId": ",",  
    "deviceIds": [],  
    "escalationGroups": [],  
    "isDevice": false,  
    "timeToNextStep": null,  
    "cmnRotaEscStepDefId": null,  
    "reminderNum": 1,  
    "rosterId": "9d9f399173062300e7af6238edf6a7e5",  
    "memberId": "119f399173062300e7af6238edf6a7e7",  
    "memberIds": [],  
    "isOverride": false,  
    "additionalEscalatees": [],  
    "forcedCommunicationChannel": ",",  
    "rotaId": "349f399173062300e7af6238edf6a75e",  
    "groupId": "096fb59173062300e7af6238edf6a783",  
    "rotaScheduleId": "119f399173062300e7af6238edf6a7df",  
    "memberScheduleId": "9d9f399173062300e7af6238edf6a7ec"  
  },  
  
  {  
    "order": 2,  
    "userId": "a8f98bb0eb32010045e1a5115206fe3a",  
    "userIds": [],  
    "deviceId": ",",  
    "deviceIds": [],  
    "escalationGroups": [],  
    "isDevice": false,  
    "timeToNextStep": null,  
    "cmnRotaEscStepDefId": null,  
    "reminderNum": 1,  
    "rosterId": "9d9f399173062300e7af6238edf6a7e5",  
    "memberId": "1d9f399173062300e7af6238edf6a7e8",  
    "memberIds": [],  
    "isOverride": false,  
    "additionalEscalatees": [],  
    "forcedCommunicationChannel": ",",  
    "rotaId": "349f399173062300e7af6238edf6a75e",  
    "groupId": "096fb59173062300e7af6238edf6a783",  
  }

```
For overlapping shift:

[  
    {  
    "order": 1,  
    "userId": "62826bf037102000044e0bfc8bcb65df1",  
    "userIds": [],  
    "deviceId": "",  
    "deviceIds": [],  
    "escalationGroups": [],  
    "isDevice": false,  
    "timeToNextStep": null,  
    "cmnRotaEscStepDefId": null,  
    "reminderNum": 1,  
    "rosterId": "9d9f399173062300e7af6238edf6a75e",  
    "memberId": "119f399173062300e7af6238edf6a7e7",  
    "memberIds": []  
    }  
]
"isOverride": false,
"additionalEscalatees": [
{
  "order": 1,
  "userId": "d2826bf03710200044e0bfc8bcbe5dc9",
  "userIds": [],
  "deviceId": "",
  "deviceIds": [],
  "escalationGroups": [],
  "isDevice": false,
  "timeToNextStep": null,
  "cmnRotaEscStepDefId": null,
  "reminderNum": 1,
  "rosterId": "b5af799173062300e7af6238edf6a7a2",
  "memberId": "7daf799173062300e7af6238edf6a7a2",
  "memberIds": [],
  "isOverride": false,
  "additionalEscalatees": [],
  "forcedCommunicationChannel": "",
  "rotaId": "35af799173062300e7af6238edf6a79e",
  "groupId": "096fb59173062300e7af6238edf6a783",
  "rotaScheduledId": "3daf799173062300e7af6238edf6a79e",
  "memberScheduleId": "06af799173062300e7af6238edf6a7cd"
}
],
"forcedCommunicationChannel": "",
"rotaId": "349f399173062300e7af6238edf6a75e",
"groupId": "096fb59173062300e7af6238edf6a783",
"rotaScheduledId": "119f399173062300e7af6238edf6a7df",
"memberScheduleId": "9d9f399173062300e7af6238edf6a7ec"
},
{
  "order": 2,
  "userId": "a8f98bb0eb32010045e1a5115206fe3a",
  "userIds": [],
  "deviceId": "",
  "deviceIds": [],
  "escalationGroups": [],
  "isDevice": false,
  "timeToNextStep": null,
  "cmnRotaEscStepDefId": null,
  "reminderNum": 1,
  "rosterId": "9d9f399173062300e7af6238edf6a7e5",
  "memberId": "1d9f399173062300e7af6238edf6a7e8"
"memberIds": [],
"isOverride": false,
"additionalEscalatees": [
  {
    "order": 2,
    "userId": "ee826bf03710200044e0fbc8bcbe5def",
    "userIds": [],
    "deviceId": "",
    "deviceIds": [],
    "escalationGroups": [],
    "isDevice": false,
    "timeToNextStep": null,
    "cmnRotaEscStepDefId": null,
    "reminderNum": 1,
    "rosterId": "b5af799173062300e7af6238edf6a7a2",
    "memberId": "f5af799173062300e7af6238edf6a7c9",
    "memberIds": [],
    "isOverride": false,
    "additionalEscalatees": [],
    "forcedCommunicationChannel": "",
    "rotaId": "35af799173062300e7af6238edf6a79e",
    "groupId": "096fb59173062300e7af6238edf6a783",
    "rotaScheduleId": "3daf799173062300e7af6238edf6a79e",
    "memberScheduleId": "8aaf799173062300e7af6238edf6a7cf"
  },
  {
    "order": 3,
    "userId": "0a826bf03710200044e0fbc8bcbe5d7a",
    "userIds": [],
    "deviceId": "",
    "deviceIds": [],
    "escalationGroups": [],
    "isDevice": false,
    "timeToNextStep": null,
    "cmnRotaEscStepDefId": null,
    "reminderNum": 1,
    "rosterId": "b5af799173062300e7af6238edf6a7a2",
    "memberId": "f5af799173062300e7af6238edf6a7c9",
    "memberIds": [],
    "isOverride": false,
    "additionalEscalatees": [],
    "forcedCommunicationChannel": "",
    "rotaId": "35af799173062300e7af6238edf6a79e",
    "groupId": "096fb59173062300e7af6238edf6a783",
    "rotaScheduleId": "3daf799173062300e7af6238edf6a79e",
    "memberScheduleId": "8aaf799173062300e7af6238edf6a7cf"
  }]
],
"forcedCommunicationChannel": "",
"rotaId": "349f399173062300e7af6238edf6a75e",
"groupId": "096fb59173062300e7af6238edf6a783",
"rotaScheduleId": "119f399173062300e7af6238edf6a7df",
"memberScheduleId": "d99f399173062300e7af6238edf6a7ef"}
For custom escalation:

```json
[
  {
    "order": 1,
    "userId": "",  
    "userIds": [ 
      "62826bf03710200044e0bfc8bcbe5df1",
      "46d44a23a9fe19810012d100cca80666"
    ]
  }
]
```
```javascript
/**
* create a custom escalation for service desk shift with condition as P1 for Incident
* taskGr is optional parameter
* if taskGr is empty then it returns default escalation plan
* if taskGr is passed then escalation policies are executed evaluated in order and
* corresponding escalation plan is returned
**/

var serviceDeskSysId = 'd625dccec0a8016700a222a0f7900d06';
var taskGr = new GlideRecord('incident');
taskGr.get('ed92e8d173d023002728660c4cf6a7bc');
var ocRotation = new OnCallRotation();
var escalationPlan = ocRotation.getEscalationPlan(serviceDeskSysId, null, null, taskGr);
gs.log(JSON.stringify(escalationPlan));
```

**OnCallRotation - getEscalationType(String rotaID)**

Gets the type of escalation based on a simple calculation. If there is one active roster in the rota, rotate through members. If there is more than one active roster in the rota, rotate through rosters. This API is applicable for a single shift, overlapping shifts, and custom escalation.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaID</td>
<td>String</td>
<td>The sys_id for the rota to check for rosters.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>• Member: if there is one roster in the rota.</td>
</tr>
<tr>
<td></td>
<td>• Roster: if there is more than one roster in the rota.</td>
</tr>
<tr>
<td></td>
<td>• No rosters in this rota: if none of the above.</td>
</tr>
<tr>
<td></td>
<td>• Custom: If the custom escalation is applicable.</td>
</tr>
</tbody>
</table>
var rotaSysId = '349f399173062300e7af6238edf6a75e';
var onCallRotation = new OnCallRotation();
var escalationType = onCallRotation.getEscalationType(rotaSysId);
gs.log(escalationType);

Output:

custom

**OnCallRotation - OnCallRotation()**

Creates an instance of the OnCallRotation class.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

var oncall = new OnCallRotation();

**OnCallRotation - spansOverlap(ArrayList parent, ArrayList child, String timeZone)**

Checks if the parent span and child spans overlap.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent</td>
<td>ArrayList</td>
<td>The parent spans to compare.</td>
</tr>
<tr>
<td>child</td>
<td>ArrayList</td>
<td>The child spans to compare.</td>
</tr>
<tr>
<td>timeZone</td>
<td>String</td>
<td>The name of the time zone.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the parent and child spans overlap.</td>
</tr>
</tbody>
</table>

var onCall = new OnCallRotation();

// get a time range we are interested in
var timeZone = gs.getSession().getTimeZoneName();
var dateStart = new GlideDateTime();
var dateEnd = new GlideDateTime().addMonths(1);

// convert glidedatetime to scheduledatetime because it is required
// by the function being called below
var scheduleStart = new GlideScheduleDateTime(dateStart);
var scheduleEnd = new GlideScheduleDateTime(dateEnd);
scheduleStart.setTimeZone(timeZone);
scheduleEnd.setTimeZone(timeZone);

// calculate rotation items based on the date and time given
// for all the groups the currently logged in user is a member of
gs.include("OnCallRotationPersonal");
var rotation = new OnCallRotationPersonal();
var myGroups = gs.getUser().getMyGroups();
var groupIter = myGroups.iterator();

while(groupIter.hasNext()){ 
  var rotaItems = rotation.onCallDuringPeriod(groupIter.next(), scheduleStart, scheduleEnd);

  // loop through rotation schedules
  for (var i = 0; i < rotaItems.length - 1; i++) {
    var parentItem = rotaItems[i];
    var nextItem = rotaItems[i + 1];

    // check if previous and next time spans overlap
    var overlaps = onCall.spansOverlap(parentItem.getTimeSpans(), nextItem.getTimeSpans(),
    gs.getSession().getTimeZoneName());
  }
}

---

**OnCallRotation - getEscalateesAt(String groupSysId, GlideDateTime gdt, Number position, GlideRecord taskGr)**

Gets the escalatees (either a user, group, or device records depending on the notification rules) at a specific time and position. This API is applicable for a single shift, overlapping shifts and custom escalation.

⚠️ Note: Set the taskGR parameter to enable escalation plan evaluation.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupSysId</td>
<td>String</td>
<td>ys_id of the group.</td>
</tr>
<tr>
<td>gdt</td>
<td>GlideDateTime</td>
<td>yyyy-MM-dd HH:mm:ss format in the UTC timezone. Default is now.</td>
</tr>
<tr>
<td>position</td>
<td>Number</td>
<td>Position of the escalation step.</td>
</tr>
<tr>
<td>taskGr</td>
<td>GlideRecord</td>
<td>Optional. Current task record. If taskGr is empty, this method returns the default escalation plan. If taskGr is passed, escalation policies are evaluated in order and this method returns the corresponding escalation plan.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>Array of escalatees (either a user, group, or device records depending on the notification rules) at a specific time and position.</td>
</tr>
</tbody>
</table>

```javascript
var groupSysId = '096fb59173062300e7af6238edf6a783';
var currentTime = new GlideDateTime();
var onCallRotation = new OnCallRotation();
var escalatees = onCallRotation.getEscalateesAt(groupSysId, currentTime, 1);
for (var i = 0; i < escalatees.length; i++) {
gs.log(escalatees[i].name);
}
```

Output:

For overlapping shifts:

Abel Tuter  
Barbara Hindley

For custom escalation:

Abel Tuter  
Beth Anglin  
Change Management  
Barbara Hindley
// Create a custom escalation for service desk shift with condition as P1 for Incident
var serviceDeskSysId = 'd625dcce0a8016700a222a0f7900d06';
// optional parameter taskGr
var taskGr = new GlideRecord('incident');
taskGr.get('ed92e8d173d023002728660c4cf6a7bc');

var ocRotation = new OnCallRotation();
var position = 1;
var escalatees = ocRotation.getEscalateesAt(serviceDeskSysId, new GlideDateTime(),
    position, taskGr);

// if taskGr is empty then it returns escalatees at a position in default escalation plan
// if taskGr is passed then escalation policies are executed evaluated in order and
// escalatees at a position in corresponding escalation plan is returned
for (var i = 0; i < escalatees.length; i++) {
    gs.log(escalatees[i].getDisplayValue());
}

OnCallRotation - getPrimaryUser()

Gets the sys_id of the current primary contact when there is a single shift.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>sys_id of the current primary contact.</td>
</tr>
</tbody>
</table>

Example:

```javascript
var groupSysId = '096fb59173062300e7af6238edf6a783';
var currentTime = new GlideDateTime();
var onCallRotation = new OnCallRotation();
var result = onCallRotation.who(groupSysId, currentTime);
var primaryUser = onCallRotation.getPrimaryUser();
```

Output:
OnCallRotation - getPrimaryUserByRota(String rotaId)

Gets the sys_id of the current primary user of the specified shift. This API is applicable for a single shift and overlapping shifts.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaId</td>
<td>String</td>
<td>sys_id of the cmn_rota.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>sys_id of the current primary user of the specified shift.</td>
</tr>
</tbody>
</table>

```javascript
var groupSysId = '096fb59173062300e7af6238edf6a783';
var rotaId = '349f399173062300e7af6238edf6a75e';
var currentTime = new GlideDateTime();
var onCallRotation = new OnCallRotation();
var result = onCallRotation.who(groupSysId, currentTime);
var primaryUser = onCallRotation.getPrimaryUserByRota(rotaId);
gs.log(primaryUser);
```

Output:

62826bf03710200044e0bfc8bcbe5df1

OnCallRotation - getPrimaryUserName()

Gets the name of the current primary contact. This API is applicable for a single shift.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the current primary contact.</td>
</tr>
</tbody>
</table>

```javascript
var groupSysId = '096fb59173062300e7af6238edf6a783';
var currentTime = new GlideDateTime();
var onCallRotation = new OnCallRotation();
var result = onCallRotation.who(groupSysId, currentTime);
var primaryUser = onCallRotation.getPrimaryUserName();
gs.log(primaryUser);
```

Output:

Abel Tuter

---

**OnCallRotation - getPrimaryUserNameByRota(String rotaId)**

Gets the name of the current primary user of the specified shift. This API is applicable for a single shift and overlapping shifts.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaId</td>
<td>String</td>
<td>sys_id of the cmn_rota.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Name of the current primary user of the specified shift.</td>
</tr>
</tbody>
</table>

```javascript
var groupSysId = '096fb59173062300e7af6238edf6a783';
var rotaId = '349f399173062300e7af6238edf6a75e';
var currentTime = new GlideDateTime();
var onCallRotation = new OnCallRotation();
var result = onCallRotation.who(groupSysId, currentTime);
var primaryUser = onCallRotation.getPrimaryUserNameByRota(rotaId);
gs.log(primaryUser);
```

Output:
OnCallRotation - getPrimaryUsers()

Gets the list of primary contacts for a shift. This API is applicable for a single shift, overlapping shifts, and custom escalation.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>List of primary contacts.</td>
</tr>
</tbody>
</table>

```javascript
var groupSysId = '096f59173062300e7af6238edf6a783';
var currentTime = new GlideDateTime();
var onCallRotation = new OnCallRotation();
var result = onCallRotation.who(groupSysId, currentTime);
var primaryUser = onCallRotation.getPrimaryUser();
gs.log(JSON.stringify(primaryUser));
```

**Output:**

```json
{
  "349f399173062300e7af6238edf6a75e": {
    "userSysId": "62826bf03710200044e0bfc8bcbe5df1",
    "userName": "Abel Tuter"
  },
  "35af799173062300e7af6238edf6a79e": {
    "userSysId": "d2826bf03710200044e0bfc8bcbe5dc9",
    "userName": "Barbara Hindley"
  }
}
```
OnCallRotation - whoIsOnCall(String groupSysIds, String rotaSysIds, String rosterSysIds, GlideDateTime gdt)

Gets the list of users that are on-call at a given time. If no time is specified, then current time is used. This API is applicable for a single shift and overlapping shifts.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupSysIds</td>
<td>String</td>
<td>Comma-separated list of sys_ids of the sys_user_group, to filter the records by.</td>
</tr>
<tr>
<td>rotaSysIds</td>
<td>String</td>
<td>Comma-separated list of cmn_rota sys_ids to filter the records by.</td>
</tr>
<tr>
<td>rosterSysIds</td>
<td>String</td>
<td>Comma-separated list of cmn_rota_roster sys_ids to filter the records by.</td>
</tr>
<tr>
<td>gdt</td>
<td>GlideDateTime</td>
<td>Date in the UTC timezone</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>List of users, groups, or devices that are on-call at a given time. For example, information about userIds, deviceIds, and escalationGroups.</td>
</tr>
</tbody>
</table>

```javascript
var groupSysId = '096fb59173062300e7af6238edf6a783';
var currentTime = new GlideDateTime();
var onCallRotation = new OnCallRotation();
var escalatees = onCallRotation.whoIsOnCall(groupSysId, "", "", currentTime);
gs.log(JSON.stringify(escalatees));
```

**Output:**

For single shift:

```json
[
    {
        "memberId": "119f399173062300e7af6238edf6a7e7",
        "memberIds": [],
        "userId": "62826bf03710200044e0bfc8cbce5df1",
        "userIds": [],
        "roster": "9d9f399173062300e7af6238edf6a7e5",
        "rota": "349f399173062300e7af6238edf6a75e",
        "group": "096fb59173062300e7af6238edf6a783",
    }
]```
"escalationGroups": [],
"deviceId": "",
"deviceIds": [],
"isDevice": false,
"order": 1,
"isOverride": false,
"rotationScheduleId": "119f399173062300e7af6238edf6a7df",
"memberScheduleId": "9d9f399173062300e7af6238edf6a7ec"
},
{
"memberId": "1d9f399173062300e7af6238edf6a7e8",
"memberIds": [],
"userId": "a8f98bb0eb32010045e1a5115206fe3a",
"userIds": [],
"roster": "9d9f399173062300e7af6238edf6a7e5",
"rota": "349f399173062300e7af6238edf6a75e",
"group": "096fb59173062300e7af6238edf6a783",
"escalationGroups": [],
"deviceId": "",
"deviceIds": [],
"isDevice": false,
"order": 2,
"isOverride": false,
"rotationScheduleId": "119f399173062300e7af6238edf6a7df",
"memberScheduleId": "d99f399173062300e7af6238edf6a7ef"
},
{
"memberId": "dd9f399173062300e7af6238edf6a7e9",
"memberIds": [],
"userId": "0a826bf03710200044e0bfc8bcbe5d7a",
"userIds": [],
"roster": "9d9f399173062300e7af6238edf6a7e5",
"rota": "349f399173062300e7af6238edf6a75e",
"group": "096fb59173062300e7af6238edf6a783",
"escalationGroups": [],
"deviceId": "",
"deviceIds": [],
"isDevice": false,
"order": 3,
"isOverride": false,
"rotationScheduleId": "119f399173062300e7af6238edf6a7df",
"memberScheduleId": "519f399173062300e7af6238edf6a7f2"}
For overlapping shift:

```json
[
{
"memberId": "119f399173062300e7af6238edf6a7e7",
"memberIds": [],
"userId": "62826bf03710200044e0bfc8bcbe5df1",
"userIds": [],
"roster": "9d9f399173062300e7af6238edf6a7e5",
"rota": "349f399173062300e7af6238edf6a75e",
"group": "096fb59173062300e7af6238edf6a783",
"escalationGroups": [],
"deviceId": "",
"deviceIds": [],
"isDevice": false,
"order": 1,
"isOverride": false,
"rotationScheduleId": "119f399173062300e7af6238edf6a7df",
"memberScheduleId": "9d9f399173062300e7af6238edf6a7ec"
},
{
"memberId": "7daf799173062300e7af6238edf6a7a2",
"memberIds": [],
"userId": "d2826bf03710200044e0bfc8bcbe5dc9",
"userIds": [],
"roster": "b5af799173062300e7af6238edf6a7a2",
"rota": "35af799173062300e7af6238edf6a79e",
"group": "096fb59173062300e7af6238edf6a783",
"escalationGroups": [],
"deviceId": "",
"deviceIds": [],
"isDevice": false,
"order": 1,
"isOverride": false,
"rotationScheduleId": "3daf799173062300e7af6238edf6a79e",
"memberScheduleId": "06af799173062300e7af6238edf6a7cd"
},
{
"memberId": "1d9f399173062300e7af6238edf6a7e8",
"memberIds": [],
"userId": "a8f98bb0eb32010045e1a5115206fe3a",
"userIds": []
}
]
"roster": "9d9f399173062300e7af6238edf6a7e5",
"rota": "349f399173062300e7af6238edf6a75e",
"group": "096fb59173062300e7af6238edf6a783",
"escalationGroups": [],
"deviceId": "",
"deviceIds": [],
"isDevice": false,
"order": 2,
"isOverride": false,
"rotationScheduleId": "119f399173062300e7af6238edf6a7df",
"memberScheduleId": "d99f399173062300e7af6238edf6a7ef"
},
{
  "memberId": "f5af799173062300e7af6238edf6a7c9",
  "memberIds": [],
  "userId": "ee826bf03710200044e0bfc8bcb85def",
  "userIds": [],
  "roster": "b5af799173062300e7af6238edf6a7a2",
  "rota": "35af799173062300e7af6238edf6a79e",
  "group": "096fb59173062300e7af6238edf6a783",
  "escalationGroups": [],
  "deviceId": "",
  "deviceIds": [],
  "isDevice": false,
  "order": 2,
  "isOverride": false,
  "rotationScheduleId": "3daf799173062300e7af6238edf6a79e",
  "memberScheduleId": "8aaf799173062300e7af6238edf6a7cf"
},
{
  "memberId": "dd9f399173062300e7af6238edf6a7e9",
  "memberIds": [],
  "userId": "0a826bf03710200044e0bfc8bcb85d7a",
  "userIds": [],
  "roster": "9d9f399173062300e7af6238edf6a7e5",
  "rota": "349f399173062300e7af6238edf6a75e",
  "group": "096fb59173062300e7af6238edf6a783",
  "escalationGroups": [],
  "deviceId": "",
  "deviceIds": [],
  "isDevice": false,
  "order": 3,
  "isOverride": false,
  "rotationScheduleId": "119f399173062300e7af6238edf6a7df",

OnCallRotation - who(String groupSysId, GlideDateTime gdt, Boolean nullifyOverrideRoster, String rotaSysIds, Boolean overrideCustomEscalation, GlideRecord taskGR)

Specifies if there is an on-call user, group, or device. This API is applicable for a single shift, overlapping shifts, and custom escalation.

ℹ️ Note: Set the taskGR parameter to enable escalation plan evaluation.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupSysId</td>
<td>String</td>
<td>Sys ID of the group to check for.</td>
</tr>
<tr>
<td>gdt</td>
<td>GlideDateTime</td>
<td>yyyy-MM-dd HH:mm:ss format in the UTC timezone. Default is now.</td>
</tr>
<tr>
<td>nullifyOverrideRoster</td>
<td>Boolean</td>
<td>True if notification list for this group is not for a specific roster, false otherwise.</td>
</tr>
<tr>
<td>rotaSysIds</td>
<td>String</td>
<td>Comma-separated list of cmn_rota sys_ids to filter the records by.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>overrideCustomEscalation</td>
<td>Boolean</td>
<td>True to override default escalation policy with custom escalation, false otherwise.</td>
</tr>
<tr>
<td>taskGR</td>
<td>String</td>
<td>Optional. Current task record. If invoked with the optional taskGr parameter, this method populates escalatees based on escalation conditions.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if at least one match is found, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var groupSysId = '096fb59173062300e7af6238edf6a783';
var currentTime = new GlideDateTime();
var onCallRotation = new OnCallRotation();
var result = onCallRotation.who(groupSysId, currentTime);
gs.log(result);
```

Output:

```javascript
true
```

The following example shows how to create a custom escalation for service desk shift with condition P1 for an incident.

```javascript
/**
 * When the who() method is invoked with the optional taskGr parameter, 
 * escalatees are populated based on the conditional escalations
 */
var serviceDeskSysId = 'd625dccec0a8016700a222a0f7900d06';
var taskGr = new GlideRecord('incident');
taskGr.get('ed92e8d173d023002728660c4cf6a7bc');

var ocRotation = new OnCallRotation();
ocRotation.who(serviceDeskSysId, null, null, null, null, taskGr);
while (ocRotation.next()) {
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
OnCallRotation - whoAt(String groupSysId, GlideDateTime gdt, String rotaSysIds)

Specifies if there is an on-call user, group, or device. This API is applicable for a single shift, overlapping shifts, and custom escalation.

Parameters

| Name       | Type            | Description                                                                 |
|------------|-----------------|                                                                            |
| groupSysId | String          | Sys ID of the group.                                                       |
| gdt        | GlideDateTime   | yyyy-MM-dd HH:mm:ss format in the UTC timezone. Default is now.            |
| rotaSysIds | String          | Comma-separated list of cmn_rota Sys IDs to filter the records by.         |

Returns

| Type      | Description                                                                 |
|-----------|                                                                            |
| Boolean   | True if there is at least one on-call user, group, or device; false otherwise. |

```javascript
var groupSysId = '096fb59173062300e7af6238edf6a783';
var currentTime = new GlideDateTime();
var onCallRotation = new OnCallRotation();
var result = onCallRotation.whoAt(groupSysId, currentTime);
gs.log(result);
```

Output:

```javascript
true
```
OnCallRotation - startEscalations(GlideRecord now_GR, String eventName, String escalationScriptName)

Start the escalations for a document(incident now_GR) and group, and specify the event name for the escalation notifications. In addition, specify the name of the business rule to call when an escalation occurs. This method should be used after a succesful call to who or whoAt methods. This API is applicable for a single shift, overlapping shifts, and custom escalation.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>Document glide record. For example, Incident now_GR.</td>
</tr>
<tr>
<td>eventName</td>
<td>String</td>
<td>Event name for the escalation notifications.</td>
</tr>
<tr>
<td>escalationScriptName</td>
<td>String</td>
<td>Business rule to call when escalation occurs</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

OnCallRotation - addCurrentToNotified()

Add the current escalation member to the list of notified users so that the API knows who has been contacted. This API is applicable for a single shift, overlapping shifts, and custom escalation.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
OnCallRotation - whoIsNext(GlideRecord escalationGr)

Escalation business rules determine who in the shift should be contacted next. The escalation record passed to the method tracks the escalation between invocations of the business rule. This API is applicable for a single shift, overlapping shifts, and custom escalation.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>escalationGr</td>
<td>GlideRecord</td>
<td>GlideRecord of Rotation escalation [cmn_rota_escalation].</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

OnCallRotation - continueEscalations(GlideRecord escalationGr)

Continue the escalations for a document(incident now_GR) and group. This API is applicable for a single shift, overlapping shifts, and custom escalation.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>escalationGr</td>
<td>GlideRecord</td>
<td>GlideRecord of Rotation escalation(cmn_rota_escalation)</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

OnCallRotation - cancelEscalations(GlideRecord escalationGr)

Cancels any escalations for the GlideRecord that were started with the startEscalations API. This API is applicable for a single shift, overlapping shifts, and custom escalation.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>escalationGr</td>
<td>GlideRecord</td>
<td>GlideRecord of Rotation escalation(cm_rota_escalation)</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**OCSeriesEventGenerator - Global**

The OCSeriesEventGenerator API generates the repeating VEVENTS for a user's iCalendar.

**OCSeriesEventGenerator - getMemberCalendar (String groupId, String rotaId, String userId, Object dateRangeObj, Object memberSchedules, String calendarLink)**

Returns an array of the repeating on-call events.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupId</td>
<td>String</td>
<td>The group sys id.</td>
</tr>
<tr>
<td>rotaId</td>
<td>String</td>
<td>The rota sys id.</td>
</tr>
<tr>
<td>userId</td>
<td>String</td>
<td>The user sys id.</td>
</tr>
<tr>
<td>dateRangeObj</td>
<td>Object</td>
<td>Contains the from and to dates for the data set.</td>
</tr>
<tr>
<td>memberSchedules</td>
<td>Object</td>
<td>The user's schedule.</td>
</tr>
<tr>
<td>calendarLink</td>
<td>String</td>
<td>A link back to the on-call calendar in the instance.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array of the repeating on-call events.</td>
</tr>
</tbody>
</table>
OCSeriesEventGenerator - getRosterICalEvents (GlideRecord rotaScheduleSpanGR, GlideRecord memberScheduleSpanGR, Array seriesStartTimes, Object excludeItems, String repeatUntil, String timeZone, String calendarLink)

Returns an array of VEVENTs that represent the on-call for the provided rotaScheduleSpanGR and the memberScheduleSpanGR.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaScheduleSpanGR</td>
<td>GlideRecord</td>
<td>A cmn_schedule_span glide record for a rota.</td>
</tr>
<tr>
<td>memberScheduleSpanGR</td>
<td>GlideRecord</td>
<td>A cmn_schedule_span glide record for a rota member.</td>
</tr>
<tr>
<td>seriesStartTimes</td>
<td>Array</td>
<td>An array of the repeating rota span start times.</td>
</tr>
<tr>
<td>excludeItems</td>
<td>Object</td>
<td>The schedule items that need to be excluded.</td>
</tr>
<tr>
<td>repeatUntil</td>
<td>String</td>
<td>A formatted ScheduleDateTime string.</td>
</tr>
<tr>
<td>timeZone</td>
<td>String</td>
<td>Time zone for the on-call rota.</td>
</tr>
<tr>
<td>calendarLink</td>
<td>String</td>
<td>A link to the on-call calendar for this user.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of VEVENTs that represent the on-call for the provided rotaScheduleSpanGR and the memberScheduleSpanGR.</td>
</tr>
</tbody>
</table>

OCSeriesEventGenerator - getRosterMemberEvents (GlideRecord rotaMemberGR, Array seriesStartTimes, Object excludeItems, String repeatUntil, String calendarLink)

Returns an array of the repeating on-call VEVENTS.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotaMemberGR</td>
<td>GlideRecord</td>
<td>A cmn_rota_member record.</td>
</tr>
<tr>
<td>seriesStartTimes</td>
<td>Array</td>
<td>An array of the repeating rota span start times.</td>
</tr>
<tr>
<td>excludeItems</td>
<td>Object</td>
<td>The schedule items that need to be excluded.</td>
</tr>
<tr>
<td>repeatUntil</td>
<td>String</td>
<td>A formatted ScheduleDateTime string.</td>
</tr>
<tr>
<td>calendarLink</td>
<td>String</td>
<td>A link to the on-call calendar for this user.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of the repeating on-call VEVENTS</td>
</tr>
</tbody>
</table>

OCTimer - Global

The OCTimer API provides the ability to track the length of time it takes for a function to complete.

OCTimer - log (String name, String entry)

Logs the entry based on the function name.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Function name.</td>
</tr>
<tr>
<td>entry</td>
<td>String</td>
<td>Text that will be part of the output in the log.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

OCTimer - millisToTime (Integer millis)

Takes a millisecond value and returns a formatted duration.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>millis</td>
<td>Integer</td>
<td>Duration in milliseconds.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A formatted duration.</td>
</tr>
</tbody>
</table>

#### OCTimer - result ()
Calculates the time taken for each function registered.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Time taken for each function registered.</td>
</tr>
</tbody>
</table>

#### OCTimer - start (String name)
Initializes a timer based on the name provided.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Function name.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
**OCTimer - stop (String name)**

Registers the end of the timer for the provided name.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Function name.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Optional - Scoped, Global**

The Optional API interacts with a single record returned by the GlideQuery, Stream, or GlideRecord APIs, even when it does not exist. Write scripts that are less likely to result in an error by handling null or undefined query results.

You can get an Optional object in these ways:

- Return an Optional object from these methods in the GlideQuery class. For more information, see GlideQuery.
  - getBy()
  - get()
  - insert()
  - insertOrUpdate()
  - update()
  - selectOne()
  - avg()
  - max()
  - min()
  - sum()

- Return an Optional object from the find() method in the Stream class. For more information on Stream, see the Stream API.

- Use the lazy() method to generate the value of the Optional if ever needed.

These methods are static and do not require an instance of the class:
You can use these static methods with any API that returns a single value such as GlideRecord.

Use the Optional API in scoped or global server-side scripts. This API requires the GlideQuery [com.sn glidequery] plugin.

Implementation

This API can work with the GlideQuery and Stream APIs in a builder pattern where the method calls chain together, each method building on the returned result of the previous method. Use methods to define the attributes of the query. The methods do not execute until you call a terminal method, a method that returns a query result, allowing you to define the requirements of the query before executing it.

If the query returns a single record, the system wraps the result in an Optional object. If the query returns a stream of records, the system wraps the result in a Stream object. These objects let you manage the result using a set of methods in each API.

For example, here's a script that performs a query on the Task table and groups the records by priority and returns groups with total reassignments greater than four.

```javascript
var query = new global.GlideQuery('task')
  .where('active', true) //Returns new GlideQuery object with a "where" clause.
  .groupBy('priority') //Returns new GlideQuery object with a "group by" clause.
  .aggregate('sum', 'reassignment_count') //Returns new GlideQuery object with a "sum(reassignment_count)" clause.
  .having('sum', 'reassignment_count', '>', 4) //Returns new GlideQuery object with a "having reassignment_count > 4" clause.
  .select() //Returns a stream of records wrapped in a Stream object.
  .toArray(10); //Terminal method in the Stream class that executes the query and returns the result.
```

Terminal methods

For performance reasons, a query only fetches data when you call a terminal method. These are the terminal methods from the Optional class:
• get()
• orElse()
• ifPresent()
• isEmpty()
• isPresent()

Optional - empty(String reason)
Returns an empty Optional object. Use this method in an Else clause to handle a query that might not return a result.

ℹ️ Note: This method is static. You do not need an instance of the class to use this method.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reason</td>
<td>String</td>
<td>Optional. Reason displayed in the log when Optional.get() is called on the empty Optional object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional</td>
<td>Object used to interact with a single record.</td>
</tr>
</tbody>
</table>

This example shows you how to generate an empty Optional object when a query does not return a result.

```java
var now_GR = new GlideRecord('task');
now_GR.addQuery('approval', 'not requested');
now_GR.query();
var optional;
if (now_GR.next()) {
    optional = Optional.of(now_GR.getUniqueValue());
} else {
    optional = Optional.empty("no results");
}
gs.info(optional.get());
```

Output:
Optional - filter(Function predicate)

Applies a predicate function, a function that takes a single value and returns true or false, to the record inside the Optional object. If the function returns true, the method returns the Optional record unchanged. If the function returns false, it returns an empty Optional object.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>predicate</td>
<td>Function</td>
<td>Predicate function to apply to the value inside the Optional object. Must return a Boolean value.</td>
</tr>
</tbody>
</table>

This example shows you how to apply a filter function to an Optional result.

```java
var filteredQuery = new global.GlideQuery('sys_user')
  .getBy({ sys_id: 'f682abf03710200044e0bfc8bcbe5d38' }, ['phone'])
  .filter(function (user) {
    return phoneRegex.test(user.phone);
  });
```

Optional - flatMap(Function fn)

Applies a function that returns an Optional object to the result of a query. Use this method to perform a second query using the result of the first.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fn</td>
<td>Function</td>
<td>Function to apply to the results of the query that returned the Optional object.</td>
</tr>
</tbody>
</table>
This example shows how to execute a query of the User table based on the result of a previous query.

```javascript
new global.GlideQuery('alm_asset')
  .whereNotNull('owned_by')
  .selectOne('owned_by')
  .flatMap(function (asset) {
    return new global.GlideQuery('sys_user')
      .getBy({ sys_id: asset.owned_by }, ['first_name', 'last_name', 'company.name'])
  })
  .ifPresent(GQ.jsonDebug);
```

Output:

```json
{
  "sys_id": "46d59205a9fe198101d603f5de37bfa3",
  "first_name": "John",
  "last_name": "Bohnhamn",
  "company": {
    "name": "ACME North America"
  }
}
```

### Optional - get()

Returns the record inside the Optional object, or throws an error if the query does not return a record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any</td>
<td>The record inside the Optional object. If the value is null or undefined, the system throws an error.</td>
</tr>
</tbody>
</table>
This example shows how to get the value of a single record.

```javascript
var value = new global.GlideQuery('sys_user')
    .selectOne('first_name') //Returns the result of the query inside an Optional object
    .get(); //Calls Optional.get() on the Optional object

gs.info(JSON.stringify(value));
```

Output:

```json
{
    "first_name": "fred",
    "sys_id": "005d500b536073005e0addeeff7b12f4"
}
```

**Optional - ifPresent(Function fn)**

Applies a function to the record within an Optional object. If the Optional object does not contain a record, the function does not execute.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fn</td>
<td>Function</td>
<td>The function to apply to the record within the Optional object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to print a value if it exists.

```javascript
var user = new global.GlideQuery('sys_user')
    .where('sys_id', 'f682abf03710200044e0bfc8bcbe5d38')
    .selectOne('zip')
    .ifPresent(function (user) {
        gs.info('Zip Code: ' + user.zip);
    });
```

**Optional - isEmpty()**

Returns true if the Optional object is empty.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the result of a query contains a value.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The query returns null or undefined.</td>
</tr>
<tr>
<td></td>
<td>• false: The query returns a value.</td>
</tr>
</tbody>
</table>

This example shows how to check whether the result of a query is empty.

```javascript
var checkEmpty = new global.GlideQuery('sys_user')
  .where('last_name', 'Barker')
  .selectOne()
  .isEmpty();

gs.info(checkEmpty);
```

Output:

```
true
```

Optional - isPresent()

Checks whether an Optional object contains a value.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the result of a query contains a value.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The query returns a value.</td>
</tr>
<tr>
<td></td>
<td>• false: The query returns null or undefined.</td>
</tr>
</tbody>
</table>

This example shows how to check whether a query returns a result.

```javascript
var checkPresent = new global.GlideQuery('sys_user')
    .where('last_name', 'Luddy')
    .selectOne('first_name')
    .isPresent();

gs.info(checkPresent);
```

Output:

```javascript
true
```

**Optional - lazy(Function lazyGetFn)**

Returns a new Optional object. Instead of containing the record, the object contains a function to get the record that is only called if and when requested in the code.

Use this method to delay getting the value until it's needed. You might do this if requesting the value from a slow source and don't want to slow down your code unnecessarily. Otherwise, you can return an Optional object using the GlideQuery and Stream APIs.

⚠️ **Note:** This method is static. You do not need an instance of the class to use this method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lazyGetFn</td>
<td>Function</td>
<td>Function that returns a single record as a result of a query. For example:</td>
</tr>
</tbody>
</table>
|          |         | ```javascript
        var userGr = new GlideRecord('sys_user');
    ``` |
This example shows how to get an Optional object based on a GlideRecord query.

```javascript
var userOptional = global.Optional.lazy(function () {
    var userGr = new GlideRecord('sys_user');
    userGr.setLimit(1);
    userGr.query();
    return userGr.next() ? userGr.getUniqueValue() : null;
});
gs.info(userOptional);
```

Output:

Optional<005d500b536073005e0addeeff7b12f4>

### Optional - map(Function fn)

Applies a function to the result of a query.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fn</td>
<td>Function</td>
<td>Function to apply to the result of the query.</td>
</tr>
</tbody>
</table>

This example shows how to apply a function that transforms a value to upper case to the result of a query.

```javascript
var value = new global.GlideQuery('sys_user')
    .whereNotNull('first_name')
    .selectOne('first_name')
    .map(function (user) {
```

---

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
return user.first_name.toUpperCase();
});

gs.info(value);

Output:

Optional<ABEL>

Optional - of(Any value)
Wraps a given value in an Optional object. For example, you can wrap the result of a GlideRecord query in an Optional object to use the associated methods.

Note: This method is static. You do not need an instance of the class to use this method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Any</td>
<td>Value inside the Optional object.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional</td>
<td>Object containing the passed-in value in the format Optional&lt;value&gt;.</td>
</tr>
</tbody>
</table>

This example shows you how to generate an Optional object based on a GlideRecord query.

```javascript
var now_GR = new GlideRecord('task');
now_GR.addQuery('approval', 'not requested');
now_GR.query();
var optional;
if (now_GR.next()) {
  optional = Optional.of(now_GR.getUniqueValue());
} else {
  optional = Optional.empty("no results");
}
gs.info(optional.get());
```

Output:
Optional - orElse(Any defaultValue)

Adds a default value within the Optional object if the query does not return any results.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultValue</td>
<td>Any</td>
<td>Value within the Optional object if the query does not return any results.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any</td>
<td>Value within the Optional object if the query does not return any results.</td>
</tr>
</tbody>
</table>

This example shows how to return a value, even when the query is incorrect.

```
var user = new global.GlideQuery('sys_user')
  .get('1234', ['first_name', 'last_name'])
  .orElse({ first_name: 'Default', last_name: 'User' });

gs.info(JSON.stringify(user))
```

Output:

```
{
  "first_name": "Default",
  "last_name": "User"
}
```

**OrderGuide - Scoped**

The OrderGuide API enables you to initialize and view order guide details.

To use this class in a scoped application, use the `sn_sc` namespace identifier. The Service Catalog Scoped API plugin (com.glideapp.servicecatalog scopedefault is required to access the OrderGuide API.

**OrderGuide - OrderGuide(String sys_id)**

Creates an instance of the OrderGuide class with the specified sys_id.
### OrderGuide - getID()

Returns the `sys_id` of the order guide.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td><code>sys_id</code> of the order guide.</td>
</tr>
</tbody>
</table>

#### Example:

```javascript
var cart = new sn_sc.OrderGuide("6690750f4f7b4200086eed18110c761");
console.log(cart.getID());
```

**Output:**

`6690750f4f7b4200086eed18110c761`

### OrderGuide - init(Map request)

Initialises the order guide with the specified catalog items and the variables, and returns the order guide.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request</td>
<td>Map</td>
<td>A JSON object with the Catalog item and variable details.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map</td>
<td>A JSON object with the initialised order guide details.</td>
</tr>
</tbody>
</table>

**Example:**

```javascript
var guide = new sn_sc.OrderGuide('6690750f4f7b4200086eed18110c761');

var map = {};  
map.variables = {};  
//map.sysparm_id = '6690750f4f7b4200086eed18110c761';  
map.variables['IOC633d0f4f7b4200086eed18110c74d'] =  
'221f3db5c6112284009f4becd3039cc9';  //Here ce433d0f4f7b4200086eed18110c74d is the sys_id  
of the variable and 221f3db5c6112284009f4becd3039cc9 is its value

var includedItems = guide.init(map)
```

**Output:**

```javascript
{
  "cascade_variable_map": {},
  "items": [
    {
      "order": "-1",
      "quantity": "1",
      "show_quantity": false,
      "sys_id": "a4022d7b87c20300e3010cf888cb0bb2"
    },
    {
      "order": "300",
      "quantity": "1",
      "show_quantity": false,
      "sys_id": "186d917a6fab7980575967d3d9d3e4f2"
    },
    {
      "order": "600",
      "quantity": "1",
      "show_quantity": false,
      "sys_id": "3b3ae7feda1be1004e5c08239e522b"
    }
  ],
  "variable_assignments":{}
}
```
**OrderGuide - isIncludeItems()**

Specifies if the **Show Include Toggle** (include_items) check box is selected for the specified order guide.

⚠️ **Note:** The **Show Include Toggle** field does not appear on the Order guide form by default.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Boolean</td>
</tr>
</tbody>
</table>

**Example**

```javascript
var orderGuide = new sn_sc.OrderGuide("6690750f4f7b4200086eeed18110c761");
console.log(orderGuide.isIncludeItems());
```

**Output:**

`true`

**OrderGuide - isTwoStep()**

Specifies if the two-step checkout is enabled.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the two-step checkout is enabled. Else returns false.</td>
</tr>
</tbody>
</table>

**Example:**

```javascript
var orderGuide=new sn_sc.OrderGuide("6690750f4f7b4200086eed18110c761");
console.log(orderGuide.isTwoStep());
```

**Output:**
```
false
```

---

**OrderGuide - isUseCustomCart()**

Specifies if a separate cart (different from that for catalog items) usage is enabled for a two-step order guide.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if a separate cart usage is enabled for a two-step order guide. Else, returns false.</td>
</tr>
</tbody>
</table>

**Example:**

```javascript
var orderGuide=new sn_sc.OrderGuide("6690750f4f7b4200086eed18110c761");
console.log(orderGuide.isUseCustomCart());
```

**Output:**
```
false
```

---

**OrderGuide - navigateFromMap(Map itemDetails)**

Navigates to the catalog items of an order guide.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>itemDetails</td>
<td>Map</td>
<td>A JSON object with details of catalog items in the order guide.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example:

```javascript
var orderGuide=new sn_sc.OrderGuide.navigateFromMap(itemdetails);
```

PADomainUtils - Global

The PADomainUtils API enables you to copy Performance Analytics records between different domains on the same instance.

Use this API in server scripts to copy Performance Analytics records such as indicators, breakdowns, and dashboards, to different domains. This API enables you to create a Performance Analytics record in one domain and copy that record to any number of additional domains.

⚠ Note: This API cannot copy records into the Global domain.

To use PADomainUtils, you must satisfy these requirements:

- Performance Analytics must be enabled.
- The user running the script must have the admin role.
- The instance must use domain separation.
- The script must be run from the global domain.
- When moving or copying records, the source and target domains must be different.

PADomainUtils - copy(String runAs)

Copies Performance Analytics records to a different domain.

To copy dashboards or scheduled jobs, see copyDashboard and copyJob.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>runAs</td>
<td>String</td>
<td>The user whose domain you want to copy records to.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
// copy all the Performance Analytics records from global to user's domain
var pa = new SNC.PADomainUtils();
pa.copy('09ff3d105f231000b12e3572f2b4775d');
```

### PADomainUtils - copyDashboard(String dashboardId, String runAs)

Copy a dashboard to another domain.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dashboardId</td>
<td>String</td>
<td>The sys_id of the dashboard to copy.</td>
</tr>
<tr>
<td>runAs</td>
<td>String</td>
<td>The user whose domain you want to copy the dashboard to.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
//Copy Incident Management dashboard from global to user's domain
var pa = new SNC.PADomainUtils();
pa.copyDashboard('a64b7031d7201100b96d45a3ce610335','09ff3d105f231000b12e3572f2b4775d');
```

### PADomainUtils - copyJob(String paJob, String runAs)

Copies a Performance Analytics scheduled data collection job record to another domain.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paJob</td>
<td>String</td>
<td>The sys_id of a Performance Analytics scheduled data collection job [sysauto_pa] record.</td>
</tr>
<tr>
<td>runAs</td>
<td>String</td>
<td>The user whose domain you want to copy the job to.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>An error message if an error occurs, or an empty string if there is no error.</td>
</tr>
</tbody>
</table>

```javascript
// No source domain needs to be set
var pa = new SNC.PADomainUtils();
// copy the ['PA Incident] Daily Data Collection job'
// set the 'run as' of the new record to be the 'acme.itil' user
// first argument is the sys_id of the sysauto_pa record
// the second is the sys_id of the acme.itil user record
pa.copyJob('82ba2023d7101100b96d45a3ce6103cd', '797d14341f1310005a3637b8ec8b7010');
```

## PADomainUtils - isWriteable(String table, String id)

Evaluate if you can write to a specific record identified by table and sys_id.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>The name of the table containing the record to query, such as pa_indicators.</td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>The sys_id of the record to query.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Indicates that you can write to the specified record. Returns true if the record exists within the domain of the current user. Returns false if the record does not exist, or is in a different domain.</td>
</tr>
</tbody>
</table>
PADomainUtils - move(String runAs)

Moves Performance Analytics records to a different domain.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>runAs</td>
<td>String</td>
<td>The user whose domain you want to copy records to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

// move all the Performance Analytics records from the global to the customers domain
var pa = new SNC.PADomainUtils();
pa.move('774190f01f1310005a3637b8ec8b70ef')

PADomainUtils - PADomainUtils()

Instantiates a new PADomainUtils object to move or copy Performance Analytics records from the global domain.

Use the PADomainUtils(String domainFrom) constructor instead when moving or copying records from a domain other than the global domain.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

// PADomainUtils initialized with the global domain
var globalUtils = new SNC.PADomainUtils();
PADomainUtils - PADomainUtils(String domainFrom)

 Instantiates a new PADomainUtils object to move or copy Performance Analytics records from the specified domain.

 Use the PADomainUtils() constructor instead when moving or copying from the global domain.

 Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainFrom</td>
<td>String</td>
<td>The domain to copy records from.</td>
</tr>
</tbody>
</table>

// c90d4b084a362312013398f051272c0d is the sys id of the ACME domain
var acmeUtils = new SNC.PADomainUtils('c90d4b084a362312013398f051272c0d');

PADomainUtils - setFoundation(Boolean foundation)

 Use this method to move or copy only foundational records in a hybrid domain configuration.

 You can implement a hybrid approach by maintaining some types of record in a parent domain and some types in child domains. Generally, the parent domain contains foundational records and the child domain contains higher-level records, such as widgets. The following types of record are considered foundational records.

- Bucket groups
- Buckets
- Scripts
- Breakdown sources
- Indicator sources
- Filters
- Breakdowns
- Managed sources
- Manual breakdowns
- Breakdown mappings
- Breakdown relations
Other Performance Analytics records such as widgets and indicators are not foundational records. Set this method to false to move or copy these additional records as well.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>foundation</td>
<td>Boolean</td>
<td>Indicates if only foundational records should be copied or moved by this PADomainUtils object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PADomainUtils</td>
<td>The object calling this function.</td>
</tr>
</tbody>
</table>

```javascript
var pa = new SNC.PADomainUtils().setFoundation(true);
pa.copy('bb6b58b01f1310005a3637b8ec8b70dd');
```

### PADomainUtils - setOverrides(Boolean overrides)

Use this method before copying records to set the sys_override value of the new record to the original parent record.

Using this method enables you to automatically override records in a parent domain. By overriding the parent records, the parent records do not impact the child domain. If the source domain is not the parent of the target domain when copying records, setting the sys_override value will not have any impact on behavior. You can specify an override only when copying records, not when moving records.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>overrides</td>
<td>Boolean</td>
<td>Indicates that copied records in a child domain should override the source record in the parent domain. This value is true by default.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PADomainUtils</td>
<td>The object calling this function.</td>
</tr>
</tbody>
</table>

```javascript
var pa = new SNC.PADomainUtils('c90d4b084a362312013398f051272c0d');
pa.setOverrides(false);
pa.copy('bb6b58b01f1310005a3637b8ec8b70dd');
```

### Paragraph - Scoped, Global

Creates a Paragraph object representing a block of text in a PDF.

This API is part of the ServiceNow PDF Generation Utilities plugin (com.snc.apppdfgenerator) and is provided within the sn_pdfgeneratorutils namespace. The plugin is activated by default.

This API is a component used with the Document API to generate a PDF.

#### Paragraph - Paragraph(String text)

Instantiates a new Paragraph object containing a string.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>Paragraph block of text.</td>
</tr>
</tbody>
</table>

The following example shows how to create a Paragraph object. For a document usage example, see Document API.

```javascript
var para = new Paragraph("hello");
```

#### Paragraph – addNewLine()

Adds an empty line after a paragraph in a document.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The following example shows how to add a new line after a paragraph in a document. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);
var myPara = new sn_pdfgeneratorutils.Paragraph("This is a paragraph.");
myPara.addNewLine();
document.addParagraph(myPara);
// save pdf as attachment to target record in the Incident table
document.saveAsAttachment("incident", ",<record_sys_id>", "addText.pdf");
```

**Paragraph – addParagraph(Paragraph paragraph)**

Adds a paragraph. You can use this method to create a block of paragraphs with automatic line breaks.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paragraph</td>
<td>Paragraph</td>
<td>Paragraph object.</td>
</tr>
</tbody>
</table>

The following example shows how to add a section of paragraphs to a document. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);
var sectionPara = new sn_pdfgeneratorutils.Paragraph("This is the first paragraph.");
var subPara1 = new sn_pdfgeneratorutils.Paragraph("Pellentesque nec neque interdum tristique at ut lacus. Nam eget sollicitudin.");
```
var subPara2 = new sn_pdfgeneratorutils.Paragraph("Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vel ultrices erat.");
var subPara3 = new sn_pdfgeneratorutils.Paragraph("Aenean fermentum lorem congue metus faucibus, vitae viverra quam eleifend. Donec sed risus quis eros suscipit efficitur.");

sectionPara.addParagraph(subPara1);
sectionPara.addParagraph(subPara2);
sectionPara.addParagraph(subPara3);

document.addParagraph(sectionPara);

// save pdf as attachment to target record in the Incident table
document.saveAsAttachment("incident", "<record_sys_id>", "filename.pdf");

### Paragraph – addString(String content)

Adds a string of text to a paragraph. This method does not automatically insert a space preceding the content.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>content</td>
<td>String</td>
<td>Information to include in a paragraph.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add a new sentence to a paragraph. For a document usage example, see [Document API](#).

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var myPara = new sn_pdfgeneratorutils.Paragraph("This is the first sentence.");

myPara.addString(" This is the second sentence in the same paragraph. Spaces are not inserted automatically.")

document.addParagraph(myPara);
```
Paragraph – addStyle(Style style)
Applies a predefined style to paragraph text.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>style</td>
<td>Style</td>
<td>Style to apply to this element.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to apply a style to a paragraph. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// Create a font color (result is purple)
var fontColor = new sn_pdfgeneratorutils.Color([0.5,0.0,0.5]);

// Create a style for your paragraph
var paraStyle = new sn_pdfgeneratorutils.Style();
paraStyle.setFontColor(fontColor);
paraStyle.setFontSize(10);

var myPara = new sn_pdfgeneratorutils.Paragraph("This paragraph has style.");

myPara.addStyle(paraStyle);

document.addParagraph(myPara);

// save pdf as attachment to target record in the Incident table
document.saveAsAttachment("incident", "<record_sys_id>", "filename.pdf");
```

Paragraph – setFixedPosition(Number left, Number bottom, Number width)
Sets a paragraph element to a fixed position on the page.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>left</td>
<td>Number</td>
<td>Indentation from the left side of the PDF page in points.</td>
</tr>
<tr>
<td>bottom</td>
<td>Number</td>
<td>Position from the bottom of the PDF page in points.</td>
</tr>
<tr>
<td>width</td>
<td>Number</td>
<td>Width of the paragraph element in points. This value determines the length at which the line breaks.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a fixed position on a page. For a document usage example, see Document API.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// Create a style
var paraStyle = new sn_pdfgeneratorutils.Style();
paraStyle.setFontSize(48);
paraStyle.setBold();

// my paragraph
var para = new sn_pdfgeneratorutils.Paragraph("Document Title");
para.setFixedPosition(204,400,240);
para.setTextAlignment("text-center");
para.addStyle(paraStyle);

document.addParagraph(para);

// save pdf as attachment to target record in the Incident table
document.saveAsAttachment("incident", "<record_sys_id>", "fileName.pdf");
```

Paragraph – setMargin(Number margin)

Sets each paragraph margin.
To set all four margins with one or more unique values, use `setMargins()`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>margin</td>
<td>Number</td>
<td>Value of the top, right, bottom, and left margins in points.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the all margins of the paragraph to 48 points.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var margins = 48.0;
var myPara = new sn_pdfgeneratorutils.Paragraph("Paragraph text with all margins set to the same value.");
myPara.setMargin(margins);
document.addParagraph(myPara);

// save pdf as attachment to target record in the Incident table
document.saveAsAttachment("incident", "<record_sys_id>", "docName.pdf");
```

**Paragraph – setMarginBottom(Number margin)**

Sets the bottom margin of a paragraph.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>margin</td>
<td>Number</td>
<td>Height of the bottom margin in points.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the bottom margin of a paragraph to one point.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var marginVal = 1.0;

var paraMarginBottom = new sn_pdfgeneratorutils.Paragraph("Paragraph text with bottom margin set.");
paraMarginBottom.setMarginBottom(marginVal);

document.addParagraph(paraMarginBottom);

// save pdf as attachment to target record in the Incident table
document.saveAsAttachment("incident", "<record_sys_id>", "docName.pdf");
```

### Paragraph – `setMarginLeft(Number margin)`

Sets the left margin of a paragraph.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>leftMargin</td>
<td>Number</td>
<td>Width of the left margin in points.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the left margin of a paragraph to one point.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var marginVal = 1.0;
```
```javascript
var paraMarginLeft = new sn_pdfgeneratorutils.Paragraph("Paragraph text with left margin set.");
paraMarginLeft.setMarginLeft(marginVal);
document.addParagraph(paraMarginLeft);

// save pdf as attachment to target record in the Incident table
document.saveAsAttachment("incident", "<record_sys_id>", "docName.pdf");
```

**Paragraph – setMarginRight(Number margin)**

Sets the right margin of a paragraph.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>margin</td>
<td>Number</td>
<td>Width of the right margin in points.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the right margin of a paragraph to one point.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var marginVal = 1.0;

var paraMarginRight = new sn_pdfgeneratorutils.Paragraph("Paragraph text with right margin set.");
paraMarginRight.setMarginRight(marginVal);
document.addParagraph(paraMarginRight);

// save pdf as attachment to target record in the Incident table
document.saveAsAttachment("incident", "<record_sys_id>", "docName.pdf");
```
Paragraph – setMargins(Number marginTop, Number marginRight, Number marginBottom, Number marginLeft)

Sets a size for each paragraph margin.

To set each margin to the same value, use setMargin().

| Parameters |
|---|---|---|
| Name | Type | Description |
| topMargin | Number | Height of the top margin in points. |
| rightMargin | Number | Width of the right margin in points. |
| bottomMargin | Number | Height of the bottom margin in points. |
| leftMargin | Number | Width of the left margin in points. |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set paragraph margins. For a document usage example, see Document API.

```javascript
var para = new sn_pdfgeneratorutils.Paragraph("Paragraph text.");

var topMargin = 1.0;
var rightMargin = 1.0;
var bottomMargin = 1.0;
var leftMargin = 1.5;

para.setMargins(marginTop, marginRight, marginBottom, marginLeft);
```

Paragraph – setMarginTop(Number margin)

Sets the top margin of a paragraph.

| Parameters |
|---|---|---|
| Name | Type | Description |
| margin | Number | Height of the top margin in points. |
The following example shows how to set the top margin of a paragraph to one point.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("LETTER");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var marginVal = 1.0;

var paraMarginTop = new sn_pdfgeneratorutils.Paragraph("Paragraph text with top margin set.");
paraMarginTop.setMarginTop(marginVal);

document.addParagraph(paraMarginTop);

// save pdf as attachment to target record in the Incident table
document.saveAsAttachment("incident", "<record_sys_id>", "docName.pdf");
```

### Paragraph – `setTextAlignment(String alignment)`

Sets the text alignment of this paragraph.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alignment</td>
<td>String</td>
<td>Text alignment position. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• text-center: Aligns text to the center.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• text-justified: Modifies the space between</td>
</tr>
<tr>
<td></td>
<td></td>
<td>characters to completely fill text between the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>left and right sides. The final line is left-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>aligned.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• text-justified-all: Justifies text alignment</td>
</tr>
<tr>
<td></td>
<td></td>
<td>including the final line.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• text-left: Align text to the left.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• text-right: Align text to the right.</td>
</tr>
</tbody>
</table>
The following example shows how to set the paragraph text to left alignment.

```javascript
var paragraph = new sn_pdfgeneratorutils.Paragraph("This paragraph text is centered.");
var alignment = "text-center";
paragraph.setTextAlignment(alignment);
```

**PAScorecard - Scoped**

The PAScorecard API enables you to fetch data about indicators and their associated records, such as breakdowns.

This API requires the Performance Analytics (com.snc.pa.analytics_center) application.

**PAScorecard - addParam(String parameter, String value)**

Add a query parameter to filter the returned scores.

Call this method multiple times on the same PAScorecard object to pass multiple parameters, such as the indicator sys_id and a breakdown sys_id. After specifying all parameters, call `query()` to run the query.

If you query a PAScorecard object with no parameters, the API returns a list of all indicators that are displayed on the Analytics Hub, with their scores.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uuid</td>
<td>String</td>
<td>Enter a colon-separated list of sys_id values to specify which indicators, breakdowns, aggregates, and domains to query. The parameter follows this format: <code>&lt;indicator_sys_id&gt;:&lt;breakdown_sys_id&gt;:&lt;elements_filter_sys_id or element_sys_ids&gt;:&lt;lvl-2 breakdown_sys_id&gt;:&lt;lvl-2 elements_filter_sys_id or element_sys_ids&gt;:&lt;aggregate_sys_id&gt;:&lt;domain_sys_id&gt;</code></td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The parameter must begin with the sys_id of an indicator record. Optionally, you can append the sys_id values of a breakdown and breakdown element to group the response based on the breakdown, and the sys_id of an aggregate to apply that aggregate. You can use a breakdown with an aggregate, or use only one. For information about obtaining the sys_id values of records, see Unique record identifier (sys_id).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>breakdown</td>
<td>String</td>
<td>Sys_id of a breakdown to return chart information organized as defined by the breakdown. For example, the sys_id of a priority breakdown to return separate task chart information for each priority value, such as the number of open incidents / Priority / 2 - High. Data type: String</td>
</tr>
<tr>
<td>breakdown_relation</td>
<td>String</td>
<td>Specify the sys_id of a breakdown relation to breakdown the returned data using that relation. You can view available breakdown relations by setting the include_available_breakdowns parameter to true.</td>
</tr>
<tr>
<td>elements_filter</td>
<td>String</td>
<td>Specify the sys_id of an elements filter to apply that filter to the returned data.</td>
</tr>
</tbody>
</table>
| display             | String | Flag that indicates the type of indicators to return. Valid values:  
  - true: Return only indicators that display on the Analytics Hub.  
  - all: Return all indicators.  
  Data type: String  
  Default: true |
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| favorites | String   | Flag that indicates whether to return only indicators that are favorites of the querying user. Valid values:  
  • true: Return only indicators that are favorites of the querying user.  
  • false: Return all indicators.  
  Data type: Boolean  
  Default: false |
| key    | String   | Flag that indicates whether to return results only for key indicators. Valid values:  
  • true: Return the trendline_scores element.  
  • false: Do not return the trendline_scores element.  
  Data type: Boolean  
  Default: false |
| target | String   | Flag that indicates whether to return results only for indicators that have a target set on the Analytics Hub. Valid values:  
  • true: Only return results for indicators that have a target set.  
  • false: Return results for all applicable indicators.  
  Data type: Boolean  
  Default: false |
| contains | String   | Comma-separated list of names or descriptions to return results only from indicators with a matching value. All the comma-separated list items must match, or no results are returned: The list follows AND logic, not OR logic.  
  Data type: String |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tags</td>
<td>String</td>
<td>Enter an indicator group sys_id to return the indicators in that group. Do not use <code>uuid</code> with this parameter.</td>
</tr>
<tr>
<td>per_page</td>
<td>String</td>
<td>Maximum number of indicators each query can return on a page. Data type: Number Default: 10 Maximum: 100</td>
</tr>
<tr>
<td>page</td>
<td>String</td>
<td>Specify the page number. For example, when querying 20 Analytics Hubs with the default <code>per_page</code> value (10), specify a page value of 2 to retrieve Analytics Hubs 11-20.</td>
</tr>
<tr>
<td>sortby</td>
<td>String</td>
<td>Value to use when sorting results. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• bullet</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• change</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• changeperc</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• date</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• default</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• direction</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• duedate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• gap</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• gapperc</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• group</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• indicator_group</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• name</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• order</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• target</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>trend</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>value</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: value</td>
</tr>
<tr>
<td>sortdir</td>
<td>String</td>
<td>Sort direction. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• asc: Denotes ascending</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• des: Denotes descending</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Descending</td>
</tr>
<tr>
<td>display_value</td>
<td>String</td>
<td>Data retrieval operation for reference and choice fields.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Based on this value, the display value and/or the actual value in the database are retrieved.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true returns display values for all of the fields.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false returns actual values from the database. If a value is not specified, this parameter defaults to false.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• all returns both actual and display values.</td>
</tr>
<tr>
<td>exclude_reference</td>
<td>String</td>
<td>Flag that indicates whether to hide additional information provided for reference fields, such as the URI to the reference resource.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Hide additional information provided for reference fields.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not hide additional information provided for reference fields.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>include_scores</td>
<td>String</td>
<td>Flag that indicates whether to return indicator scores for the entire time range selected on the Analytics Hub. If a value is not specified, this parameter defaults to false and returns only the most recent score value. To constrain the date range of the scores that are returned, combine this parameter with the <strong>from</strong> and <strong>to</strong> parameters.</td>
</tr>
<tr>
<td>from</td>
<td>String</td>
<td>Earliest date to return scores from. Only scores from this date or later are returned. This parameter requires that <strong>include_scores</strong> is set to <strong>true</strong>.</td>
</tr>
<tr>
<td>to</td>
<td>String</td>
<td>Latest date from which to return scores. Only scores from this date or earlier are returned. This parameter requires that <strong>include_scores</strong> is set to <strong>true</strong>.</td>
</tr>
<tr>
<td>step</td>
<td>String</td>
<td>Numeric value to skip scores, based on the indicator frequency. For example, specify a value of 3 to return only scores from every third day for a daily indicator, or from every third week for a weekly indicator. Data type: Number</td>
</tr>
<tr>
<td>limit</td>
<td>String</td>
<td>Maximum number of scores to return. Data type: Number</td>
</tr>
<tr>
<td>include_available_breakdowns</td>
<td>String</td>
<td>Flag that indicates whether to return all available breakdowns for an indicator. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Return all available breakdowns for an indicator.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Return no breakdowns.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data type: Boolean                                                                                                                                         Default: false</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| include_available_aggregates | String  | Flag that indicates whether to return all possible aggregates for an indicator, including aggregates that have already been applied. Valid values:  
• true: Return all possible aggregates for an indicator, including aggregates that have already been applied.  
• false: Return no aggregates.  
Data type: Boolean
Default: false |
| include_realtime     | String  | Flag that indicates whether to return the realtime_enabled element which indicates if real-time scores are enabled for the indicator, and the realtime_value element which contains the real-time score value. This parameter is not supported for formula indicators. Valid values:  
• true: Return the realtime_enabled element.  
• false: Do not return the realtime_enabled element.  
Data type: Boolean
Default: false |
| include_target_color_scheme | String  | Flag that indicates whether to return the target_color_scheme element that contains the minimum and maximum values, and the color of each section of the target color scheme for the Analytics Hub. Valid values:  
• true: Return the target_color_scheme element.  
• false: Do not return the target_color_scheme element.  
Data type: Boolean |
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>include_forecast_scores</td>
<td>String</td>
<td>Flag that indicates whether to return the forecast_scores element that contains an array of date-value pairs that define the forecast data for the Analytics Hub. This parameter requires that the include_scores parameter is also set to true.</td>
</tr>
<tr>
<td>include_trendline_scores</td>
<td>String</td>
<td>Flag that indicates whether to return the trendline_scores element that contains an array of date-value pairs that define the Analytics Hub trendline. This parameter requires that the include_scores parameter is also set to true.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

In this example, the uuid parameter specifies the Number of open incidents indicator, and the breakdown parameter specifies the Priority breakdown. Both parameters have the sys_id of the respective records as their values. The query() function returns the results as an object.

```javascript
var sc = new SNC.PAScorecard(); //in a scoped app, do not use the SNC namespace
sc.addParam('uuid', 'fb007202d7130100b96d45a3ce6103b4'); // Number of open incidents
sc.addParam('breakdown', '0df47e02d7130100b96d45a3ce610399'); // by Priority
var result = sc.query(); // Query results, which are returned as an object
for (var i = 0; i < result.length; i++)
gs.info(result[i].name + ': ' + result[i].value + ' ' + result[i].unit.display_value);
```

**PAScorecard - query()**

Perform a query based on the specified parameters and return the results as an object.

Before calling this method, configure parameters for the PAScorecard object by calling `addParam(String parameter, String value)`.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The PAScorecard object.</td>
</tr>
</tbody>
</table>

PAScorecard - result()

Get the last query result as an object.

This method does not perform a query. To perform a query before returning the result, use query().

This function cannot run in a scope other than global.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The results from the last query, returned as a JS object.</td>
</tr>
</tbody>
</table>

PASnapshot - Scoped

The PASnapshot API enables you to query information about Performance Analytics snapshots. Snapshots are the lists of records (sys_ids) that are collected at the time that the scores for those records are collected. A snapshot is made only for indicators with Collect records selected.

You can query information about a snapshot at a certain date using the indicator sys_id and date, and perform comparisons between snapshots for an indicator at different dates.
PASnapshot - getCompareIDs(String sys_id, Number date1, Number date2, String type)

Compare records in snapshots for a specified indicator at multiple dates, such as to identify records included in one snapshot but not the other.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The indicator sys_id.</td>
</tr>
<tr>
<td>date1</td>
<td>Number</td>
<td>The date of the first snapshot, in the format yyyymmdd.</td>
</tr>
<tr>
<td>date2</td>
<td>Number</td>
<td>The date of the second snapshot, in the format yyyymmdd.</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>Specifies what data to retrieve. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• all1: all records in the first snapshot</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• all2: all records in the second snapshot</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• shared: records that are in both snapshots</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• movedin: records that are in the first snapshot, but not the second</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• movedout: records that are in the second snapshot, but not the first</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A comma-separated list of sys_id values.</td>
</tr>
</tbody>
</table>

```javascript
var snapshot2 = PASnapshot.getCompareIDs('fb007202d7130100b96d45a3ce6103b4', 20160430, 20160531, 'shared');
gs.info(snapshot2);
```

Output: *** Script:
09c01200d7002100b81145a3ce6103ab,19c01200d7002100b81145a3ce6103e9,fcc01200d7002
....

PASnapshot - getCompareQuery(String sys_id, Number date1, Number date2, String type)

Get the query used to compare records in snapshots for a specified indicator at multiple dates.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The indicator sys_id.</td>
</tr>
<tr>
<td>date1</td>
<td>Number</td>
<td>The date of the first snapshot, in the format yyyymmdd.</td>
</tr>
<tr>
<td>date2</td>
<td>Number</td>
<td>The date of the second snapshot, in the format yyyymmdd.</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>Specifies what data to retrieve. Valid values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• all1: all records in the first snapshot</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• all2: all records in the second snapshot</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• shared: records that are in both snapshots</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• movedin: records that are in the second snapshot, but not the first</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• movedout: records that are in the first snapshot, but not the second</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The table, view, and encoded query as a JSON string.</td>
</tr>
</tbody>
</table>

```javascript
var snapshot4 = PASnapshot.getCompareQuery('fb007202d7130100b96d45a3ce6103b4', 20160530, 20160531, 'all1');
gs.info(snapshot4);
```

Output: *** Script: `{"view":null,"query":"sys_idINjavascript:new PAUtils().getCompareSnapshotIDs("fb007202d7130100b96d45a3ce6103b4","20160530","20160531","all1"),"table":"incident"}`

---

### PASnapshot - getIDs(String sys_id, Number date)

Get the sys_id values for all records contained in the snapshot for a specified indicator at the specified date.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The indicator sys_id.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>date</td>
<td>Number</td>
<td>The date when the snapshot was taken, in the format yyyymmdd.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A comma-separated list of sys_id values.</td>
</tr>
</tbody>
</table>

```javascript
var snapshot1 = PASnapshot.getIDs('fb007202d7130100b96d45a3ce6103b4', 20160530);
gs.info(snapshot1);
```

Output: *** Script:
09c01200d7002100b81145a3ce6103ab,19c01200d7002100b81145a3ce6103e9,fcc01200d7002 ....

PASnapshot - getQuery(String sys_id, Number date)

Get the query used to generate the snapshot for a specified indicator at the specified date.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The indicator sys_id.</td>
</tr>
<tr>
<td>date</td>
<td>Number</td>
<td>The date when the snapshot was taken, in the format yyyymmdd.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The table, view, and encoded query as a JSON string.</td>
</tr>
</tbody>
</table>

```javascript
var snapshot3 = PASnapshot.getQuery('fb007202d7130100b96d45a3ce6103b4', 20160530);
gs.info(snapshot3);
```
PDFGenerationAPI API - Scoped, Global

Provides support for PDF conversion and handling PDF fields.

This API is part of the ServiceNow PDF Generation Utilities plugin (com.snc.apppdfgenerator) and is provided within the sn_pdfgeneratorutils namespace. The plugin is activated by default.

These methods can also be used for documents created by non-catalog items. The methods in this class enable the following tasks:

- Dynamically generate a PDF from an HTML string and attach it to a record
- Fill fields in a PDF
- Sign a PDF
- Unflattened, flattened, or partially flattened
- Retrieving PDF field data

Related APIs:
- PdfMergeSignRequestor
- SVGToPDFConversionAPI

**PDFGenerationAPI – convertToPDF(String html, String targetTable, String targetTableSysId, String pdfName)**

Converts an HTML string to a PDF document.

To generate a PDF with header and footer information, such as page numbers, use `convertToPDFWithHeaderFooter()`.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>html</td>
<td>String</td>
<td>HTML to convert to a PDF document.</td>
</tr>
<tr>
<td>targetTable</td>
<td>String</td>
<td>Name of the table on which to attach the converted PDF.</td>
</tr>
<tr>
<td>targetTableSysId</td>
<td>String</td>
<td>Sys_id of the record on which to attach the converted PDF.</td>
</tr>
<tr>
<td>pdfName</td>
<td>String</td>
<td>Name to give the PDF.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Default: Sys_id of the PDF in the Attachments [sys_attachment] table.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing sys_id of the PDF attachment if conversion is successful, error message otherwise.</td>
</tr>
</tbody>
</table>

```
{
    "attachment_id": "String",
    "message": "String",
    "request_id": "String",
    "status": "String"
}
```

<Object>.attachment_id

If HTML conversion is successful, sys_id of the converted and attached PDF. The file is listed in the Attachments [sys_attachment] table.

Data type: String

<Object>.message

Message confirming success or error.

Possible values:

- Conversion failed. – No PDF created. Make sure the values provided are accurate.
- Conversion is successful. – The HTML successfully converted to PDF.
- Exception while reading Source document contents. PDF header not found. – Input attachment provided is not a valid PDF. Provide the correct attachment sys_id.
- Given target record [<tableName> - <targetTableSysId>] does not exist. – Target table sys_id is not in the table provided. Make sure you include the correct table name for the record.
- No Form associated with pdf to fill. attachmentSysId: <sys_id>
- No editable fields exist with specified names. Please check and try again. field names: <field names>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|      | • Request cannot proceed as the attachment with sys_id \{{{0}\}} did not pass security scan – The PDF did not pass the antivirus scan.  
• Request cannot proceed as the attachment with sys_id \{{{0}\}} is pending security scan – The PDF requires an antivirus scan.  
• Request completed successfully – Operation is successful.  
• Undefined – Sys_id provided does not exist or is not a PDF attachment.          |

- **Data type**: String

### `<Object>.request_id`
- Sys_id of the change producer request record.
- **Data type**: String

### `<Object>.status`
- Status indicating whether the operation is successful. Possible values:
  - success - Operation was successful.
  - failure – Operation was not successful. The `message` provides details.
- **Data type**: String

The following example shows how to convert HTML to a PDF and attach it to a record in the Incident [incident] table.

```javascript
var v = new sn_pdfgeneratorutils.PDFGenerationAPI;

// (Option) get HTML from the description field of an incident record
var gr = new GlideRecord("incident");
var html;

if (gr.get("<tableSysId=")) {
    html = gr.description.toString();
}
```
var result = v.convertToPDF(html, "incident", "<target_sys_id>", "myPDF");
gs.info(JSON.stringify(result));

Output:

{"attachment_id":"<sys_id>","message":"Conversion is successful.","request_id":"<change_sys_id>","status":"success"}

PDFGenerationAPI – convertToPDFWithHeaderFooter(String html, String targetTable, String targetTableSysId, String pdfName, Object headerFooterInfo)

Converts an HTML string into a PDF with header and footer content.

Use this method to generate PDFs with page settings:
- Header and footer information
- Margin sizes
- Orientation
- Enumeration
- Page size

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>html</td>
<td>String</td>
<td>HTML to convert to a PDF document.</td>
</tr>
<tr>
<td>targetTable</td>
<td>String</td>
<td>Name of the table on which to attach the converted PDF.</td>
</tr>
<tr>
<td>targetTableSysId</td>
<td>String</td>
<td>Sys_id of the record on which to attach the converted PDF.</td>
</tr>
<tr>
<td>pdfName</td>
<td>String</td>
<td>Name to give the PDF. Default: Sys_id of the PDF in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>headerFooterInfo</td>
<td>Object</td>
<td>Defines PDF header and footer details.</td>
</tr>
</tbody>
</table>

{ "FooterImageAlignment": "String", "FooterImageAttachmentId": "String", "FooterImageHeight": "String", "FooterText": "String", "FooterTextAlignment": "String", "GeneratePageNumber": "String" }
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>headerFooterInfo. FooterImageAlignment</td>
<td>String</td>
<td>Sets the image position in the footer. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BOTTOM_CENTER: Position the image in the bottom center of the footer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BOTTOM_LEFT: Position the image in the bottom left area of the footer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BOTTOM_RIGHT: Position the image in the bottom right area of the footer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TOP_CENTER: Position the image in the top center of the footer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TOP_LEFT: Position the image in the top left area of the footer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TOP_RIGHT: Position the image in the top right area of the footer.</td>
</tr>
<tr>
<td>headerFooterInfo. FooterImageAttachmentId</td>
<td>String</td>
<td>Sys_id of the footer image in the Attachments [sys_attachment] table. To determine if the file type is supported in your instance, Navigate to System Properties &gt; Security and check if it's listed in List of file extensions (comma-separated) that can be attached field.</td>
</tr>
<tr>
<td>headerFooterInfo. FooterImageHeight</td>
<td>String</td>
<td>Height of footer image. Default: 50 points</td>
</tr>
<tr>
<td>headerFooterInfo. FooterText</td>
<td>String</td>
<td>Footer text to place at the bottom of each PDF page.</td>
</tr>
</tbody>
</table>
# Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>headerFooterInfo.</td>
<td>String</td>
<td>Sets the text position in the footer.</td>
</tr>
<tr>
<td>FooterTextAlignment</td>
<td></td>
<td>Make sure this value does not match or conflict with the area provided in headerFooterInfo.FooterImageAlignment. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BOTTOM_CENTER: Position the text in the bottom center of the footer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BOTTOM_LEFT: Position the text in the bottom left area of the footer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• BOTTOM_RIGHT: Position the text in the bottom right area of the footer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TOP_CENTER: Position the text in the top center of the footer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TOP_LEFT: Position the text in the top left area of the footer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TOP_RIGHT: Position the text in the top right area of the footer.</td>
</tr>
<tr>
<td>headerFooterInfo.</td>
<td>String</td>
<td>Flag that indicates whether to generate a PDF page number.</td>
</tr>
<tr>
<td>GeneratePageNumber</td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Generate page numbers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not generate page numbers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td>headerFooterInfo.</td>
<td>String</td>
<td>Sets the image position in the header.</td>
</tr>
<tr>
<td>HeaderImageAlignment</td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• center: Position the image in the center of the header.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• left: Position the image on the left side of the header.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• right: Position the image on the right side of the header.</td>
</tr>
<tr>
<td>HeaderImageAttachmentId</td>
<td></td>
<td>To</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>determine if the file type is supported in your instance, Navigate to <strong>System Properties &gt; Security</strong> and check if it’s listed in <strong>List of file extensions (comma-separated) that can be attached</strong> field.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>headerFooterInfo.</td>
<td>String</td>
<td>Height of the header image.</td>
</tr>
<tr>
<td>HeaderImageHeight</td>
<td></td>
<td>Default: 50 points</td>
</tr>
<tr>
<td>headerFooterInfo.</td>
<td>String</td>
<td>Size of the left and right margins. If positioned in the left or right side of the page, header/footer details are placed within in this area.</td>
</tr>
<tr>
<td>LeftOrRightMargin</td>
<td></td>
<td>Default: 36 points</td>
</tr>
<tr>
<td>headerFooterInfo.</td>
<td>String</td>
<td>Page orientation.</td>
</tr>
<tr>
<td>PageOrientation</td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td>PageOrientation</td>
<td></td>
<td>• PORTRAIT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LANDSCAPE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Portrait</td>
</tr>
<tr>
<td>PageSize</td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td>PageSize</td>
<td></td>
<td>• A4 – 595 × 842 points</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LETTER – 612 × 792 points</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LEDGER – 792 x 1224 points</td>
</tr>
<tr>
<td>headerFooterInfo.</td>
<td>String</td>
<td>Size of the top and bottom margins. Header and footer details are placed within in this area.</td>
</tr>
<tr>
<td>TopOrBottomMargin</td>
<td></td>
<td>Default: 72 points</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing sys_id of the PDF attachment if conversion is successful, error message otherwise.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|                    | {  
|                    |    "attachment_id": "String",  
|                    |    "message": "String",  
|                    |    "request_id": "String",  
|                    |    "status": "String"  
|                    | }                                                                                                                                                                                                          |

**attachment_id**

If HTML conversion is successful, sys_id of the converted and attached PDF. The file is listed in the Attachments [sys_attachment] table.

Data type: String

**message**

Message confirming success or error.

Possible values:

- Conversion failed. – No PDF created. Make sure the values provided are accurate.
- Conversion is successful. – The HTML successfully converted to PDF.
- Footer Image alignment and text alignment cannot be in the same region with same alignment: 
  `<footerImageAlignment value>` – Make sure that `headerFooterInfo.FooterImageAlignment` and `headerFooterInfo.FooterTextAlignment` values are not in the same area.
- Exception while reading Source document contents. PDF header not found. – Input attachment provided is not a valid PDF. Provide the correct attachment sys_id.
- Given target record `[<tableName> - <targetTableSysId>]` does not exist. – Target table sys_id is not in the table provided. Make sure you include the correct table name for the record.
- Invalid footer image alignment: `<invalid_option>` is provided. – Provide a valid option in the `headerFooterInfo.FooterImageAlignment` property.
- Invalid footer text alignment: " + `<invalid_option>` + " is provided. – Provide a valid option in the `headerFooterInfo.footerTextAlignment` property.
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|      | • No Form associated with pdf to fill. attachmentSysId: `<sys_id>`  
      | • No editable fields exist with specified names. Please check and try again. field names: `<field names>`  
      | • Request cannot proceed as the attachment with sys_id `{0}` did not pass security scan – The PDF did not pass the antivirus scan.  
      | • Request cannot proceed as the attachment with sys_id `{0}` is pending security scan – The PDF requires an antivirus scan.  
      | • Request completed successfully – Operation is successful.  
      | • Unable to get the footer image. sysId: + `<value provided>` – Make sure the sys_id provided for headerFooterInfo.footerImageId is accurate.  
      | • Unable to get the header image. sysId: + `<value provided>` – Make sure the sys_id provided for headerFooterInfo.headerImageId is accurate.  
      | • Undefined – Sys_id provided does not exist or is not a PDF attachment.  |

Data type: String

`<Object>.request_id` Sys_id of the change producer request record.  
Data type: String

`<Object>.status` Status indicating whether the operation is successful. Possible values:  
• success - Operation was successful.  
• failure – Operation was not successful. The message provides details.  
Data type: String
The following example shows how to convert HTML to a PDF named "myPDF" and add the PDF as an attachment to a record in the Incident [incident] table. The PDF contains header and footer provided via attachment.

```javascript
var v = new sn_pdfgeneratorutils.PDFGenerationAPI;

// (Option) get HTML from the description field of an incident record
var gr = new GlideRecord("incident");
var html;
if (gr.get("<tableSysId>")) {
    html = gr.description.toString();
}

var hfInfo = new Object();
hfInfo["HeaderImageAttachmentId"] = "<hdrImgAttSysId>";
hfInfo["HeaderImageAlignment"] = "left";
hfInfo["FooterImageAttachmentId"] = "<ftrImgAttSysId>";
hfInfo["FooterImageAlignment"] = "TOP_CENTER";
hfInfo["FooterText"] = "Sample Footer Message";
hfInfo["PageSize"] = "A4";
hfInfo["GeneratePageNumber"] = "false";
hfInfo["TopOrBottomMargin"] = "36";
hfInfo["LeftOrRightMargin"] = "24";

var result = v.convertToPDFWithHeaderFooter(html, "incident", "<targetTbl_sys_id>",
"myPDF", hfInfo);
gs.info(JSON.stringify(result));
```

Output:

```javascript
{"attachment_id":"<sys_id>","message":"Conversion is successful.","request_id":"<change_sys_id>","status":"success"}
```

**PDFGenerationAPI – fillDocumentFieldsAndFlatten(Object fieldsMap, String sysId, String tableName, String tableSysId, String pdfName, Object flatten)**

Fills fields in an editable PDF, flattens the data fields, and attaches it to the provided record.

Use the following methods to determine if the PDF is fillable and get field information:

- `isDocumentFillable()`
- `getDocumentFields()`
- `getDocumentFieldsType()`
PDFGenerationAPI provides additional fill methods with different options:

- fillDocumentFields() – Fills fields in an editable PDF and attaches it to the provided record.
- fillFieldsAndMergeSignature() – Fills fields in an editable PDF, adds signature image, flattens the data fields, and attaches it to the provided record.
- getFilledDocumentWithSignatureAsBase64() – Fills fields in an editable PDF, creates an image, and converts it to a Base64-encoded PDF.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldsMap</td>
<td>Object</td>
<td>Optional. Key value map by PDF field name and value to fill. Use the getDocumentFields() method to get the list of available fields.</td>
</tr>
<tr>
<td>sysId</td>
<td>String</td>
<td>Sys_id of a PDF in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table containing the record to which the PDF is attached. You can find this value in the same row as the attachment listed in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>tableSysId</td>
<td>String</td>
<td>Sys_id of the record to which the PDF is attached. You can find this value in the same row as the attachment listed in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>pdfName</td>
<td>String</td>
<td>Name to give the PDF. Default: Sys_id of the PDF in the Attachments [sys_attachment] table.</td>
</tr>
</tbody>
</table>
| flatten         | Object| Optional. Flattening fields enable locking the fields so that other users cannot change the information. Specify the key as "FlattenType" and provide a flattening option as a string. Valid values:  
  - donot_flatten - Do not flatten any fields.  
  - partially_flatten - Flatten only the fields which are modified.  
  - fully_flatten - Flattens all the fields. |
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Default:</strong> fully_flatten</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;FlattenType&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Object</strong></td>
<td>Object containing sys_id of the updated PDF attachment if successful, error message otherwise.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
</tbody>
</table>
|      |   "attachment_id": "String",
|      |   "message": "String",
|      |   "status": "String"
|      | } |
| **<Object>.attachment_id** | If the operation is successful, sys_id of the filled PDF. The file is listed in the Attachments [sys_attachment] table. Data type: String |
| **<Object>.message** | Message confirming success or error. Valid values: |
|      | • Exception while reading Source document contents. PDF header not found. – Input attachment provided is not a valid PDF. Provide the correct attachment sys_id. |
|      | • Given target record [<tableName> - <targetTableSysId>] does not exist. – Target table sys_id is not in the table provided. Make sure you include the correct table name for the record. |
|      | • No Form associated with pdf to fill. attachmentSysId: <sys_id> |
|      | • No editable fields exist with specified names. Please check and try again. field names: <field names> |
|      | • Request cannot proceed as the attachment with sys_id [{0}] did not pass security scan – The PDF did not pass the antivirus scan. |
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>•</td>
<td>Request cannot proceed as the attachment with sys_id [{}0] is pending security scan – The PDF requires an antivirus scan.</td>
</tr>
<tr>
<td>•</td>
<td>Request completed successfully – Operation is successful.</td>
</tr>
<tr>
<td>•</td>
<td>Undefined – Sys_id provided does not exist or is not a PDF attachment.</td>
</tr>
</tbody>
</table>

Data type: String

`<Object>.status` Status indicating whether the operation is successful. Possible values:

- success - Operation was successful.
- failure – Operation was not successful. The `message` provides details.

Data type: String

The following example shows how to fill fields and flatten an editable PDF.

```javascript
var fieldMap = new Object();
fieldMap["Last Name First Name Middle Initial"] = "Tuter Abel E."
fieldMap["Date of Birth"] = "08101952"
fieldMap["US SSN"] = "111-22-9999"
fieldMap["Address"] = "PO Box 344"
fieldMap["City"] = "Jerome"
fieldMap["State"] = "AZ"
fieldMap["Zip"] = "86331"

var flatten = new Object();
flatten["FlattenType"] = "partially_flatten"

var v = new sn_pdfgeneratorutils.PDFGenerationAPI;
var result = v.fillDocumentFieldsAndFlatten(fieldMap, "<attachmentSysId>", "<tableName>", "<tableSysId>", "pdfName", flatten);
gs.info(JSON.stringify(result));
```

Output:

"attachment_id": "<sys_id>", "message": "Request completed successfully.", "status": "success"
PDFGenerationAPI – fillDocumentFields(Object fieldsMap, String sysId, String tableName, String tableSysId, String pdfName)

Fills fields in an editable PDF and attaches it to the provided record.

Use the following methods to determine if the PDF is fillable and get field information:

- isDocumentFillable()
- getDocumentFields()
- getDocumentFieldsType()

PDFGenerationAPI provides additional fill methods with different options:

- fillDocumentFieldsAndFlatten() – Fills fields in an editable PDF, flattens the data fields, and attaches it to the provided record.
- fillFieldsAndMergeSignature() – Fills fields in an editable PDF, adds signature image, flattens the data fields, and attaches it to the provided record.
- getFilledDocumentWithSignatureAsBase64() – Fills fields in an editable PDF, creates an image, and converts it to a Base64-encoded PDF.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldsMap</td>
<td>Object</td>
<td>Optional. Key value map by PDF field name and value to fill. Use the getDocumentFields() method to get the list of available fields.</td>
</tr>
<tr>
<td>sysId</td>
<td>String</td>
<td>Sys_id of a PDF in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table containing the record to which the PDF is attached. You can find this value in the same row as the attachment listed in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>tableSysId</td>
<td>String</td>
<td>Sys_id of the record to which the PDF is attached. You can find this value in the same row as the attachment listed in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>pdfName</td>
<td>String</td>
<td>Name to give the PDF. Default: Sys_id of the PDF in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Object</td>
<td>Object containing sys_id of the updated PDF attachment if successful, error message otherwise.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;attachment_id&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>}</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.attachment_id</td>
<td>If the operation is successful, sys_id of the filled PDF. The file is listed in the Attachments [sys_attachment] table. Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.message</td>
<td>Message confirming success or error. Valid values:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Exception while reading Source document contents. PDF header not found. – Input attachment provided is not a valid PDF. Provide the correct attachment sys_id.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Given target record [&lt;tableName&gt; - &lt;targetTableSysId&gt;] does not exist. – Target table sys_id is not in the table provided. Make sure you include the correct table name for the record.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• No Form associated with pdf to fill. attachmentSysId: &lt;sys_id&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• No editable fields exist with specified names. Please check and try again. field names: &lt;field names&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Request cannot proceed as the attachment with sys_id [{0}] did not pass security scan – The PDF did not pass the antivirus scan.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Request cannot proceed as the attachment with sys_id [{0}] is pending security scan – The PDF requires an antivirus scan.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Request completed successfully – Operation is successful.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Undefined – Sys_id provided does not exist or is not a PDF attachment.</td>
<td></td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.status</td>
<td>Status indicating whether the operation is successful. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• success - Operation was successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – Operation was not successful. The message provides details.</td>
</tr>
</tbody>
</table>

The following example shows how to fill fields in an editable PDF.

```javascript
var fieldMap = new Object();
fieldMap["Address"] = "Address value here";
fieldMap["State"] = "State value here";

var v = new sn_pdfgeneratorutils.PDFGenerationAPI;
var result = v.fillDocumentFields(fieldMap, "<attachmentSysId>", "<tableName>",
"<tableSysId>", "pdfName");
gs.info(JSON.stringify(result));
```

Output:

```json
{"attachment_id": "<sys_id>", "message": "Request completed successfully.", "status": "success"}
```

**PDFGenerationAPI – fillFieldsAndMergeSignature(Object fieldsMap, String sysId, String tableName, String tableSysId, String pdfName, PdfMergeSignRequestor requestor, Object flatten)**

Fills fields in an editable PDF, adds signature image, flattens the data fields, and attaches it to the provided record.

Use the following methods to determine if the PDF is fillable and get field information:

- isDocumentFillable()
- getDocumentFields()
- getDocumentFieldsType()

**PDFGenerationAPI** provides additional fill methods with different options:
- `fillDocumentFields()` – Fills fields in an editable PDF and attaches it to the provided record.
- `fillDocumentFieldsAndFlatten()` – Fills fields in an editable PDF, flattens the data fields, and attaches it to the provided record.
- `getFilledDocumentWithSignatureAsBase64()` – Fills fields in an editable PDF, creates an image, and converts it to a Base64-encoded PDF.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldsMap</td>
<td>Object</td>
<td>Optional. Key value map by PDF field name and value to fill. Use the <code>getDocumentFields()</code> method to get the list of available fields.</td>
</tr>
<tr>
<td>sysId</td>
<td>String</td>
<td>Sys_id of a PDF in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table containing the record to which the PDF is attached. You can find this value in the same row as the attachment listed in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>tableSysId</td>
<td>String</td>
<td>Sys_id of the record to which the PDF is attached. You can find this value in the same row as the attachment listed in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>pdfName</td>
<td>String</td>
<td>Name to give the PDF. Default: Sys_id of the PDF in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>requestor</td>
<td>PdfMergeSignRequestor</td>
<td>Signature input returned from <code>pdfMergeSignRequestor</code>.</td>
</tr>
</tbody>
</table>
| flatten       | Object        | Optional. Flattening fields enable locking the fields so that other users cannot change the information. Specify the key as "FlattenType" and provide a flattening option as a string. Valid values:  
  - donot_flatten - Do not flatten any fields.  
  - partially_flatten - Flatten only the fields which are modified.  
  - fully_flatten - Flattens all the fields. |
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Default: fully_flatten</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;FlattenType&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing sys_id of the updated PDF attachment if successful, error message otherwise.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;attachment_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>&lt;Object&gt;.attachment_id</td>
<td>If the operation is successful, sys_id of the filled PDF. The file is listed in the Attachments [sys_attachment] table. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.message</td>
<td>Message confirming success or error. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• Exception while reading Source document contents. PDF header not found. – Input attachment provided is not a valid PDF. Provide the correct attachment sys_id.</td>
</tr>
<tr>
<td></td>
<td>• Given target record [&lt;tableName&gt; - &lt;targetTableSysId&gt;] does not exist. – Target table sys_id is not in the table provided. Make sure you include the correct table name for the record.</td>
</tr>
<tr>
<td></td>
<td>• No Form associated with pdf to fill. attachmentSysId: &lt;sys_id&gt;</td>
</tr>
<tr>
<td></td>
<td>• No editable fields exist with specified names. Please check and try again. field names: &lt;field names&gt;</td>
</tr>
<tr>
<td></td>
<td>• Request cannot proceed as the attachment with sys_id [{0}] did not pass security scan – The PDF did not pass the antivirus scan.</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Request cannot proceed as the attachment with sys_id [{0}] is pending security scan – The PDF requires an antivirus scan.</td>
</tr>
<tr>
<td></td>
<td>• Request completed successfully – Operation is successful.</td>
</tr>
<tr>
<td></td>
<td>• Undefined – Sys_id provided does not exist or is not a PDF attachment.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>&lt;Object&gt;.status</th>
<th>Status indicating whether the operation is successful. Possible values:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• success - Operation was successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – Operation was not successful. The <strong>message</strong> provides details.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to fill fields with signature with default settings to completely flatten the fields.

```javascript
var fieldMap = new Object();
fieldMap["Address_Salutation"] = "Address value here";

var paramMap = new Object();
paramMap["FlattenType"] = "partially_flatten";

var requestor = new sn_pdfgeneratorutils.PdfMergeSignRequestor;
requestor.createRequest("<attachmentSysId>", "incident", "<tableSysId>", "filledPdf");
requestor.addSignatureMapping(6, 40, 50, 188, 44, "<signatureSysId>");

var v = new sn_pdfgeneratorutils.PDFGenerationAPI;
var result = v.fillFieldsAndMergeSignature(fieldMap, "<attachmentSysId>", "incident", "<tableSysId>", requestor, "filledPdf", paramMap);
gs.info(JSON.stringify(result));
```

Output:

```json
{"attachment_id":"5440d993bed3010d66be1191396194e","message":"Request completed successfully.","status":"success"}
```
PDFGenerationAPI – getDocumentFields(String sysId)

Gets a list of editable fields in a PDF document. Enables listing editable PDF fields without manually opening the file to check.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysId</td>
<td>String</td>
<td>Sys_id of a PDF in the Attachments [sys_attachment] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing ID of the signed PDF, error message otherwise.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;attachment_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>&lt;Object&gt;.fields</td>
<td>If the request is successful, list containing the name of each field in the PDF.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array of strings</td>
</tr>
<tr>
<td></td>
<td>&quot;fields&quot;: [&quot;field_name&quot;]</td>
</tr>
<tr>
<td>&lt;Object&gt;.message</td>
<td>Message confirming success or error. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Exception while reading Source document contents.</td>
</tr>
<tr>
<td></td>
<td>PDF header not found. – Input attachment provided is not a valid PDF.</td>
</tr>
<tr>
<td></td>
<td>Provide the correct attachment sys_id.</td>
</tr>
<tr>
<td></td>
<td>• Request cannot proceed as the attachment with sys_id [{0}] did not pass security scan – The PDF did not pass the antivirus scan.</td>
</tr>
<tr>
<td></td>
<td>• Request cannot proceed as the attachment with sys_id [{0}] is pending security scan – The PDF requires an antivirus scan.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Request completed successfully – Operation is successful.</td>
<td></td>
</tr>
<tr>
<td>• Undefined – Sys_id provided does not exist or is not a PDF attachment.</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

<Object>.status

Status indicating whether the operation is successful. Possible values:

• success - Operation was successful.
• failure – Operation was not successful. The message provides details.

Data type: String

The following example shows how to retrieve fields in a PDF attachment.

```javascript
var v = new sn_pdfgeneratorutils.PDFGenerationAPI;
var result = v.getDocumentFields("attachmentSysId");
gs.info(JSON.stringify(result));
```

Output:

```json
{"message":"Request completed successfully.","fields":["NP_formFillable","reset","print","1SSN","Signature.1","5sigDate","Check Box21"],"status":"success"}
```

PDFGenerationAPI – getDocumentFieldsType(String sysId)

Gets the field type of set of editable fields from a PDF document.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysId</td>
<td>String</td>
<td>Sys_id of a PDF in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Object</td>
<td>Object containing each PDF field type if successful, error message otherwise.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>`{</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;fields_type&quot;: {Object},</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.fields_type</td>
<td>Object listing each field in the specified PDF if successful, error message otherwise.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;fields_type&quot;: {</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;&lt;field type&gt;&quot;: {Object},</td>
<td></td>
</tr>
<tr>
<td></td>
<td>}</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.fields_type. &lt;field&gt;</td>
<td>Object containing page number of each field. The &lt;field&gt; name represents the field label, for example, &quot;SSN&quot;, or an automated label representing the type.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;&lt;field&gt;&quot;: {</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;fieldsDetails&quot;: [Array], // Check boxes, radio buttons, choice boxes only</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;pageNumber&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>}</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.fields_type. &lt;field&gt;.fieldsDetails</td>
<td>List of objects containing field name and corresponding value of each option for choice field types. Applicable types:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Check box</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Choice box</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Combo box</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Multi select choice box</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>“fieldsDetails”: [</td>
<td></td>
<td></td>
</tr>
<tr>
<td>“fieldName”: “String”,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>“value”: “String”</td>
<td></td>
<td></td>
</tr>
<tr>
<td>]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.fields_type.</td>
<td>Name of a choice field.</td>
<td></td>
</tr>
<tr>
<td>&lt;field&gt;.fieldsDetails.fieldName</td>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.fields_type.</td>
<td>Value of a choice field.</td>
<td></td>
</tr>
<tr>
<td>&lt;field&gt;.fieldsDetails.value</td>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.fields_type.</td>
<td>PDF page number corresponding to this field.</td>
<td></td>
</tr>
<tr>
<td>&lt;field&gt;.pageNumber</td>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.fields_type.</td>
<td>PDF field type.</td>
<td></td>
</tr>
<tr>
<td>&lt;field&gt;.type</td>
<td>Possible values:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• check_box</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• choice_box</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• combo_box</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• multi_select_choice_box</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• push_button</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• radio_button</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• signature</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• text</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.message</td>
<td>Message confirming success or error. Possible values:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Exception while reading Source document contents. PDF header not found. – Input attachment provided is not a valid PDF. Provide the correct attachment sys_id.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Request cannot proceed as the attachment with sys_id [{0}] did not pass security scan – The PDF did not pass the antivirus scan.</td>
<td></td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Request cannot proceed as the attachment with sys_id [{0}] is pending security scan – The PDF requires an antivirus scan.</td>
</tr>
<tr>
<td></td>
<td>• Request completed successfully – Operation is successful.</td>
</tr>
<tr>
<td></td>
<td>• Undefined – Sys_id provided does not exist or is not a PDF attachment.</td>
</tr>
</tbody>
</table>

Data type: String

<Object>.status

Status indicating whether the operation is successful. Possible values:

• success - Operation was successful.
• failure – Operation was not successful. The message provides details.

Data type: String

The following example shows how to retrieve field types in a PDF attachment. Results include manual returns for readability and are truncated for brevity.

```javascript
var v = new sn_pdfgeneratorutils.PDFGenerationAPI;
var result = v.getDocumentFieldsType("<attachmentSysId>");
gs.info(JSON.stringify(result));
```

Output:

```json
{ "fields_type":{ "1ADDLINE2.25":{ "pageNumber":2,"type":"text"}, "1ADDLINE2.24":{ "pageNumber":2,"type":"text"}, "1ADDLINE2.23":{ "pageNumber":2,"type":"text"}, "1ADDLINE2.22":{ "pageNumber":2,"type":"text"} , "1ADDLINE2.11":{ "pageNumber":2,"type":"text"}, 
"Check Box1":{ "fieldsDetails":{["fieldName":"Yes"]},"pageNumber":2,"type":"check_box"}, "4consentDate.6":{ "pageNumber":4,"type":"text"}, "4consentDate.7":{ "pageNumber":4,"type":"text"} }, "3SSN.9":{ "pageNumber":3,"type":"text"}, "3SSN.8":{ "pageNumber":3,"type":"text"}, "3SSN.7":{ "pageNumber":3,"type":"text"}, "pageNumber":2,"type":"check_box"}, "Check Box8":{ "fieldsDetails":{["fieldName":"Off"]},"pageNumber":2,"type":"check_box"}, "Check Box8":{ "fieldsDetails":{["fieldName":"yes"]},
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
PDFGenerationAPI – getFilledDocumentWithSignatureAsBase64(Object fieldsMap, String sysId, PdfMergeSignRequestor requestor, Object flatten)

Fills fields in an editable PDF, creates an image, and converts it to a Base64-encoded PDF.

Base64 encoding enables you to output a PDF as a string within a text document, such as HTML or JSON, without damaging the binary character syntax.

Use the following methods to determine if the PDF is fillable and get field information:

- isDocumentFillable()
- getDocumentFields()
- getDocumentFieldsType()

PDFGenerationAPI provides additional fill methods with different options:
• **fillDocumentFields()** – Fills fields in an editable PDF and attaches it to the provided record.

• **fillDocumentFieldsAndFlatten()** – Fills fields in an editable PDF, flattens the data fields, and attaches it to the provided record.

• **fillFieldsAndMergeSignature()** – Fills fields in an editable PDF, adds signature image, flattens the data fields, and attaches it to the provided record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldsMap</td>
<td>Object</td>
<td>Optional. Key value map by PDF field name and value to fill. Use the <code>getDocumentFields()</code> method to get the list of available fields.</td>
</tr>
<tr>
<td>sysId</td>
<td>String</td>
<td>Sys_id of a PDF in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>requestor</td>
<td>PdfMergeSign</td>
<td>Signature input returned from <code>pdfMergeSignRequestor</code>.</td>
</tr>
</tbody>
</table>
| flatten  | Object          | Optional. Flatting fields enable locking the fields so that other users cannot change the information. Specify the key as "FlattenType" and provide a flattening option as a string. Valid values:  
  • donot_flatten - Do not flatten any fields.  
  • partially_flatten - Flatten only the fields which are modified.  
  • fully_flatten - Flattens all the fields.  
  Default: fully_flatten |

```
{
  "FlattenType": "String"
}
```

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>If successful, PDF converted to Base64 format is added to the Attachments table [sys_attachment]. Contents</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>reflect the PDF attachment provided with fields and signature filled. The fields are not editable unless an alternative flattening option was provided with the <code>flatten</code> parameter.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.message</code></td>
<td>Message confirming success or error. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• Exception while reading Source document contents. PDF header not found. – Input attachment provided is not a valid PDF. Provide the correct attachment sys_id.</td>
</tr>
<tr>
<td></td>
<td>• Given target record <code>[&lt;tableName&gt; - &lt;targetTableSysId&gt;]</code> does not exist. – Target table sys_id is not in the table provided. Make sure you include the correct table name for the record.</td>
</tr>
<tr>
<td></td>
<td>• No Form associated with pdf to fill. attachmentSysId: <code>&lt;sys_id&gt;</code></td>
</tr>
<tr>
<td></td>
<td>• No editable fields exist with specified names. Please check and try again. field names: <code>&lt;field names&gt;</code></td>
</tr>
<tr>
<td></td>
<td>• Request cannot proceed as the attachment with sys_id <code>{0}</code> did not pass security scan – The PDF did not pass the antivirus scan.</td>
</tr>
<tr>
<td></td>
<td>• Request cannot proceed as the attachment with sys_id <code>{0}</code> is pending security scan – The PDF requires an antivirus scan.</td>
</tr>
<tr>
<td></td>
<td>• Request completed successfully – Operation is successful.</td>
</tr>
<tr>
<td></td>
<td>• Undefined – Sys_id provided does not exist or is not a PDF attachment.</td>
</tr>
<tr>
<td><code>Data type:</code></td>
<td>String</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.status</code></td>
<td>Status indicating whether the operation is successful. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• success - Operation was successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – Operation was not successful. The <code>message</code> provides details.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The following example shows how to load two fields in a PDF attachment, flatten the fields, and convert the PDF to Base64 format.

```javascript
var mymap = new Object();
mymap["City"] = "City value here";
mymap["State"] = "XX";

// create a requestor
var requestor = new sn_pdfgeneratorutils.PdfMergeSignRequestor;
requestor.createRequest("<sys_id>", "tableName", "<tableSysId>", "pdfName");
requestor.addSignatureMapping(6, 40, 50, 188, 44, "<signImgSysId>");
var processedRequestObj = requestor.processRequest();

var v = new sn_pdfgeneratorutils.PDFGenerationAPI;

var result = v.getFilledDocumentWithSignatureAsBase64(mymap, "<attachmentSysId>", processedRequestObj);
gs.info (JSON.stringify(result));
```

PDFGenerationAPI – `getPdfPageSizes(String sysId)`

Gets the page size of a PDF document.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysId</td>
<td>String</td>
<td>Sys_id of a PDF in the Attachments [sys_attachment] table.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing the size of each page if successful, error message otherwise.</td>
</tr>
</tbody>
</table>

```javascript
{
  "pages_size": {Object},
  "message": "String",
}
```
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;status&quot;: &quot;String&quot;</td>
<td></td>
</tr>
</tbody>
</table>

**<Object>.pages_size**

If the operation is successful, width and height of each PDF page in points. The page number is returned as a string and the measurement values are returned as number data types.

Data type: Object

```
"pages_size": {"<page number>": [width, height]}
```

**<Object>.message**

Message confirming success or error. Possible values:

- Request cannot proceed as the attachment with sys_id [0] did not pass security scan – The PDF did not pass the antivirus scan.
- Request cannot proceed as the attachment with sys_id [0] is pending security scan – The PDF requires an antivirus scan.
- Request completed successfully – Operation is successful.
- Undefined – Sys_id provided does not exist or is not a PDF attachment.

Data type: String

**<Object>.status**

Status indicating whether the operation is successful. Possible values:

- success - Operation was successful.
- failure – Operation was not successful. The message provides details.

Data type: String

The following example shows how to display the width and height of each page in a PDF attachment.

```javascript
var v = new sn_pdfgeneratorutils.PDFGenerationAPI;
var result = v.getPdfPageSizes (<attachmentSysId>);
```
gs.info(JSON.stringify(result));

Output:

```json
{
  "pages_size": {
    "1": [612, 792],
    "2": [612, 792],
    "3": [612, 792],
    "4": [612, 792],
    "5": [612, 792]
  },
  "message": "Request completed successfully.",
  "status": "success"
}
```

**PDFGenerationAPI – isDocumentFillable(String sysId)**

Checks if the PDF document contains editable fields.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysId</td>
<td>String</td>
<td>Sys_id of a PDF in the Attachments [sys_attachment] table.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing the size of each page if successful, error message otherwise.</td>
</tr>
</tbody>
</table>

```json
{
  "document_editable": "String",
  "message": "String",
  "status": "String"
}
```

- `<Object>.document_editable`
  - If the operation is successful, flag indicating whether the document is editable.
  - Valid values:
    - true: PDF document has editable fields.
    - false: PDF document does not have editable fields.
  - Data type: Boolean value provided as a string

- `<Object>.message`
  - Message confirming success or error.
  - Possible values:
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Exception while reading Source document contents. PDF header not found. – Input attachment provided is not a valid PDF. Provide the correct attachment sys_id.</td>
<td></td>
</tr>
<tr>
<td>• Request cannot proceed as the attachment with sys_id [{0}] did not pass security scan – The PDF did not pass the antivirus scan.</td>
<td></td>
</tr>
<tr>
<td>• Request cannot proceed as the attachment with sys_id [{0}] is pending security scan – The PDF requires an antivirus scan.</td>
<td></td>
</tr>
<tr>
<td>• Request completed successfully – Operation is successful.</td>
<td></td>
</tr>
<tr>
<td>• Undefined – Sys_id provided does not exist or is not a PDF attachment.</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

<Object>.status
Status indicating whether the operation is successful. Possible values:
• success - Operation was successful.
• failure – Operation was not successful. The message provides details.

Data type: String

The following example shows how to determine if PDF document fields are editable.

```javascript
var v = new sn_pdfgeneratorutils.PDFGenerationAPI;
var result = v.isDocumentFillable("<attachmentSysId>");
gs.info(JSON.stringify(result));
```

Output:

```javascript
{"message":"Request completed successfully.","document_editable":"true","status":"success"}
```

PDFGenerationAPI – PDFGenerationAPI()
Instantiates a new PDFGenerationAPI object.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to create a PDFGenerationAPI object.

```javascript
var v = new sn_pdfgeneratorutils.PDFGenerationAPI;
```

**PdfMergeSignRequestor API - Scoped, Global**

Adds an image representing a signature to a PDF document.

This API is part of the ServiceNow PDF Generation Utilities plugin (com.snc.apppdfgenerator) and is provided within the sn_pdfgeneratorutils namespace. The plugin is activated by default.

This method creates a signature object that can be implemented in a PDF using PDFGenerationAPI.

**PdfMergeSignRequestor - PdfMergeSignRequestor()**

Instatiates a new PdfMergeSignRequestor object.

The following example shows how to create a PdfMergeSignRequestor object.

```javascript
var v = new sn_pdfgeneratorutils.PdfMergeSignRequestor;
```

**PdfMergeSignRequestor - addSignatureMapping(Number pageNumber, Number leftMargin, Number topMargin, Number boxWidth, Number boxHeight, String sysId)**

Assigns signature size and position requirements in the PDF.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pageNumber</td>
<td>Number</td>
<td>Number of the page on which to insert the signature.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>leftMargin</td>
<td>Number</td>
<td>Value in points representing the left margin area of the page at which to insert the signature.</td>
</tr>
<tr>
<td>topMargin</td>
<td>Number</td>
<td>Value in points representing the top margin area of the page at which to insert the signature image.</td>
</tr>
<tr>
<td>boxWidth</td>
<td>Number</td>
<td>Value in points representing width of the box to contain the signature.</td>
</tr>
<tr>
<td>boxHeight</td>
<td>Number</td>
<td>Value in points representing height of the box to contain the signature image.</td>
</tr>
<tr>
<td>sysId</td>
<td>String</td>
<td>Sys_id of the signature image in the Attachments [sys_attachment] table.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add the signature mapping. For a complete example, see `processRequest()`.

```javascript
var requestor = new sn_pdfgeneratorutils.PdfMergeSignRequestor;

// For the purpose of this example, set signature sizes and page number for signature placement
var page = 2;
var leftMargin = 48;
var topMargin = 60;
var signatureWidth = 96;
var signatureHeight = 36;

requestor.addSignatureMapping(page, leftMargin, topMargin, signatureWidth, signatureHeight, "<signatureSysId>");
```

**PdfMergeSignRequestor - createRequest(String targetSysId, String targetTable, String tableSysId, String targetFileName)**

Creates a signature request with source and target inputs.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>targetSysId</td>
<td>String</td>
<td>Sys_id of a PDF in the Attachments [sys_attachment] table. Use this value as the target PDF on which to add a signature.</td>
</tr>
<tr>
<td>targetTable</td>
<td>String</td>
<td>Name of the table containing the record to which the PDF is attached. You can find this value in the same row as the attachment listed in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>tableSysId</td>
<td>String</td>
<td>Sys_id of the record to which the PDF is attached. You can find this value in the same row as the attachment listed in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>targetFileName</td>
<td>String</td>
<td>Name of the target PDF without extension.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to create a signature request. For a complete example, see `processRequest()`.

```javascript
var requestor = new sn_pdfgeneratorutils.PdfMergeSignRequestor;
requestor.createRequest("<sys_id>", "tableName", "<tableSysId>", "pdfFileName");
```

**PdfMergeSignRequestor -processRequest()**

Processes requests and adds the signatures.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing the size of each page if successful, error message otherwise.</td>
</tr>
</tbody>
</table>

```json
{
    "attachment_id": "String",
    "message": "String",
    "status": "String"
}
```

**<Object>.attachment_id**

If the request is successful, sys_id of the signed and attached PDF. The file is listed in the Attachments [sys_attachment] table.

Data type: String

**<Object>.message**

Possible values:

- Request cannot proceed as the attachment with sys_id [{0}] did not pass security scan – The PDF did not pass the antivirus scan.

- No signature mapping specified. Cannot process this request – Provide signature mapping using the addSignatureMapping() method.

- Request completed successfully.

- Request failure. Exceptions while trying to add signatures to document. Please check again.

- This request cannot be completed as the requested page does not exist. page No: <page number> 

Data type: String

**<Object>.status**

Status indicating whether the operation is successful. Possible values:

- success - Operation was successful.

- failure – Operation was not successful. The message provides details.

Data type: String
The following example shows how to process the signature request.

```javascript
var requestor = new sn_pdfgeneratorutils.PdfMergeSignRequestor;

requestor.createRequest("<sys_id>", "tableName", "<tableSysId>", pdfFileName);

// For the purpose of this example, set signature sizes and page number for signature placement
var page = 6;
var leftMargin = 40;
var topMargin = 50;
var signatureWidth = 188;
var signatureHeight = 44;

requestor.addSignatureMapping(page, leftMargin, topMargin, signatureWidth, signatureHeight,
   "<signatureSysId>" OTHERWISE

var result = requestor.processRequest();
gs.info(JSON.stringify(result));
```

### PdfPage - Scoped, Global

Creates a PdfPage object representing a PDF page and its attributes; such as size, width, and color.

This API is part of the ServiceNow PDF Generation Utilities plugin (com.snc.apppdfgenerator) and is provided within the sn_pdfgeneratorutils namespace. The plugin is activated by default.

This API is a component used with the Document API to generate a PDF.

#### PdfPage - PdfPage(String pageSize, String orientation)

Instantiates a new PdfPage object.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>pageSize</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
The following example shows how to create a `PdfPage` object.

```javascript
var pdfpage = new sn_pdfgeneratorutils.PdfPage("A4", "LANDSCAPE");
```

**PdfPage – getBottom()**

Gets the Y-coordinate of the lower edge of a PDF page.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Decimal value of the Y-coordinate at the lower edge of the PDF page.</td>
</tr>
</tbody>
</table>

The following example shows how to get the Y-coordinate of the lower edge of a PDF page.

```javascript
var pdfpage = new sn_pdfgeneratorutils.PdfPage("A4", "LANDSCAPE");
var bottom = pdfpage.getBottom();
```

**PdfPage – getLeft()**

Gets the X-coordinate of the left edge of a PDF page.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Decimal value of the X-coordinate on the left edge of the page.</td>
</tr>
</tbody>
</table>

The following example shows how to get the X-coordinate of the left edge of a PDF page.

```javascript
var pdfpage = new sn_pdfgeneratorutils.PdfPage("A4", "LANDSCAPE");
var left = pdfpage.getLeft();
```

**PdfPage – getPdfPageSize()**

Gets the size of a PDF page

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to get the PDF page size.

```javascript
var pdfpage = new sn_pdfgeneratorutils.PdfPage("A4", "LANDSCAPE");
var size = pdfpage.getPdfPageSize();
```

**PdfPage – getRight()**

Gets the X-coordinate of the right edge of a PDF page.
The following example shows how to get the X-coordinate of the right edge of a PDF page.

```javascript
var pdfpage = new sn_pdfgeneratorutils.PdfPage("A4", "LANDSCAPE");
var right = pdfpage.getRight();
```

**PdfPage – getTop()**

Gets the Y-coordinate of the upper edge of a PDF page.

The following example shows how to get the Y-coordinate of the upper edge of a PDF page.

```javascript
var pdfpage = new sn_pdfgeneratorutils.PdfPage("A4", "LANDSCAPE");
var top = pdfpage.getTop();
```

**PdfPage – getWidth()**

Gets the width of a PDF page.
The following example shows how to retrieve the width of a PDF page.

```javascript
var pdfPage = new sn_pdfgeneratorutils.PdfPage("A4", "LANDSCAPE");
var width = PdfPage.getWidth();
```

### PlaybookExperience API - Scoped

Provides methods for handling playbook executions.

The PlaybookExperience API requires the Playbook Experience Core plugin (com.glide.playbook_experience.config) and is provided within the sn_playbook namespace.

This API requires at least one playbook in the Process Definition [sys_pd_process_definition] table. To use this API, you must have the roles required to view and cancel a running process in Process Automation Designer. For more information, see Process Automation Designer.

#### PlaybookExperience - cancelPlaybooksByParentRecord(GlideRecord parentRecord, String cancellationReason, String scopedName, String playbookExperienceId)

Cancels playbook executions for a given parent record.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parentRecord</td>
<td>GlideRecord</td>
<td>The parent record to cancel playbook executions for. The parent record can be any record that has playbook executions, such as an interaction record or an onboarding case record.</td>
</tr>
<tr>
<td>cancellationReason</td>
<td>String</td>
<td>The reason for cancelling the playbook executions.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopedName</td>
<td>String</td>
<td>Optional. The scoped name of the playbook to cancel. The scoped name is from the Process Definition [sys_pd_process_definition] table in the format <code>scope.name</code>. If provided, only executions of this playbook are cancelled for the given parent record. If not provided, all executions of all playbooks are cancelled for the given parent record.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing the cancelled playbook executions and any skipped playbook executions that were unable to be cancelled.</td>
</tr>
<tr>
<td></td>
<td>{ &quot;canceledPlaybookContext&quot;: [Array], &quot;skippedPlaybookContext&quot;: [Array] }</td>
</tr>
</tbody>
</table>

Object `.canceledPlaybookContext` is an array of cancelled playbook executions. Each playbook execution is an object in the array. Data type: Array

```json
"canceledPlaybookContext": [ 
  { 
    "can_read": Boolean, 
    "canceled_by": "String", 
    "cancellation_reason": "String", 
    "errors": [Array],
  }
]```
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|                                    | ```
|                                   | "parent_record": "String",
|                                   | "parent_table": "String",
|                                   | "playbook_id": "String",
|                                   | "playbook_table": "String",
|                                   | "scoped_name": "String",
|                                   | "state": {Object},
|                                   | "sys_id": "String",
|                                   | "title": "String"
|                                   | ```

---

<Object>.canceledPlaybookContext.can_read
Flag that indicates whether the current user can read the playbook execution record. The current user must have read access to the parent record to be able to read the playbook execution record.

Valid values:

- true: Current user has read access to the playbook execution record.
- false: Current user doesn’t have read access to the playbook execution record.

Data type: Boolean

<Object>.canceledPlaybookContext.canceled_by
User ID of the user who cancelled the playbook execution.
Data type: String

<Object>.canceledPlaybookContext.cancellation_reason
The cancellation reason input by the user who cancelled the playbook execution.
Data type: String

<Object>.canceledPlaybookContext.errors
List of cancellation errors. Each error is an object in the array.
Data type: Array

 ```
<table>
<thead>
<tr>
<th>errors</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
| ```

Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.canceledPlaybookContext.errors.message</td>
<td>The error message. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.canceledPlaybookContext.errors.type</td>
<td>The type of error. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.canceledPlaybookContext.parent_record</td>
<td>Sys_id of the parent record that playbook executions were cancelled for. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.canceledPlaybookContext.parent_table</td>
<td>The name of the table that the parent record is from. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.canceledPlaybookContext.playbook_table</td>
<td>The name of the table that the playbook is from, generally the Process Definitions [sys_pd_process_definition] table. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.canceledPlaybookContext.playbook_context</td>
<td>The playbook from the Process Definitions [sys_pd_process_definition] table in the format scope.name. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.canceledPlaybookContext.playbook_execution</td>
<td>The playbook execution from the Process Executions [sys_pd_context] table. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;state&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;displayValue&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>&lt;Object&gt;.canceledPlaybookContext.state.displayValue</td>
<td>The display value of the playbook execution state. Data type: String</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.canceledPlaybookContext.state.value</td>
<td>The value of the playbook execution state. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.canceledPlaybookContext.sys_id</td>
<td>Sys_id of the playbook execution from the Process Executions [sys_pd_context] table. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.canceledPlaybookContext.title</td>
<td>The label of the playbook execution from the Process Executions [sys_pd_context] table. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.skippedPlaybookContext</td>
<td>List of the skipped playbook executions. Each playbook execution is an object in the array. For descriptions of the object properties, see the <code>canceledPlaybookContext</code> array. Data type: Array</td>
</tr>
</tbody>
</table>

This example shows how to cancel all executions of a specific playbook (in this case, the Playbook Experience Demo) for a given interaction record. To use this method in a UI action or business rule, pass the `current` object as the `parentRecord` instead.

```javascript
var parentRecord = new GlideRecordUtil().getGR("interaction", "d91742531b343010a26c98a1b24bcbe0");
```
```javascript
var cancellationReason = "Cancelling this playbook";

// demo playbook from Process Automation Experience Demo store app
var scopedName = "sn_pad_demo.playbook_experience_demo";

// demo playbook experience from Process Automation Experience Demo store app
var playbookExperienceId = "a56d8d93ff311010cc0853ea793bf1a6";

var cancelPlaybookReturn = 
    sn_playbook.PlaybookExperience.cancelPlaybooksByParentRecord(parentRecord, 
    cancellationReason, scopedName, playbookExperienceId);

gs.info(JSON.stringify(cancelPlaybookReturn, null, 2));

Output:

{
    "canceledPlaybookContext": [
    {
        "can_read": true,
        "sys_id": "d02782533d343010ac50ee17e75d3466",
        "scoped_name": "sn_pad_demo.playbook_experience_demo",
        "canceled_by": "admin",
        "playbook_table": "sys_pd_process_definition",
        "state": {
            "displayValue": "Pending Cancel",
            "value": "PENDING_CANCEL"
        },
        "title": "Playbook Experience Demo",
        "parent_record": "d91742531b343010a26c98a1b24bcbe0",
        "playbook_id": "0d35ee1807301010cc08d9630ad3002a",
        "cancellation_reason": "Cancelling this playbook",
        "parent_table": "interaction",
        "errors": []
    }
    ],
    "skippedPlaybookContext": []
}
```

**PlaybookExperience - getPlaybooksForParentRecord(GlideRecord parentRecord)**

Gets a list of playbook executions for a given parent record.
# Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parentRecord</td>
<td>GlideRecord</td>
<td>The parent record to get playbook executions for. The parent record can be any record that can have playbook executions, such as an interaction record or an onboarding case record.</td>
</tr>
</tbody>
</table>

# Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of playbook executions for the parent record. Each playbook execution is an object in the array.</td>
</tr>
<tr>
<td>&lt;Array&gt;.can_read</td>
<td>Flag that indicates whether the current user can read the playbook execution record. The current user must have read access to the parent record to be able to read the playbook execution record.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Current user has read access to the playbook execution record.</td>
</tr>
<tr>
<td></td>
<td>• false: Current user doesn’t have read access to the playbook execution record.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Returns (continued)</td>
</tr>
</tbody>
</table>
| <Array>.canceled_by | The User ID of the user who cancelled the playbook execution. Empty if the playbook is not cancelled.  
Data type: String |
| <Array>.cancellation_reason | The cancellation reason input by the user who cancelled the playbook execution. Empty if the playbook is not cancelled.  
Data type: String |
| <Array>.errors | List of errors. Each error is an object in the array.  
Data type: Array |
| <Array>.errors.message | The error message.  
Data type: String |
| <Array>.errors.type | The type of error.  
Data type: String |
| <Array>.parent_record | Sys_id of the parent record.  
Data type: String |
| <Array>.parent_table | The name of the table that the parent record is from.  
Data type: String |
Data type: String |
| <Array>.playbook_table | The name of the table that the playbook is from, generally the Process Definitions [sys_pd_process_definition] table.  
Data type: String |
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Array&gt;.scoped_name</td>
<td>The scoped name of the playbook from the Process Definitions [sys_pd_process_definition] table in the format scope.name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Array&gt;.state</td>
<td>The state of the playbook execution from the Process Executions [sys_pd_context] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;state&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;displayValue&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>&lt;Array&gt;.state.displayValue</td>
<td>The display value of the playbook execution state.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Array&gt;.state.value</td>
<td>The value of the playbook execution state.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Array&gt;.title</td>
<td>The label of the playbook execution from the Process Executions [sys_pd_context] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

This example shows how to get playbook executions for a given interaction record. To use this method in a UI action or business rule, pass the current object as the parentRecord instead.

```javascript
var parentRecord = new GlideRecordUtil().getGR("interaction", "148776e5818d7410f87701eb89fdc824");
var playbook = sn_playbook.PlaybookExperience.getPlaybooksForParentRecord(parentRecord);
gs.info(JSON.stringify(playbook, null, 2));
```

**Output:**

```javascript
[
]
```
PlaybookExperience - parentRecordContainsPlaybook(GlideRecord parentRecord, String scopedName)

Checks whether a parent record has playbook executions.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parentRecord</td>
<td>GlideRecord</td>
<td>The parent record to check for playbook executions. The parent record can be any record that can have playbook executions, such as an interaction record or an onboarding case record.</td>
</tr>
<tr>
<td>scopedName</td>
<td>String</td>
<td>Optional. The scoped name of the playbook to check for. The scoped name is from the Process Definition [sys_pd_process_definition] table in the format scope.name. If provided, only executions of this playbook are checked for. If not provided, executions of all playbooks are checked for.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the parent record has playbook executions.</td>
</tr>
</tbody>
</table>
This example shows how to check if a given interaction record has executions of a specific playbook (in this case, the Playbook Experience Demo). To use this method in a UI action or business rule, pass the `current` object as the `parentRecord` instead.

```javascript
var parentRecord = new GlideRecordUtil().getGR("interaction", "148776e5818d7410f87701eb89fdc824");

// demo playbook from Process Automation Experience Demo store app
var scopedName = "sn_pad_demo.playbook_experience_demo";

var hasPlaybooks = sn_playbook.PlaybookExperience.parentRecordContainsPlaybook(parentRecord, scopedName);
gs.info(hasPlaybooks);
```

Output:

```
true
```

**PlaybookExperience - triggerPlaybook(String scopedName, GlideRecord parentRecord)**

Initiates a playbook for a parent record.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopedName</td>
<td>String</td>
<td>The scoped name of the playbook to initiate. The scoped name is from the Process Definition [sys_pd_process_definition] table in the format scope.name.</td>
</tr>
<tr>
<td>parentRecord</td>
<td>GlideRecord</td>
<td>The parent record to initiate a playbook for. The parent record can be any record that can have playbook executions, such as an interaction record or an onboarding case record.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the playbook execution from the Process Executions [sys_pd_context] table that was created for the parent record. Null if a playbook execution wasn't successfully created.</td>
</tr>
</tbody>
</table>

This example shows how to initiate a playbook for a given interaction record. To use this method in a UI action or business rule, pass the current object as the parentRecord instead.

```javascript
var parentRecord = new GlideRecordUtil().getGR("interaction", "148776e5818d7410f87701eb89f0dc824");

// demo playbook from Process Automation Experience Demo store app
var scopedName = "sn_pad_demo.playbook_experience_demo";

var playbookExecution = sn_playbook.PlaybookExperience.triggerPlaybook(scopedName, parentRecord);
gs.info(playbookExecution);
```

Output:

```
f059958267c02d410952864f0fed358cc
```

**PredictabilityEstimate - Global**

Scriptable object used in Predictive Intelligence stores. This object provides estimation of how predictable fields of a dataset can be, and which features can be useful for predicting those fields.

The **PredictabilityEstimate** API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the **sn_ml** namespace.

The predictability estimate setup-to-training flow is as follows:

1. Create a dataset using the **DatasetDefinition** API.
2. Use the **constructor** to create a predictability estimate object.
3. Add the predictability estimate object to the predictability estimate store using the **PredictabilityEstimateStore - add()** method.
4. Train the predictability estimate using the `submitTrainingJob()` method. This creates a version of the object that you can manage using the `PredictabilityEstimateVersion` API.

5. Get estimated predictive values using the `PredictabilityEstimateVersion - getResults()` method.

ℹ️ **Note:** This API runs with full privileges. To restrict user access, include an access control mechanism in the script.

For usage guidelines, refer to **Using ML APIs**.

**PredictabilityEstimate - PredictabilityEstimate(Object config)**

Creates a predictability estimate.

To get new predictability estimates on the same dataset, use this constructor to create a new `PredictabilityEstimate` object with a unique name.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>Object</td>
<td>JavaScript object containing configuration properties of the predictability estimate.</td>
</tr>
<tr>
<td>config.dataset</td>
<td>Object</td>
<td><code>DatasetDefinition</code> name.</td>
</tr>
<tr>
<td>config.domainName</td>
<td>String</td>
<td>Optional. Domain name associated with this dataset. See <strong>Domain separation and Predictive Intelligence</strong>. Default: Current domain, for example, &quot;global&quot;.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config.inputFieldNames</td>
<td>Array</td>
<td>List of candidate input fields as strings to be considered for estimation.</td>
</tr>
<tr>
<td>config.label</td>
<td>String</td>
<td>Identifies the prediction task.</td>
</tr>
<tr>
<td>config.minRowCount</td>
<td>String</td>
<td>Optional. Minimum number of records required in the dataset for training. Default: 10000</td>
</tr>
<tr>
<td>config.predictedFieldName</td>
<td>String</td>
<td>Identifies a field to be trained for predictability.</td>
</tr>
<tr>
<td>config.trainingFrequency</td>
<td>String</td>
<td>Optional. The frequency to retrain the model. Possible values: every_30_days, every_60_days, every_90_days, every_120_days, every_180_days, run_once. Default: run_once</td>
</tr>
</tbody>
</table>

The following example shows how to create an estimation job and add it to the PredictabilityEstimate store.

```javascript
var myIncidentData = new sn_ml.DatasetDefinition({
  'tableName' : 'incident',
  'encodedQuery' : 'activeANYTHING'
});

var myEstimate = new sn_ml.PredictabilityEstimate({
  'label': "predictability estimate",
  'dataset' : myIncidentData,
  'inputFieldNames': ['short_description'],
  'predictedFieldName': 'category'
});
```
PredictabilityEstimate - cancelTrainingJob()
Cancels a job for a predictability estimate object that has been submitted for training.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to cancel an existing training job.

```javascript
var myEstimate =
    sn_ml.PredictabilityEstimateStore.get('ml_sn_global_global_predictability_estimate');
myEstimate.cancelTrainingJob();
```

PredictabilityEstimate - getActiveVersion()
Gets the active `PredictabilityEstimateVersion` object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Active <code>PredictabilityEstimateVersion</code> object.</td>
</tr>
</tbody>
</table>

The following example shows how to get an active `PredictabilityEstimate` version from the store and return its training status.
var mlEstimate =
    sn_ml.PredictabilityEstimateStore.get('ml_x_snc_global_global_predictability_estimate');

gs.print(JSON.stringify(JSON.parse(mlEstimate.getActiveVersion().getStatus()), null, 2));

Output:

```
{
    "state": "predictability_estimate_complete",
    "percentComplete": "100",
    "hasJobEnded": "true"
}
```

**PredictabilityEstimate - getAllVersions()**

Gets all versions of a predictability estimate.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Existing versions of a solution object. See also PredictabilityEstimateVersion API.</td>
</tr>
</tbody>
</table>

The following example shows how to get all PredictabilityEstimate version objects and call the getVersionNumber() and getStatus() estimate version methods on them.

```javascript
var mlEstimate =
    sn_ml.PredictabilityEstimateStore.get('ml_x_snc_global_global_predictability_estimate');

var mlEstimateVersions = mlEstimate.getAllVersions();

for (i = 0; i < mlEstimateVersions.length; i++) {
    gs.print("Version " + mlEstimateVersions[i].getVersionNumber() + " Status: " + mlEstimateVersions[i].getStatus() +"\n");
}
```

Output:
PredictabilityEstimate - getLatestVersion()

Gets the latest version of a predictability estimate.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>PredictabilityEstimateVersion object corresponding to the latest version of a PredictabilityEstimate().</td>
</tr>
</tbody>
</table>

The following example shows how to get the latest version of a predictability estimate and return its training status.

```javascript
var mlEstimate = 
sn_ml.PredictabilityEstimateStore.get('ml_x_snc_global_global_predictability_estimate');
gs.print(JSON.stringify(JSON.parse(mlEstimate.getLatestVersion().getStatus()), null, 2));
```

Output:

```javascript
{
  "state": "predictability_estimate_complete",
  "percentComplete": "100",
  "hasJobEnded": "true"
}
```

PredictabilityEstimate - getName()

Gets the name of the object to use for interaction with the store.
The following example shows how to update `PredictabilityEstimate` dataset information and print the name of the object.

```javascript
// Update estimate
var myIncidentData = new sn_ml.DatasetDefinition({
  'tableName' : 'incident',
  'fieldNames' : ['category', 'short_description', 'priority'],
  'encodedQuery' : 'activeANYTHING'
});

var myEstimate = new sn_ml.PredictabilityEstimate({
  'label': "my estimate",
  'dataset' : myIncidentData,
  'inputFieldNames':['short_description'],
  'predictedFieldName': 'category'
});

// update estimate
sn_ml.PredictabilityEstimateStore.update('ml_x_snc_global_global_my_definition_4', myEstimate);

// print estimate name
gs.print('Estimate Name: '+myEstimate.getName());
```

Output:

```
Estimate Name: ml_x_snc_global_global_my_definition_4
```

**PredictabilityEstimate - getProperties()**

Gets predictability estimate object properties.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Contents of the Dataset and PredictabilityEstimate() object details in the PredictabilityEstimateStore</td>
</tr>
</tbody>
</table>

```json
{
    "datasetProperties": {
        "encodedQuery": "String",
        "fieldDetails": [Array],
        "fieldNames": [Array],
        "tableName": "String"
    }
}
```

```json
<Object>.datasetProperties
```

Lists the properties of the DatasetDefinition() object associated with the estimate.

```json
{
    "encodedQuery": "String",
    "fieldDetails": [Array],
    "fieldNames": [Array],
    "tableName": "String"
}
```

```json
<Object>.datasetProperties$tableName
```

Name of the table for the dataset. For example, "tableName": "Incident".

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;.datasetProperties.fieldNames</code></td>
<td>List of field names from the specified table as strings. For example, <code>&quot;fieldNames&quot;: [&quot;short_description&quot;, &quot;priority&quot;]</code>. Data type: Array.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.datasetProperties.fieldNames.fieldDetails</code></td>
<td>List of JavaScript objects that specify field properties. Data type: Array.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.datasetProperties.fieldNames.fieldDetails.&lt;object&gt;.name</code></td>
<td>Name of the field defining the type of information to restrict this dataset to. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.datasetProperties.fieldNames.fieldDetails.&lt;object&gt;.type</code></td>
<td>Machine-learning field type. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.datasetProperties.fieldDetails.encodedQuery</code></td>
<td>Encoded query string in standard Glide format. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.domainName</code></td>
<td>Domain name associated with this dataset. See Domain separation and Predictive Intelligence. Data type: String.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.inputFieldNames</code></td>
<td>List of candidate input fields as strings to be considered for estimation. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.label</code></td>
<td>Identifies the prediction task.</td>
</tr>
<tr>
<td>{}</td>
<td></td>
</tr>
<tr>
<td><code>&lt;Object&gt;.name</code></td>
<td>System-assigned name. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.predictedFieldName</code></td>
<td>Identifies a field to be trained for predictability. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.scope</code></td>
<td>Object scope. Currently the only valid value is <code>global</code>. Data type: String.</td>
</tr>
</tbody>
</table>
| `<Object>.trainingFrequency` | The frequency to retrain the model. Possible values:  
  • `every_30_days`  
  • `every_60_days`  
  • `every_90_days`  
  • `every_120_days`  
  • `every_180_days`  
  • `run_once`  
  Default: `run_once`  
  Data type: String. |
The following example gets properties of a predictability estimate object in the store.

```javascript
var mySolution = sn_ml.PredictabilityEstimateStore.get('ml_sn_global_global_predictability_estimate');

gs.print(JSON.stringify(JSON.parse(mySolution.getProperties()), null, 2));
```

Output:

```json
*** Script: { 
  "datasetProperties": { 
    "tableName": "incident", 
    "fieldNames": [ 
      "category", 
      "short_description", 
      "priority", 
      "assignment_group.name"
    ], 
    "fieldDetails": [ 
      { 
        "name": "category", 
        "type": "nominal" 
      }, 
      { 
        "name": "short_description", 
        "type": "text"
      }
    ]
  },
  "domainName": "global",
  "inputFieldNames": [ 
    "short_description"
  ],
  "label": "my estimate definition",
  "name": "ml_x_snc_global_global_my_definition_26",
  "predictedFieldname": "category",
  "processingLanguage": "en",
  "scope": "global",
  "stopwords": [ 
    "Default English Stopwords"
  ],
  "trainingFrequency": "run_once"
}
```
**PredictabilityEstimate - getVersion(String version)**

Gets a predictability estimate by provided version number.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Existing version number of a predictability estimate.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Specified version of the <code>PredictabilityEstimate()</code> object on which you can call <code>PredictabilityEstimateVersion</code> API methods.</td>
</tr>
</tbody>
</table>

The following example shows how to get the training status of a predictability estimate by version number.

```javascript
var mlEstimate = 
    sn_ml.PredictabilityEstimateStore.get('ml_x_snc_global_global_predictability_estimate');

gs.print(JSON.stringify(JSON.parse(mlEstimate.getVersion('1').getStatus()), null, 2));
```

**Output:**

```javascript
{
  "state": "predictability_estimate_complete",
  "percentComplete": "100",
  "hasJobEnded": "true"
}
```

**PredictabilityEstimate - setActiveVersion(String version)**

Activates a specified version of a predictability estimate in the store.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Name of the <code>PredictabilityEstimate()</code> object version to activate. Activating this version deactivates any other version.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to activate a predictability estimate version in the store.

```javascript
sn_ml.PredictabilityEstimate.setActiveVersion("ml_x_snc_global_global_my_estimate_definition");
```

**PredictabilityEstimate - submitTrainingJob()**

Submits a training job.

ℹ️ **Note:** Before running this method, you must first add a predictability estimate to the store using the `PredictabilityEstimateStore - add()` method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td><code>PredictabilityEstimateVersion</code> object corresponding to the <code>PredictabilityEstimate</code> being trained.</td>
</tr>
</tbody>
</table>

The following example shows how to create a dataset, apply it to a predictability estimate, add it to a store, and submit the training job.

```javascript
// Create a dataset
var myData = new sn_ml.DatasetDefinition({
    'tableName': 'incident',
    'fieldNames': ['assignment_group', 'short_description', 'description'],
    'encodedQuery': 'activeANYTHING'
});

// Create an estimate
```
var myEstimate = new sn_ml.PredictabilityEstimate({
  'label': "my estimate definition",
  'dataset': myData,
  'predictedFieldName': 'assignment_group',
  'inputFieldNames': ['short_description']
});

// Add the estimate to the store to later be able to retrieve it.
var my_unique_name = sn_ml.PredictabilityEstimateStore.add(myEstimate);

// Train the estimate - this is a long running job
var myEstimateVersion = myEstimate.submitTrainingJob();

PredictabilityEstimateStore - Global
Enables storing and retrieving predictability estimates.

The PredictabilityEstimateStore API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the sn_ml namespace.

PredictabilityEstimateStore - add(Object mLEstimate)
Adds a new predictability estimate object to the store and returns a unique name.

ℹ️ Note: Label values do not need to be unique. For example, if you run this method with the same label 10 times, this method adds 10 different uniquely-named objects to the store.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mLEstimate</td>
<td>PredictabilityEstimate()</td>
<td>PredictabilityEstimate() object to add to the store.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>System-generated predictability estimate name.</td>
</tr>
</tbody>
</table>
The following example shows how to add a predictability estimate to the store. Use `PredictabilityEstimate - submitTrainingJob()` to run the training job after adding it to the store.

```javascript
// Create a dataset
var myData = new sn_ml.DatasetDefinition({
    'tableName' : 'incident',
    'fieldNames' : ['assignment_group', 'short_description', 'description'],
    'encodedQuery' : 'activeANYTHING'
});

// Create an estimate
var myEstimate = new sn_ml.PredictabilityEstimate({
    'label': "my estimate definition",
    'dataset' : myData,
    'predictedFieldName' : 'assignment_group',
    'inputFieldNames': ['short_description']
});

// Add the estimate to the store to later be able to retrieve it.
var my_unique_name = sn_ml.PredictabilityEstimateStore.add(myEstimate);
```

**PredictabilityEstimateStore - deleteObject(String name)**

Removes a specified predictability estimate object from the store.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the <code>PredictabilityEstimate()</code> object to be deleted.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to delete a predictability estimate from the store.
PredictabilityEstimateStore - get(String name)

Gets a predictability estimate object from a store.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of a predictability estimate in a store.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>PredictabilityEstimate object. Returns an error if the object does not exist.</td>
</tr>
</tbody>
</table>

The following example shows how to get a predictability estimate object from the store using the `get()` method and view its training status using the `PredictabilityEstimate - getActiveVersion()` and `PredictabilityEstimateVersion - getStatus()` methods.

```javascript
// Get status
var mlEstimate = sn_ml.PredictabilityEstimateStore.get('ml_incident_categorization');
gs.print(JSON.stringify(JSON.parse(mlEstimate.getActiveVersion().getStatus(), null, 2)));
```

Output:

```javascript
{
  "state":"solution_complete",
  "percentComplete":"100",
  "hasJobEnded":"true"
}
```

PredictabilityEstimateStore - getAllNames(Object options)

Gets the names of all predictability estimate definition records in the store.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Options for restricting results within the specified properties.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>`{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;label&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;domainName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;scope&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>);</td>
</tr>
<tr>
<td>options.label</td>
<td>String</td>
<td>Optional. Label of your solution object.</td>
</tr>
<tr>
<td>options.domainName</td>
<td>String</td>
<td>Optional. Name of the domain for your solution object. Refer to Domain separation and Predictive Intelligence.</td>
</tr>
<tr>
<td>options.scope</td>
<td>String</td>
<td>Optional. Name of an application scope for your solution object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of strings representing predictability estimate object names in the store.</td>
</tr>
</tbody>
</table>

In the following example, the `getAllNames()` method returns a list of all names in the store.

```javascript
gs.print(JSON.stringify(JSON.parse(sn_ml.PredictabilityEstimateStore.getAllNames()), null, 2));
```

**Output:**

```
[
  "ml_x_snc_global_global_predictability_estimate_1",
  "ml_x_snc_global_global_predictability_estimate",
  "ml_x_snc_global_global_predictability_estimate_2",
  "ml_x_snc_global_global_my_estimate_definition"
]
```

In the following example, the `getAllNames()` method returns only names associated with values set in the `options` parameter.

```
var options = {
  'label': 'my estimate definition',
  'domainName': 'global',
```
var solNames = sn_ml.PredictabilityEstimateStore.getAllNames(options);
gs.print(JSON.stringify(JSON.parse(solNames), null, 2));

Output:

```
[
  "ml_x_snc_global_global_my_estimate_definition"
]
```

**PredictabilityEstimateStore - update(String name, Object mlEstimate)**

Updates a predictability estimate object in a store.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the predictability estimate to update.</td>
</tr>
<tr>
<td>mlEstimate</td>
<td>PredictabilityEstimate()</td>
<td>object properties to update.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to update a predictability estimate object in the store.

```javascript
var estimateUpdate = new sn_ml.PredictabilityEstimate({
  'label': 'my estimate definition',
  'dataset': myData,
  'predictedFieldName': 'assignment_group',
  'inputFieldNames': ['short_description']
});

sn_ml.PredictabilityEstimateStore.update('ml_sn_global_global_incident_service',
  estimateUpdate);
```

**PredictabilityEstimateVersion - Global**

Scriptable object used in Predictive Intelligence stores.
The `PredictabilityEstimateVersion` API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the `sn_ml` namespace.

This API is used for working with predictability estimate versions based on `PredictabilityEstimate` API objects in the `PredictabilityEstimate` store.

The system activates the most recent version of the predictability estimate when it completes training, and only allows one version to be active at a time. However, you can activate any previously trained version you want to use to make predictions.

Methods in this API are accessible using the following `PredictabilityEstimate` methods:

- `getActiveVersion()`
- `getAllVersions()`
- `getLatestVersion()`
- `getVersion()`

`PredictabilityEstimateVersion - getProperties()`

Gets predictability estimate object properties and version number.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Object</td>
</tr>
</tbody>
</table>

```json
{
    "datasetProperties": {Object},
    "domainName": "String",
    "inputFieldNames": [Array],
    "label": "String",
    "name": "String",
    "predictedFieldName": "String",
    "scope": "String",
    "trainingFrequency": "String",
}
```
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>versionNumber</em></td>
<td>&quot;Number&quot;</td>
</tr>
</tbody>
</table>

Properties properties of the `DatasetDefinition()` object associated with the predictability estimate.

```json
{
  "encodedQuery": "String",
  "fieldDetails": [Array],
  "fieldNames": [Array],
  "tableName": "String"
}
```

Data type: Object.

### `<Object>.datasetProperties` properties for the dataset. For example,

```json
{
  "tableName": "Incident"
}
```

Data type: String.

### `<Object>.datasetProperties.fieldNames` fields from the specified table as strings. For example,

```json
"fieldNames": ["short_description", "priority"]
```

Data type: Array.

### `<Object>.datasetProperties.fieldDetails` object fields to specify field properties.

```json
[
  {
    "name": "String",
    "type": "String"
  }
]
```

Data type: Array.

### `<Object>.datasetProperties.fieldDetails.<object>.name` Name of the field defining the type of information to restrict this dataset to.

Data type: String.

### `<Object>.datasetProperties.fieldDetails.<object>.type` Machine-learning field type.

Data type: String.
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.datasetProperties.fieldDetails.encodedQuery</td>
<td>Encoded query string in standard Glide format. See Encoded query strings. Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.domainName</td>
<td>Domain name associated with this dataset. See Domain separation and Predictive Intelligence. Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.inputFieldNames</td>
<td>List of candidate input fields as strings to be considered for estimation. Data type: String.</td>
</tr>
</tbody>
</table>
| <Object>.isActive | Flag that indicates whether this version is active. Valid values:  
  - true: Version is active.  
  - false: Version is not active.  
  Data type: String |
| <Object>.label | Identifies the prediction task.  
  ```json  
  {  
    "label": "my first prediction"  
  }  
  ```  
  Data type: String. |
| <Object>.name | System-assigned name. Data type: String. |
| <Object>.predictedFieldName | Field to be trained for predictability. Data type: String. |
| <Object>.scope | Object scope. Currently the only valid value is global. Data type: String |
| <Object>.trainingFrequency | Frequency to retrain the model. Possible values:  
  - every_30_days  
  - every_60_days |
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• every_90_days</td>
<td></td>
</tr>
<tr>
<td>• every_120_days</td>
<td></td>
</tr>
<tr>
<td>• every_180_days</td>
<td></td>
</tr>
<tr>
<td>• run_once</td>
<td></td>
</tr>
</tbody>
</table>

Default: run_once  
Data type: String.

<Object>.versionNumber

Version number of the PredictabilityEstimate object.  
Data type: String.

The following example gets properties of the active object version in the store.

```javascript
// Get properties
var mlEstimate = sn_ml.PredictabilityEstimateStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlEstimate.getActiveVersion().getProperties()), null, 2));
```

Output:

```json
"datasetProperties": {
    "encodedQuery": "activeANYTHING^EQ",
    "fieldNames": [
        "short_description",
        "category"
    ],
    "tableName": "incident"
},
"domainName": "global",
"inputFieldNames": [
    "short_description"
],
"isActive": "true",
"label": "Incident Categorization_Trainer",
"name": "ml_incident_categorization",
"predictedFieldName": "category",
"processingLanguage": "en",
"stopwords": [
PredictabilityEstimateVersion - getResults()
Returns JSON results containing suggested input fields for an output field.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JSON object results containing suggested input field options for an output field.</td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;output field name&gt;</td>
<td>Output field name, e.g., category, containing suggested input fields.</td>
</tr>
</tbody>
</table>

```json
{
  "<output field name>": {
    "nominalInputFields": [Array],
    "textInputFields": [Array]
  }
}
```

Data type: Object

<table>
<thead>
<tr>
<th>&lt;Object&gt;.&lt;output field name&gt;.nominalInputFields</th>
<th>Nominal input field details.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;nominalInputFields&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;fieldName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;modelImprovement&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;.&lt;output field name&gt;.nominalInputFields.fieldName</code></td>
<td>Nominal input field name. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.&lt;output field name&gt;.nominalInputFields.modelImprovement</code></td>
<td>Score as a relative indication of how likely this field is to improve results. Data type: Number as a string.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.textInputFields</code></td>
<td>Text input field details.</td>
</tr>
<tr>
<td>&quot;textInputFields&quot;: [</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.textInputFields.fieldName</code></td>
<td>Text input field name. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.textInputFields.density</code></td>
<td>Value between 0 and 1.0 that represents frequency that the field is not empty. A value of 1.0 means that the field is not empty in all rows, and a value of 0 indicates that the field is empty in all rows. Data type: Number as a string.</td>
</tr>
</tbody>
</table>

The following example shows how to get results for a selected version of a predictability estimate in the store.

```javascript
// Get results
var estimateName = "ml_x_snc_global_global_predictability_estimate;"
var mlEstimate = sn_ml.PredictabilityEstimateStore.get(estimateName);
var results = mlEstimate.getActiveVersion().getResults();
```
Output:

```javascript
{
  "category": {
    "nominalInputFields": [
      {
        "fieldName": "number",
        "modelImprovement": "0.167052396325189"
      },
      {
        "fieldName": "task_effective_number",
        "modelImprovement": "0.167052396325189"
      }
    ],
    "textInputFields": [
      {
        "fieldName": "short_description",
        "density": "1.0"
      }
    ]
  }
}
```

**PredictabilityEstimateVersion - getStatus(Boolean includeDetails)**

Gets training completion status.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>includeDetail</td>
<td>Boolean</td>
<td>Flag that indicates whether to return status details. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Return additional details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not return additional details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: False</td>
</tr>
<tr>
<td>Returns</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>Object</td>
<td>JavaScript object containing training status information for a <strong>PredictabilityEstimate</strong> object.</td>
<td></td>
</tr>
</tbody>
</table>
| | ```json
{
    "state": "String",
    "percentComplete": "Number as a String",
    "hasJobEnded": "Boolean value as a String",
    "details": {Object}
}
``` |
| **<Object>.state** | Training completion state. If the training job reaches a terminal state, the job does not leave that state. If the state is terminal, the **hasJobEnded** property is set to **true**. Possible values: |
| | • *fetching_files_for_training* |
| | • *preparing_data* |
| | • *retry* |
| | • *solution_cancelled* *(terminal)* |
| | • *solution_complete* *(terminal)* |
| | • *solution_error* *(terminal)* |
| | • *solution_incomplete* |
| | • *training_request_received* |
| | • *training_request_timed_out* *(terminal)* |
| | • *training_solution* |
| | • *uploading_solution* |
| | • *waiting_for_training* |
| Data type: String | |
| **<Object>.hasJobEnded** | Flag that indicates whether training is complete. Valid values: |
| | • **true**: Training is complete. |
| | • **false**: Training is incomplete. |
| Data type: Boolean value as a String | |
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.percentComplete</td>
<td>Number between zero and 100 representing training percent complete. If the completion percentage is less than 100, the job might be in a terminal state. For example, if training times out. Data type: Number as a String</td>
</tr>
<tr>
<td>&lt;Object&gt;.details</td>
<td>Object containing a list of additional training details. Data type: Object</td>
</tr>
</tbody>
</table>

The following example shows a successful result with training complete.

```javascript
// Get status
var mlEstimate = sn_ml.PredictabilityEstimateStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlEstimate.getActiveVersion().getStatus(true), null, 2)));
```

Output:

```
{
  "state": "solution_complete",
  "percentComplete": "100",
  "hasJobEnded": "true",
  "details": {"stepLabel": "Solution Complete"} // This information is only returned if getStatus(true);
}
```

The following example shows an unsuccessful result with training complete.

```javascript
// Get status
var mlEstimate = sn_ml.PredictabilityEstimateStore.get('ml_x_snc_global_global_my_estimate_definition');
var trainingStatus = mlEstimate.getLatestVersion().getStatus();

gs.print(JSON.stringify(JSON.parse(trainingStatus), null, 2));
```

Output:

```
{
  "state": "solution_complete",
  "percentComplete": "100",
```
**PredictabilityEstimateVersion - getVersionNumber()**

Gets the version number of a predictability estimate object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Version number.</td>
</tr>
</tbody>
</table>

The following example shows how to get a version number.

```javascript
// Get version number
var mlEstimate =
    sn_ml.PredictabilityEstimateStore.get('ml_x_snc_global_global_predictability_estimate');

gs.print("Version number:
    "+JSON.stringify(JSON.parse(mlEstimate.getActiveVersion().getVersionNumber()), null, 2));
```

Output:

```
Version number: 1
```

**PreferenceDestination - Scoped, Global**

The PreferenceDestination API updates user notification preferences.

A notification destination is somewhere that a notification can be delivered to, such as a specific email address or phone number. This API is based on notifications from the Notification [sys_notification] table. Notifications are sent through channels such as email or Workspace. A channel can be used to send notifications to multiple types of destinations. For example, an email channel could send notifications to both personal email and work email destinations. Destination types are listed in the Notification Destination Type [sys_notif_destination_type] table.

A user can update their own notification preferences, and an admin can update notification preferences for any user. User notification preferences control which destinations will receive which notifications.
Use this API with the `Preferences` API. Before calling any methods in this class, you must call the `Preferences.getDestinations()` or `Preferences.getDestinationsByChannel()` methods to instantiate a `PreferenceDestination` object.

This class uses the sn_notification namespace identifier.

**PreferenceDestination - getChannel()**

Returns the channel that is used to send notifications to the destination.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>GlideRecord</td>
</tr>
</tbody>
</table>

This example prints the channel that is used to send notifications for each of Abel Tuter's destinations.

```javascript
var recipient = new GlideRecord('sys_user');
recipient.get('last_name', 'Tuter');
var prefs = new sn_notification.Preferences(recipient);
var dests = prefs.getDestinations();
for (var i=0; i<dests.length; i++) {
    gs.print('Channel for destination '+(i+1)+' is '+dests[i].getChannel().name);
}
```

**Output:**

```
Channel for destination 1 is Workspace
```

**PreferenceDestination - getDeliverTo()**

Returns the identifier for the destination.
This example prints the identifier for each of Abel Tuter's destinations. The output shows that the identifier for Abel's Workspace destination is the user sys_id.

```
var recipient = new GlideRecord('sys_user');
recipient.get('last_name', 'Tuter');
var prefs = new sn_notification.Preferences(recipient);
var dests = prefs.getDestinations();
for (var i=0; i<dests.length; i++) {
    gs.print('The identifier for '+recipient.first_name+'\'s '+
       '+dests[i].getDestinationType().name+' destination is '+dests[i].getDeliverTo());
}
```

Output:

```
The identifier for Abel\'s Workspace destination is 62826bf03710200044e0bfc8bcbe5df1
```

PreferenceDestination - getDestinationType()

Returns the type of destination, such as personal email or work email.

```
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>GLideRecord</td>
<td>GlideRecord from the Notification Destination Type [sys_notif_destination_type] table for the destination type.</td>
</tr>
</tbody>
</table>
```
This example prints the identifier for each of Abel Tuter's destinations. The output shows that the identifier for Abel's Workspace destination is the user sys_id.

```javascript
var recipient = new GlideRecord('sys_user');
recipient.get('last_name', 'Tuter');
var prefs = new sn_notification.Preferences(recipient);
var dests = prefs.getDestinations();
for (var i=0; i<dests.length; i++) {
    gs.print('The identifier for '+recipient.first_name+'s
    '+dests[i].getDestinationType().name+' destination is '+dests[i].getDeliverTo());
}
```

Output:

```
The identifier for Abel's Workspace destination is 62826bf03710200044e0bfc8bcbe5df1
```

**PreferenceDestination - isActive(GlideRecord notification)**

Checks if the destination has permission to receive a notification.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notification</td>
<td>GlideRecord</td>
<td>Optional. Specify a notification to check if the destination has permission to receive that notification. An exception is thrown if the notification doesn't exist or if the notification isn't readable by the destination's user. If no notification is specified, this method checks if the destination has permission to receive any notifications.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the destination has permission to receive a notification. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The destination has permission receive a notification.</td>
</tr>
<tr>
<td></td>
<td>• false: The destination doesn't have permission to receive a notification.</td>
</tr>
</tbody>
</table>
This example checks if each of Abel Tuter's destinations has permission to receive the Approval Request notification.

```javascript
var recipient = new GlideRecord('sys_user');
recipient.get('last_name', 'Tuter');
var prefs = new sn_notification.Preferences(recipient);
var dests = prefs.getDestinations();
var notif = new GlideRecord('sys_notification');
notif.get('name', 'Approval Request');
for (var i=0; i<dests.length; i++) {
    var active = dests[i].isActivenotif) ? 'active' : 'inactive';
    gs.print(dests[i].getDestinationType().name + ' ' + dests[i].getDeliverTo() + ' is ' +
             active + ' for this notification');
}
```

Output:

```
Workspace 62826bf03710200044e0bfc8bcbe5df1 is active for this notification
```

PreferenceDestination - isOverridden(GlideRecord notification)
Checks if a notification overrides a user's preferences for the destination.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notification</td>
<td>GlideRecord</td>
<td>Specify a notification to check if it overrides user preferences for the destination. The notification should be a GlideRecord from the Notification [sys_notification] table.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean | Flag that indicates whether the notification overrides user preferences for the destination.  
Valid values:  
• true: User preferences for the destination are ignored for this notification.  
• false: The destination receives or doesn't receive the notification according to the user preference. |
This example checks if the Approval Request notification overrides user preferences for one of Abel Tuter's destinations. If the notification doesn’t override user preferences, the destination is given permission to receive the notification. Otherwise, no preference is set because the notification overrides the preference.

```javascript
var recipient = new GlideRecord('sys_user');
recipient.get('last_name', 'Tuter');
var prefs = new sn_notification.Preferences(recipient);
var dests = prefs.getDestinations();
var notif = new GlideRecord('sys_notification');
notif.get('name', 'Approval Request');
var firstDest = dests[0];
if (!firstDest.isOverridden(notif)) {
    dests[0].setActive(notif, true);
} else {
    gs.print('This notification overrides user preference, so don’t bother to set a preference.');
}
```

PreferenceDestination - setActive(GlideRecord notification, Boolean active)
Sets the user preference for a destination to receive or not receive notifications.

The isOverridden() method can be called first to check if the user preference will be ignored, but it is not required.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notification</td>
<td>GlideRecord</td>
<td>Optional. If a notification is specified, the user preference is set for the destination to receive or not receive that notification. The notification should be a GlideRecord from the Notification [sys_notification] table. If no notification is specified, the user preference is set for the destination to receive or not receive all notifications.</td>
</tr>
<tr>
<td>active</td>
<td>Boolean</td>
<td>Flag that indicates whether the destination has permission to receive a notification. Valid values:</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• true</td>
<td>Type</td>
<td>The destination has permission to receive a notification.</td>
</tr>
<tr>
<td>• false</td>
<td>Type</td>
<td>The destination doesn't have permission to receive a notification.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example checks if the Approval Request notification overrides user preferences for one of Abel Tuter's destinations. If the notification doesn't override user preferences, the destination is given permission to receive the notification. Otherwise, no preference is set because the notification overrides the preference.

```javascript
var recipient = new GlideRecord('sys_user');
recipient.get('last_name', 'Tuter');
var prefs = new sn_notification.Preferences(recipient);
var dests = prefs.getDestinations();
var notif = new GlideRecord('sys_notification');
notif.get('name', 'Approval Request');
var firstDest = dests[0];
if (!firstDest.isOverridden(notif)) {
    dests[0].setActive(notif, true);
} else {
    gs.print('This notification overrides user preference, so don’t bother to set a preference.');
}
```

Preferences - Scoped, Global

The Preferences API gets notification destinations for a user.

A notification destination is somewhere that a notification can be delivered to, such as a specific email address or phone number. This API is based on notifications from the Notification [sys_notification] table. Notifications are sent through channels such as email or Workspace. A channel can be used to send notifications to multiple types of destinations. For example, an email channel could send notifications to both personal email and work email.
destinations. Destination types are listed in the Notification Destination Type [sys_notif_destination_type] table.

This API is used with the PreferenceDestination API to update user notification preferences.

This class uses the sn_notification namespace identifier.

**Preferences - getDestinations()**

Returns a user's notification destinations.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array of PreferenceDestination objects. If the user doesn't have any destinations, the array is empty.</td>
<td></td>
</tr>
</tbody>
</table>

This example gets all of Abel Tuter's notification destinations. The output shows that Abel has one destination.

```javascript
var recipient = new GlideRecord('sys_user');
recipient.get('last_name', 'Tuter');
var prefs = new sn_notification.Preferences(recipient);
var dests = prefs.getDestinations();
gs.print(dests);
```

Output:

[object PreferenceDestination]

**Preferences - getDestinationsByChannel(GlideRecord channel)**

Returns a user's notification destinations that use a specified channel.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel</td>
<td>GlideRecord</td>
<td>GlideRecord from the Notification Channel [sys_notification_channel] table for the channel you want to filter on.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array of PreferenceDestination objects. If the user doesn't have any notification destinations that use the channel or if the channel doesn't exist, the array is empty.</td>
</tr>
</tbody>
</table>

This example gets all of Abel Tuter's destinations that use the Workspace channel. The output shows that Abel has one destination that uses the Workspace channel.

```javascript
var recipient = new GlideRecord('sys_user');
recipient.get('last_name', 'Tuter');
var prefs = new sn_notification.Preferences(recipient);
var channel = new GlideRecord('sys_notification_channel');
channel.get('name', 'Workspace');
var dests = prefs.getDestinationsByChannel(channel);
gs.print(dests);
```

Output:

```
[object PreferenceDestination]
```

### Preferences - Preferences(GlideRecord recipient)

Instantiates a Preferences object for a specified user.

This example instantiates a Preferences object for the user Abel Tuter.
var recipient = new GlideRecord('sys_user');
recipient.get('last_name', 'Tuter');
var prefs = new sn_notification.Preferences(recipient);

**ProbeHandlerCim - Global**
Configures a CIM probe.
Use in any server-side script where you need to configure a CIM probe.

**ProbeHandlerCim - getCimQueries()**
Returns a string containing the CIM fields to fetch

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The CIM fields to fetch</td>
</tr>
</tbody>
</table>

**ProbeHandlerCim - getNamespace()**
Returns the CIM name space.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name space</td>
</tr>
</tbody>
</table>
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A hash map of the parameters</td>
</tr>
</tbody>
</table>

#### ProbeHandlerCim - setProbeParameters(Object params)
Sets the probe parameters.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>params</td>
<td>Object</td>
<td>The parameters to add</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### ProbeHandlerCim - run()
Runs the probe.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
ProbeHandlerMulti - Global

Probe handler for MultiProbes.

This script include does not work with probe parameters that have value_scripts and does not work with JavaScript probes.

Use in any server-side script where you need to configure a MultiProbe.

ProbeHandlerMulti - addParameters(String parent, String sysID, GlideRecord probe)

Adds the probe parameters to the given XML parent element, taken from the given probe's parameter table.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>parent</td>
</tr>
<tr>
<td>sysID</td>
</tr>
<tr>
<td>probe</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

ProbeHandlerMulti - addProbes(String parent, String sysID)

Adds the probes for this multiprobe to the document.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>parent</td>
</tr>
<tr>
<td>sysID</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**ProbeHandlerMulti - run()**

Runs the probe.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**ProbeHandlerMulti - runProbeHandler(String params, String sysID, GlideRecord probe)**

If the given probe’s GlideRecord is present and contains a probe handler, runs the probe handler and adds any parameters it creates to the given parameter element.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>params</td>
<td>String</td>
<td>The XML parameter element to add parameters to.</td>
</tr>
<tr>
<td>sysID</td>
<td>String</td>
<td>The sys_id of the probe to get parameters from.</td>
</tr>
<tr>
<td>probe</td>
<td>GlideRecord</td>
<td>Optional GlideRecord of the probe. If present, and there is a probe handler, it is called.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
**ProbeHandlerMulti - setParameter(String parent, String name, String value)**

Sets the value of a given XML parameter element.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent</td>
<td>String</td>
<td>The XML parent element to add the parameter to.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the parameter to set</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The value to set the parameter to.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**ProbeHandlerSNMP - Global**

Discovery probe handler for SNMP probes, invoked when an SNMP probe is about to be created.

Use in any server-side script where you need to configure an SNMP probe.

**ProbeHandlerSNMP - addBoolean(String dbName, String paramName, String paramMap)**

Adds a boolean value to the parameter map.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbName</td>
<td>String</td>
<td>The database name</td>
</tr>
<tr>
<td>paramName</td>
<td>String</td>
<td>The parameter name</td>
</tr>
<tr>
<td>paramMap</td>
<td>String</td>
<td>The parameter map</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
 ProbeHandlerSNMP - addDebug(String paramMap)

Adds debugging to the probe handler.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramMap</td>
<td>String</td>
<td>The parameter map</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

 ProbeHandlerSNMP - addJavascriptHandlers(String paramMap)

Adds a JavaScript handler.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramMap</td>
<td>String</td>
<td>The parameter map</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

 ProbeHandlerSNMP - addOIDAutoResolve(String paramMap)

Adds OID automatic resolution.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramMap</td>
<td>String</td>
<td>The parameter map</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

ProbeHandlerSNMP - addOIDSpec(String paramMap)

Adds OID specifications to the parameter map.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramMap</td>
<td>String</td>
<td>The parameter map</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

ProbeHandlerSNMP - addSensorHandler(String paramMap)

Adds a sensor handler.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramMap</td>
<td>String</td>
<td>The parameter map</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

ProbeHandlerSNMP - addStage(String paramMap)

Adds the stage to the parameter map.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramMap</td>
<td>String</td>
<td>The parameter map</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**ProbeHandlerSNMP - addTimingSpecs(String paramMap)**

Adds timing specifications to the parameter map.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramMap</td>
<td>String</td>
<td>The parameter map</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**ProbeHandlerSNMP - getParameters()**

Returns the parameters added by this probe handler.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A hash map of the parameters added by this probe handler.</td>
</tr>
</tbody>
</table>
**ProbeHandlerSNMP - run()**

Runs the probe.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

**ProbeHandlerSNMP - setProbeParameters(String params)**

Sets the value of a given XML parameter element.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>params</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

**ProbeHandlerWMI - Global**

Discovery probe handler for WMI probes.

Use in any server-side script where you need to configure an WMI probe.

**ProbeHandlerWMI - getParameters()**

Returns the parameters added by this probe handler.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A hash map of the parameters added by this probe</td>
</tr>
</tbody>
</table>

**ProbeHandlerWMI - getWMIFields()**

Generates a string containing the WMI fields to fetch.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The WMI fields to fetch.</td>
</tr>
</tbody>
</table>

**ProbeHandlerWMI - run()**

Runs the probe.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
ProbeHandlerWMI - setProbeParameters(Object params)
Sets the probe parameters.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>params</td>
<td>Object</td>
<td>The probe parameters</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### Queue - Global

Use the `Queue` API (Advanced Work Assignment) to route a document to a queue.

The `Queue` API is provided within the `sn_awa` namespace.

Requires the following:
- Advanced Work Assignment plugin (com.glide.awa)
- Role: awa_integration_user or admin

#### Queue - get(String queue_id)

Gets a queue record by sys_id.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queue_id</td>
<td>String</td>
<td>Represents sys_id of queue record from awa_queue table</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if queue record exists, false otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var queue = sn_awa.Queue.get("<queue_id>");
var workItem = queue.route(
    document_sys_id: "<sys_id>",
```
Queue – route(Object options)

Routes an active work item to the queue object.

If an active work item does not exist for the specified document, this method creates a work item and set its fields accordingly.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Contains document sys_id and table associated with work item:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>document_sys_id: &quot;&lt;sys_id&gt;&quot;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>document_table: &quot;&lt;table&gt;&quot;, e.g., interaction</code></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if successful, error message otherwise.</td>
</tr>
</tbody>
</table>

```javascript
var queue = sn_awa.Queue.get("<queue_id>"));
var workItem = queue.route({
    document_sys_id: "<sys_id>",
    document_table: "<table>
});
```

Queue - Scoped

The Queue API allows you to retrieve or join a Connect Support chat queue.

To use this class in a scoped application, use the `sn_connect` namespace identifier. The Connect Scriptable APIs plugin (ID: com.glide.connect.scriptable) should be enabled to access the Queue API.

**Scoped Queue - get(String sysID)**

Retrieves a specified chat queue.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SysID</td>
<td>String</td>
<td>Sys_id of a queue from the Chat Queue [chat_queue] table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Returns a conversation queue object.</td>
</tr>
</tbody>
</table>

```javascript
var queue = sn_connect.Queue.get("ab73be7dc09a4300964f336ee6b74361");
```

 Scoped Queue - join(String question)
 Adds the current user to an existing Connect Support chat queue and posts the specified question.

Before calling this method, you must call the Scoped Queue - get(String sysID) method to retrieve the chat queue to which to attach the current user.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question</td>
<td>String</td>
<td>Question to add to the chat queue.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>GlideRecord object for the Chat Queue Entry.</td>
</tr>
</tbody>
</table>

```javascript
var queue = sn_connect.Queue.get("ab73be7dc09a4300964f336ee6b74361");
queue.join("How do I access my email?");
```

QuickRanges - Global

Generates IP network, range, and address entries from a convenient comma-separated input field using conventional CIDR network notation, hyphenated range entries, or individual IP addresses.

Use with any server-side discovery script.
QuickRanges - createItem(String table, String id, String type)
Creates a new discovery range item.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>The table where the item will be created.</td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>The identifier to use for the new item.</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>The type of entries to generate: IP address, IP network, or IP range.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>The created entry</td>
</tr>
</tbody>
</table>

QuickRanges - onMakeRanges()
Returns the IP network, range, and address information to use when generating the entries.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

RecordToHTML - Global
A utility class to turn a record into HTML.
The RecordToHTML class is available to server-side scripts.
RecordToHTML - RecordToHTML(String table, String sys_id, String pattern, Boolean link)

Creates an instance of RecordToHTML class.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>String</td>
<td>The record's table name</td>
</tr>
<tr>
<td>sys_id</td>
<td>String</td>
<td>The sys_id of the record</td>
</tr>
<tr>
<td>pattern</td>
<td>String</td>
<td>The pattern of the string to generate. The pattern may include ${} escapes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>for fields whose values should be included. For example, the pattern</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;sys_id: ${sys_id}&quot; would substitute the actual sys_id for the escape.</td>
</tr>
<tr>
<td>link</td>
<td>Boolean</td>
<td></td>
</tr>
</tbody>
</table>

```
var r2html = new RecordToHTML("incident","e8e875b0c0a80164009dc852b4d677d5",
    "incident: ${number}-${short_description} (${user})", true);
gs.print(r2html.toString());
```

Output: incident: INC00005-CPU load high for over 10 minutes

RecordToHTML - setValue(String fieldName, String value)

Sets the specified field to the specified value.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldName</td>
<td>String</td>
<td>Name of the field to change.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value to set the field to.</td>
</tr>
</tbody>
</table>

```
var r2html = new RecordToHTML("incident","e8e875b0c0a80164009dc852b4d677d5", "incident: ${number}-${short_description} (${user})", true);
```
r2html.setValue("user", gs.getUserName());
gs.print(r2html.toString());

Output: incident: INC00005-CPU load high for over 10 minutes (john.roberts)

**RecordToHTML - toString()**
Converts the record to a string.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

```javascript
var r2html = new RecordToHTML("incident","e8e875b0c0a80164009dc852b4d677d5",
            "incident: ${number}-${short_description}", true);
gs.print(r2html.toString());
```

Output: incident: INC00005-CPU load high for over 10 minutes

**RegressionSolution - Global**
Scriptable object used in Predictive Intelligence stores.

The `RegressionSolution` API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the `sn_ml` namespace.

The solution setup-to-training flow is as follows:

1. Create a dataset using the `DatasetDefinition` API.
2. Optional. Build an encoder using the `Encoder` API.
3. Use the constructor to create a regression solution object.
4. Add the solution object to the regression solution store using the `RegressionSolutionStore - add()` method.
5. Train the solution using the `submitTrainingJob()` method. This creates a version of the object that you can manage using the `RegressionSolutionVersion` API.
**Note:** This API runs with full privileges. To restrict user access, include an access control mechanism in the script.

For usage guidelines, refer to [Using ML APIs](#).

**RegressionSolution - RegressionSolution(Object config)**

Creates a regression solution.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>Object</td>
<td>JavaScript object containing configuration properties of the solution.</td>
</tr>
<tr>
<td>config.dataset</td>
<td>Object</td>
<td>DatasetDefinition name.</td>
</tr>
<tr>
<td>config.domainName</td>
<td>String</td>
<td>Optional. Domain name associated with this dataset. See Domain separation and Predictive Intelligence. Default: Current domain, for example, &quot;global&quot;.</td>
</tr>
<tr>
<td>config.encoder</td>
<td>Object</td>
<td>Optional. Trained encoder object to assign to this solution. See Encoder - Encoder(Object config).</td>
</tr>
<tr>
<td>config.inputFieldNames</td>
<td>Array</td>
<td>List of input field names as strings. The model uses these fields used to make predictions.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>config.label</td>
<td>String</td>
<td>Identifies the prediction task.</td>
</tr>
<tr>
<td>config.minRowCount</td>
<td>String</td>
<td>Optional. Minimum number of records required in the dataset for training. Default: 10000</td>
</tr>
<tr>
<td>config.predictedFieldName</td>
<td>String</td>
<td>Mandatory unless setting <strong>predictedInterval</strong>. Identifies a field to be trained for predictability.</td>
</tr>
<tr>
<td>config.predictedInterval</td>
<td>Array</td>
<td>Mandatory unless setting <strong>predictedFieldName</strong>. Sets a range of fields to train your solution for confidence. Supports providing 2 non-numeric date fields. For example, 'predictedInterval': ['sys_updated_on', 'sys_created_on'].</td>
</tr>
<tr>
<td>config.processingLanguage</td>
<td>String</td>
<td>Optional. Processing language in two-letter ISO 639-1 language code format. Default: &quot;en&quot;</td>
</tr>
<tr>
<td>config.stopwords</td>
<td>Array</td>
<td>Optional. Preset list of strings that the system automatically generates based on the <strong>language</strong> property setting. For details, see <a href="#">Create a custom stopwords list</a>. Default: English Stopwords</td>
</tr>
<tr>
<td>config.trainingFrequency</td>
<td>String</td>
<td>The frequency to retrain the model. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_30_days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_60_days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_90_days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_120_days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_180_days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• run_once</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
<th>Default: run_once</th>
</tr>
</thead>
</table>

The following example shows how to create an object and add it to the RegressionSolution store.

```javascript
var myNewData = new sn_ml.DatasetDefinition({
    'tableName': 'incident',
    'fieldNames': ['category', 'short_description', 'priority'],
    'fieldDetails': [
        {'name': 'category', 'type': 'nominal'},
        {'name': 'short_description', 'type': 'text'}
    ],
    'encodedQuery': 'activeANYTHING'
});

var mySimSolution = new sn_ml.SimilaritySolution({
    'label': 'my solution definition',
    'dataset': myNewData,
    'predictedFieldName': 'category',
    'inputFieldNames': ['short_description']
});

var mySimilarityName = sn_ml.SimilaritySolutionStore.add(mySimSolution);
```

The following example shows how to create an object to train using the `predictedInterval` property.

```javascript
var myIncidentData = new sn_ml.DatasetDefinition({
    'tableName': 'incident',
    'fieldNames': ['short_description', 'sys_updated_on', 'sys_created_on'],
    'encodedQuery': 'activeANYTHING'
});

var mySolution = new sn_ml.RegressionSolution({
    'label': 'reg assinGroup',
});
```
_regressionSolution - cancelTrainingJob()

Cancels a job for a solution object that has been submitted for training.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to cancel an existing training job.

```javascript
var mySolution = sn_ml.RegressionSolutionStore.get('ml_sn_global_global_regression');
mySolution.cancelTrainingJob();
```

_regressionSolution - getActiveVersion()

Gets the active RegressionSolutionVersion object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Active RegressionSolutionVersion object.</td>
</tr>
</tbody>
</table>
The following example shows how to get an active `RegressionSolution` version from the store and return its training status.

```javascript
var mlSolution = sn_ml.RegressionSolutionStore.get('ml_x_snc_global_global_regression');
gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getStatus()), null, 2));
```

Output:

```
{
  "state": "solution_complete",
  "percentComplete": "100",
  "hasJobEnded": "true"
}
```

**RegressionSolution - getAllVersions()**

Gets all versions of a `RegressionSolution` object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Existing versions of a solution object. See also <code>RegressionSolutionVersion</code> API.</td>
</tr>
</tbody>
</table>

The following example shows how to get all `RegressionSolution` version objects and call the `getUrlVersionNumber()` and `getStatus()` solution version methods on them.

```javascript
var mlSolution = sn_ml.RegressionSolutionStore.get('ml_x_snc_global_global_regression');
var mlSolutionVersions = mlSolution.getAllVersions();

for (i = 0; i < mlSolutionVersions.length; i++) {
gs.print("Version " + mlSolutionVersions[i].getUrlVersionNumber() + " Status: " + mlSolutionVersions[i].getStatus() +"\n");
}
```

Output:
RegressionSolution - getLatestVersion()

Gets the latest version of a solution.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>RegressionSolutionVersion object corresponding to the latest version of a RegressionSolution().</td>
</tr>
</tbody>
</table>

The following example shows how to get the latest version of a solution and return its training status.

```javascript
var mlSolution = sn_ml.RegressionSolutionStore.get('ml_x_snc_global_global_regression');
gs.print(JSON.stringify(JSON.parse(mlSolution.getLatestVersion().getStatus()), null, 2));
```

Output:

```json
{
  "state": "solution_complete",
  "percentComplete": "100",
  "hasJobEnded": "true"
}
```

RegressionSolution - getName()

Gets the name of the object to use for interaction with the store.
The following example shows how to update `RegressionSolution` dataset information and print the name of the object.

```javascript
// Update solution
var myIncidentData = new sn_ml.DatasetDefinition({
    'tableName' : 'incident',
    'fieldNames' : ['category', 'short_description', 'priority'],
    'encodedQuery' : 'activeANYTHING'
});

var eligibleFields = JSON.parse(myIncidentData.getEligibleFields('regression'));

var myRegression = new sn_ml.RegressionSolution({
    'label': "my regression solution",
    'dataset' : myIncidentData,
    'inputFieldNames': eligibleFields['eligibleInputFieldNames'],
    'predictedFieldName': 'category'
});

// update solution
sn_ml.RegressionSolutionStore.update('ml_x_snc_global_global_my_solution_definition_4',
    myRegression);

// print solution name
gs.print('Solution Name: ' + myRegression.getName());
```

Output:

```
Solution Name: ml_x_snc_global_global_my_solution_definition_4
```

**RegressionSolution - getProperties()**

Gets solution object properties.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Contents of the Dataset and RegressionSolution() object details in the RegressionSolutionStore object.</td>
</tr>
</tbody>
</table>

```json
{
  "datasetProperties":
  {Object},
  "domainName": "String",
  "encoder": {Object},
  "inputFieldNames": [Array],
  "label": "String",
  "name": "String",
  "predictedFieldName": "String",
  "predictedInterval": [Array],
  "processingLanguage": "String",
  "scope": "String",
  "stopwords": [Array],
  "trainingFrequency": "String"
}
```

```json
<Object>.datasetProperties
```

Lists the properties of the DatasetDefinition object associated with the solution.

```json
{
  "encodedQuery": "String",
}
```
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>&quot;fieldDetails&quot;:</strong></td>
</tr>
<tr>
<td></td>
<td>[Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;fieldName&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;tableName&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

Data type: Object.

**<Object>.datasetProperties.tableName**

Name of the table for the dataset. For example, "tableName" : "Incident".

Data type: String.

**<Object>.datasetProperties.fieldNames**

List of field names from the specified table as strings. For example, "fieldNames" : ["short_description", "priority"].

Data type: Array.

**<Object>.datasetProperties.fieldNames.fieldDetails**

List of JavaScript objects that specify field properties.

```
[
  {
    "name": "String",
    "type": "String"
  }
]
```

Data type: Array.

**<Object>.datasetProperties.fieldNames.fieldDetails.<object>.name**

Name of the field defining the type of information to restrict this dataset to.

Data type: String.

**<Object>.datasetProperties.fieldDetails.<object>.type**

Machine-learning field type.
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;.datasetProperties.fieldDetails.encodedQuery</code></td>
<td>Encoded query string in standard Glide format. See <a href="#">Encoded query strings</a>. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.domainName</code></td>
<td>Domain name associated with this dataset. See <a href="#">Domain separation and Predictive Intelligence</a>. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.encoder</code></td>
<td>Encoder object assigned to this solution. See <a href="#">Encoder</a>. Object.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.inputFieldNames</code></td>
<td>List of input field names as strings. The model uses these fields used to make predictions. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.label</code></td>
<td>Identifies the prediction task. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.name</code></td>
<td>System-assigned name. Data type: String.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;.predictedFieldName</code></td>
<td>Identifies a field to be trained for predictability. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.predictedInterval</code></td>
<td>Range of values specifying the prediction confidence level. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.processingLanguage</code></td>
<td>Processing language in two-letter ISO 639-1 language code format. Data type: String</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.scope</code></td>
<td>Object scope. Currently the only valid value is <code>global</code>. Data type: String</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.stopwords</code></td>
<td>Optional. Preset list of strings that the system automatically generates based on the <code>language</code> property setting. For details, see Create a custom stopwords list. Data type: Array.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.trainingFrequency</code></td>
<td>The frequency to retrain the model. Possible values:</td>
</tr>
<tr>
<td></td>
<td>* every_30_days</td>
</tr>
<tr>
<td></td>
<td>* every_60_days</td>
</tr>
<tr>
<td></td>
<td>* every_90_days</td>
</tr>
<tr>
<td></td>
<td>* every_120_days</td>
</tr>
</tbody>
</table>
The following example gets properties of a solution object in the store.

```javascript
var mySolution = sn_ml.RegressionSolutionStore.get('ml_sn_global_global_regression_solution');

gs.print(JSON.stringify(JSON.parse(mySolution.getProperties()), null, 2));
```

Output:

```json
*** Script: {
  "datasetProperties": {
    "tableName": "cloudinfratext",
    "fieldNames": [
      "short_description",
      "sourcedc",
      "targetdc",
      "dbsize",
      "duration"
    ],
  },
  "domainName": "global",
  "encoderProperties": {
    "datasetsProperties": [],
    "name": "wc_regression"
  },
  "inputFieldNames": [
    "short_description",
    "sourcedc",
    "targetdc",
    "dbsize"
  ],
  "label": "Regression Test for DB Restore",
  "name": "ml_x_snc_global_global_regression",
  "predictedFieldNames": "duration",
}***
```
RegressionSolution - getVersion(String version)

Gets a solution by provided version number.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Existing version number of a solution.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Specified version of the RegressionSolution() object on which you can call</td>
</tr>
<tr>
<td></td>
<td>RegressionSolutionVersion API methods.</td>
</tr>
</tbody>
</table>

The following example shows how to get the training status of a solution by version number.

```javascript
var mlSolution = sn_ml.RegressionSolutionStore.get('ml_x_snc_global_global_regression');
gs.print(JSON.stringify(JSON.parse(mlSolution.getVersion('1').getStatus()), null, 2));
```

Output:

```json
{
    "state": "solution_complete",
    "percentComplete": "100",
    "hasJobEnded": "true"
}
```

RegressionSolution - setActiveVersion(String version)

Activates a specified version of a solution in the store.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Name of the <code>RegressionSolution()</code> object version to activate. Activating this version deactivates any other version.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to activate a solution version in the store.

```javascript
sn_ml.RegressionSolution.setActiveVersion("ml_incident_categorization");
```

**RegressionSolution - submitTrainingJob()**

Submits a training job.

⚠️ **Note:** Before running this method, you must first add a solution to the store using the `RegressionSolutionStore - add()` method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td><code>RegressionSolutionVersion</code> object corresponding to the <code>RegressionSolution</code> being trained.</td>
</tr>
</tbody>
</table>

The following example shows how to create a dataset, apply it to a solution, add the solution to a store, and submit the training job.

```javascript
// Create a dataset
var myData = new sn_ml.DatasetDefinition({
  'tableName' : 'incident',
});

```
RegressionSolutionStore - Global

Enables storing and retrieving solutions.

The `RegressionSolutionStore` API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the `sn_ml` namespace.

`RegressionSolutionStore - add(Object mlSolution)`

Adds a new solution object to the store and returns a unique name.

⚠️ **Note:** Label values do not need to be unique. For example, if you run this method with the same label 10 times, this method adds 10 different uniquely-named objects to the store.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mlSolution</td>
<td>RegressionSolution</td>
<td>object to add to the store.</td>
</tr>
</tbody>
</table>
The following example shows how to add a solution to the store. Use `RegressionSolution: submitTrainingJob()` to train the Solution after adding it to the store.

```javascript
// Create a dataset
var myData = new sn_ml.DatasetDefinition({
  'tableName': 'incident',
  'fieldNames': ['assignment_group', 'short_description', 'description'],
  'encodedQuery': 'activeANYTHING'
});

// Create an solution
var mySolution = new sn_ml.RegressionSolution({
  'label': 'my solution definition',
  'dataset': myData,
  'predictedFieldName': 'assignment_group',
  'inputFieldNames': ['short_description']
});

// Add the solution to the store to later be able to retrieve it.
var my_unique_name = sn_ml.RegressionSolutionStore.add(mySolution);
```

**RegressionSolutionStore - deleteObject(String name)**

Removes a specified solution object from the store.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the RegressionSolution() object to be deleted.</td>
</tr>
</tbody>
</table>
The following example shows how to delete a solution from the store.

```javascript
sn_ml.RegressionSolutionStore.deleteObject("ml_sn_global_global_solution");
```

**RegressionSolutionStore - get(String name)**

Gets a solution object from the store.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of a solution in the store.</td>
</tr>
</tbody>
</table>

The following example shows how to get a solution object from the store using the `get()` method and view its training status using the `RegressionSolution - getActiveVersion()` and `RegressionSolutionVersion - getStatus()` methods.

```javascript
// Get status
var mlSolution = sn_ml.RegressionSolutionStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getStatus(), null, 2)));
```

Output:

```json
{
  "state":"solution_complete",
  "percentComplete":"100",
  "hasJobEnded":"true"
}
```

**RegressionSolutionStore - getAllNames(Object options)**

Gets the names of all solution definition records in the store.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Options for restricting results within the specified properties.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>`{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;label&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;domainName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;scope&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>`};</td>
</tr>
<tr>
<td>options.label</td>
<td>String</td>
<td>Optional. Label of your solution object.</td>
</tr>
<tr>
<td>options.domainName</td>
<td>String</td>
<td>Optional. Name of the domain for your solution object. Refer to <a href="#">Domain separation and Predictive Intelligence</a>.</td>
</tr>
<tr>
<td>options.scope</td>
<td>String</td>
<td>Optional. Name of an application scope for your solution object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of strings representing solution object names in the store.</td>
</tr>
</tbody>
</table>

In the following example, the `getAllNames()` method returns a list of all names in the store.

```js
gs.print(JSON.stringify(JSON.parse(sn_ml.RegressionSolutionStore.getAllNames()), null, 2));
```

**Output:**

```json
[
  "ml_x_snc_global_global_my_solution_definition_3",
  "ml_incident_assignment",
  "ml_x_snc_global_global_my_solution_definition",
  "ml_x_snc_global_global_my_solution_definition_2",
  "ml_sn_global_global_incident_service"
]
```

In the following example, the `getAllNames()` method returns only names associated with values set in the `options` parameter.
var options = {
    'label': 'my solution definition',
    'domainName': 'global',
    'scope': 'global'
};
var solNames = sn_ml.RegressionSolutionStore.getAllNames(options);
gs.print(JSON.stringify(JSON.parse(solNames), null, 2));

Output:

```
[
    "ml_x_snc_global_global_my_solution_definition"
]
```

**RegressionSolutionStore - update(String name, Object mlsolution)**

Replaces an existing object in the store with the object passed as a parameter. The object name provided must be empty or match.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the solution to update.</td>
</tr>
<tr>
<td>mlsolution</td>
<td>RegressionSolution()</td>
<td>object properties to update.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to update a solution object in the store.

```
var solutionUpdate = new sn_ml.RegressionSolution({
    'label': 'my solution definition',
    'dataset': myData,
    'predictedFieldName': 'assignment_group',
    'inputFieldNames': ['short_description']
});

sn_ml.RegressionSolutionStore.update('ml_sn_global_global_incident_service',
    solutionUpdate);
```
RegressionSolutionVersion - Global

Scriptable object used in Predictive Intelligence stores.

The RegressionSolutionVersion API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the sn_ml namespace.

This API is used for working with solution versions based on RegressionSolution API objects in the RegressionSolution store.

The system creates a solution version each time you train a solution definition. Most versions are created during scheduled solution training.

Methods in this API are accessible using the following RegressionSolution methods:

• `getActiveVersion()`
• `getAllVersions()`
• `getLatestVersion()`
• `getVersion()`

RegressionSolutionVersion - getProperties()

Gets solution object properties and version number.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Contents of the Dataset and RegressionSolution version details. Results vary by object property setup.</td>
</tr>
</tbody>
</table>

```json
{
    "datasetProperties": {Object},
    "domainName": "String",
    "encoder": {Object},
    "inputFieldNames": [Array],
    "isActive": "Boolean",
    "label": "String",
    "name": "String",
    "predictedFieldName": "String",
```
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;predictedInterval&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;processingLanguage&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;scope&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;stopwords&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;versionNumber&quot;: &quot;Number&quot;</td>
</tr>
</tbody>
</table>

### <Object>.datasetProperties properties of the DatasetDefinition() object associated with the solution.

```json
{
   "encodedQuery": "String",
   "fieldDetails": [Array],
   "fieldNames": [Array],
   "tableName": "String"
}
```

### <Object>.datasetProperties tableName property of the dataset. For example, "tableName" : "Incident".

Data type: String.

### <Object>.datasetProperties fieldNames property specifies field names from the specified table as strings. For example, "fieldNames" : ["short_description", "priority"].

Data type: Array.

### <Object>.datasetProperties fieldDetails property specifies field properties.

```json
[
   {
      "name": "String",
      "type": "String"
   }
]
```

Data type: Array.

### <Object>.datasetProperties fieldDetails.type property of the FieldDefinition object inform you to restrict this dataset to.

Data type: String.

### <Object>.datasetProperties fieldDetails.type property of the FieldDefinition object inform you to restrict this dataset to.

Data type: String.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.datasetProperties.fieldDetails.encodedQuery</td>
<td>String encoded query string in standard Glide format. See Encoded query strings.</td>
</tr>
<tr>
<td>&lt;Object&gt;.domainName</td>
<td>Domain name associated with this dataset. See Domain separation and Predictive Intelligence.</td>
</tr>
<tr>
<td>&lt;Object&gt;.encoder</td>
<td>Encoder object assigned to this solution. See Encoder - Encoder(Object config).</td>
</tr>
<tr>
<td>&lt;Object&gt;.inputFieldNames</td>
<td>List of input field names as strings. The model uses these fields used to make predictions.</td>
</tr>
</tbody>
</table>
| <Object>.isActive | Flag that indicates whether this version is active. Valid values:  
  - true: Version is active.  
  - false: Version is not active. |
| <Object>.label | Identifies the prediction task.  
  ```json  
  {  
    "label": "my first prediction"  
  }  
  ``` |
| <Object>.name | System-assigned name. |
| <Object>.predictedFieldName | Field to be trained for predictability. |
| <Object>.predictedInterval | Range of values specifying the prediction confidence level. |
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;.processingLanguage</code></td>
<td>Processing language in two-letter ISO 639-1 language code format. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.scope</code></td>
<td>Object scope. Currently the only valid value is <code>global</code>. Data type: String</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.stopwords</code></td>
<td>Optional. Preset list of strings that the system automatically generates based on the <code>language</code> property setting. For details, see Create a custom stopwords list. Data type: Array.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.versionNumber</code></td>
<td>Version number of the RegressionSolution object.</td>
</tr>
</tbody>
</table>

The following example gets properties of the active object version in the store.

```javascript
// Get properties
var mlSolution = sn_ml.RegressionSolutionStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getProperties()), null, 2));
```

Output:

```json
{
  "datasetProperties": {
    "encodedQuery": "",
    "fieldNames": ["short_description", "sourcedc", "targetdc", "dbsize", "duration"],
    "tableName": "cloudinfratext"
  },
  "domainName": "global",
  "encoderProperties": {
    "datasetsProperties": [],
    "name": "wc_regression"
  }
}
```
RegressionSolutionVersion - getStatus(Boolean includeDetails)

Gets training completion status.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| includeDetails | Boolean  | Flag that indicates whether to return status details. Valid values:  
|              |          | • true: Return additional details.  
|              |          | • false: Do not return additional details. Default: False |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Object | JavaScript object containing training status information for a RegressionSolution object.  

```javascript
{
  "state": "String",
  "percentComplete": "Number as a String",
  "hasJobEnded": "Boolean value as a String",
}
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;details&quot;: {Object}</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.state</td>
<td>Training completion state. If the training job reaches a terminal state, the job does not leave that state. If the state is terminal, the hasJobEnded property is set to true. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• fetching_files_for_training</td>
</tr>
<tr>
<td></td>
<td>• preparing_data</td>
</tr>
<tr>
<td></td>
<td>• retry</td>
</tr>
<tr>
<td></td>
<td>• solution_cancelled (terminal)</td>
</tr>
<tr>
<td></td>
<td>• solution_complete (terminal)</td>
</tr>
<tr>
<td></td>
<td>• solution_error (terminal)</td>
</tr>
<tr>
<td></td>
<td>• solution_incomplete</td>
</tr>
<tr>
<td></td>
<td>• training_request_received</td>
</tr>
<tr>
<td></td>
<td>• training_request_timed_out (terminal)</td>
</tr>
<tr>
<td></td>
<td>• training_solution</td>
</tr>
<tr>
<td></td>
<td>• uploading_solution</td>
</tr>
<tr>
<td></td>
<td>• waiting_for_training</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.hasJobEnded</td>
<td>Flag that indicates whether training is complete. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Training is complete.</td>
</tr>
<tr>
<td></td>
<td>• false: Training is incomplete.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean value as a String</td>
</tr>
<tr>
<td>&lt;Object&gt;.percentComplete</td>
<td>Number between zero and 100 representing training percent complete. If the completion percentage is less than 100, the job might be in a terminal state. For example, if training times out.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number as a String</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Object&gt;.details</code></td>
<td>Object containing a list of additional training details.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
</tbody>
</table>

The following example shows a successful result with training complete.

```javascript
// Get status
var mlSolution = sn_ml.RegressionSolutionStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getStatus(true), null, 2)));
```

Output:

```
{
    "state": "solution_complete",
    "percentComplete": "100",
    "hasJobEnded": "true",
    "details": {"stepLabel": "Solution Complete"} // This information is only returned if getStatus(true);
}
```

The following example shows an unsuccessful result with training complete.

```javascript
// Get status
var solutionName = 'ml_x_snc_global_global_regression_solution';
var mlSolution = sn_ml.RegressionSolutionStore.get(solutionName);
var trainingStatus = mlSolution.getLatestVersion().getStatus();

gs.print(JSON.stringify(JSON.parse(trainingStatus), null, 2));
```

Output:

```
{
    "state": "solution_error",
    "percentComplete": "100",
    "hasJobEnded": "true"
}
```

RegressionSolutionVersion - getVersionNumber()

Gets the version number of a solution object.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Version number.</td>
</tr>
</tbody>
</table>

The following example shows how to get a version number.

```javascript
// Get version number
var mlSolution = sn_ml.RegressionSolutionStore.get('ml_incident_categorization');

gs.print("Version number:");
+JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getVersionNumber()), null, 2));
```

Output:

Version number: 1

RegressionSolutionVersion - predict(Object input, Object options)

Gets the input data for a prediction.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>Object</td>
<td>GlideRecord or array of JSON objects containing field names and values as key-value pairs.</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional values for filtering prediction results.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;apply_threshold&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;top_n&quot;: Number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td>options.apply_threshold</td>
<td>Boolean</td>
<td>Flag that indicates whether to check the threshold value for the solution and apply it to the result set. Valid values:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>• true: Return results in which confidence is greater than threshold.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Return all results.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: True</td>
</tr>
</tbody>
</table>

options.top_nNumber   | Number | If provided, returns the top results, up to the specified number of predictions.                                                              |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JSON object containing the prediction results sorted by sys_id or record_number.</td>
</tr>
<tr>
<td></td>
<td>{&lt;identifier&gt;: [Array]}</td>
</tr>
</tbody>
</table>

<Object>.<identifier>  | List of objects with details for each prediction result.                                                                                     |
|                      | <identifier>: [                                                                                                                        |
|                      |   {                                                                                                                                      |
|                      |     "confidence": Number,                                                                                                                |
|                      |     "predictedSysId": "String",                                                                                                         |
|                      |     "predictedValue": "String",                                                                                                          |
|                      |     "threshold": Number                                                                                                                 |
|                      |   }                                                                                                                                       |
|                      | ]                                                                                                                                         |

Data type: Array

<Object>.<identifier>.object | Value associated with the prediction.                                                                                                       |
|                           | For example, 53.84.                                                                                                                        |
|                           | Data type: Number                                                                                                                         |

<Object>.<identifier>.predictedSysId | The sys_id of the predicted value. Results can be from any table on which information is being predicted.                                 |
|                                   | Data type: String                                                                                                                          |

<Object>.<identifier>.predictedValue | Value representing the prediction result.                                                                                                  |
|                                   | Data type: String                                                                                                                          |
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td>&lt;Object&gt;.&lt;identifier&gt;.&lt;object&gt;.threshold</td>
</tr>
<tr>
<td>Value of the configured threshold associated with the prediction.</td>
<td></td>
</tr>
<tr>
<td>Data type: Number</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to display prediction results for a `predict()` method that takes a GlideRecord by `sys_id` for input and includes optional parameters to restrict to top three results and exclude the threshold value.

```javascript
var mlSolution = sn_ml.RegressionSolutionStore.get('ml_incident_categorization');

// single GlideRecord input
var input = new GlideRecord("incident");
input.get("<sys_id>");

// configure optional parameters
var options = {};
options.top_n = 3;
options.apply_threshold = false;

var results = mlSolution.getVersion(1).predict(input, options);
// pretty print JSON results
gs.print(JSON.stringify(JSON.parse(results), null, 2));
```

```javascript
{
    "<sys_id/now_GR>": [
    {
        "confidence": 62.10782320780268,
        "threshold": 20.36,
        "predictedValue": "Clone Issues",
        "predictedSysId": ""
    },
    {
        "confidence": 6.945237375770391,
        "threshold": 16.63,
        "predictedValue": "Instance Administration",
        "predictedSysId": ""
    },
    {
        "confidence": 5.321061076300759,
```
The following example shows how to display prediction results for a `predict()` method that takes an array of field names as key-value pairs for input and includes optional parameters to restrict to top three results and exclude the threshold value.

```javascript
var mlSolution = sn_ml.RegressionSolutionStore.get('ml_incident_categorization');
// key-value pairs input
var input = ["short_description":"my email is not working"], 
{short_description:"need help with password"});
// configure optional parameters
var options = {};
options.top_n = 3;
options.apply_threshold = false;
var results = mlSolution.predict(input, options);
// pretty print JSON results
gs.print(JSON.stringify(JSON.parse(results), null, 2));

{
"i": [
{
"confidence": 37.5023032262591,
"threshold": 10.72,
"predictedValue": "Authentication",
"predictedSysId": ""
},
{
"confidence": 24.439964862166583,
"threshold": 23.7,
"predictedValue": "Administration",
"predictedSysId": ""
},
{
"confidence": 11.736320486031047,
"threshold": 100,
"predictedValue": "Security",
"predictedSysId": ""
}
```
The RenderProperties API provides methods about the current page and is available in Jelly scripts and in UI-action conditions and scripts.

Access RenderProperties methods using the static variable `RP`.

### RenderProperties - getEncodedQuery()

Returns the encoded query from the URL sent to the page.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>
**RenderProperties - getListControl()**

Returns the list control object for the page.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SysListControl object</td>
<td>The list control object for the page.</td>
</tr>
</tbody>
</table>

**RenderProperties -getParameterValue(String parameterName)**

Returns the value of the specified URL parameter.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameterName</td>
<td>String</td>
<td>Name of the parameter passed on the URL.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The parameter's value.</td>
</tr>
</tbody>
</table>

**RenderProperties - getReferringURL()**

Returns the URL where the request originated.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The URL of the page where the request originated.</td>
</tr>
</tbody>
</table>

**RenderProperties - getViewID()**

Returns the view the page is using.

This method is not available in scoped applications.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_id of the view being used.</td>
</tr>
</tbody>
</table>

**RenderProperties - getViewName()**

Returns the name of the view in use.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the view being used.</td>
</tr>
</tbody>
</table>

**RenderProperties - getWindowProperties()**

Returns the window's properties.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The window's properties</td>
</tr>
</tbody>
</table>

**RenderProperties - isInDevStudio()**

Returns true if the page is part of Studio.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the page is part of Studio.</td>
</tr>
</tbody>
</table>

**RenderProperties - isInteractive()**

Returns true if this is an interactive session. An interactive session is when a user has logged in as opposed to a REST request.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if this is an interactive session.</td>
</tr>
</tbody>
</table>
RenderProperties - isManyToMany()

Returns true when the `sysparm_collection_related_file` URL parameter is set.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true when the <code>sysparm_collection_related_file</code> URL parameter is set.</td>
</tr>
</tbody>
</table>

RenderProperties - isRelatedList()

Returns true when the `sys_is_related_list` URL-parameter is true. Returns false if the parameter is not present.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the URL parameter <code>sys_is_related_list</code> is true.</td>
</tr>
</tbody>
</table>

RenderProperties - Scoped

The RenderProperties API provides methods about the current page and is available in Jelly scripts and in UI-action conditions and scripts.

Access `RenderProperties` methods using the static variable `RP`.

Scoped `ScopedRenderProperties - getEncodedQuery()`

Returns the encoded query from the URL sent to the page.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Returns the encoded query from the URL sent to the form.</td>
</tr>
</tbody>
</table>

Scoped ScopedRenderProperties - getListControl()

Returns the list control object for the page.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScopedSysListControl object</td>
<td>The list control object for the page.</td>
</tr>
</tbody>
</table>

Scoped ScopedRenderProperties - getParameterValue(String parameterName)

Returns the value of the specified URL parameter.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameterName</td>
<td>String</td>
<td>Name of the parameter passed on the URL.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The parameter's value.</td>
</tr>
</tbody>
</table>
**Scoped ScopedRenderProperties - getReferringURL()**

Returns the URL where the request originated.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The URL of the page where the request originated.</td>
</tr>
</tbody>
</table>

**Scoped ScopedRenderProperties - getViewName()**

Returns the name of the view in use.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the view being used.</td>
</tr>
</tbody>
</table>

**Scoped ScopedRenderProperties - getWindowProperties()**

Returns the window's properties.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The window's properties</td>
</tr>
</tbody>
</table>

**Scoped ScopedRenderProperties - isInDevStudio()**

Returns true if the page is part of Studio.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the page is part of Studio.</td>
</tr>
</tbody>
</table>

**Scoped ScopedRenderProperties - isInteractive()**

Returns true if this is an interactive session. An interactive session is when a user has logged in as opposed to a REST request.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if this is an interactive session.</td>
</tr>
</tbody>
</table>

**Scoped ScopedRenderProperties - isManyToMany()**

Returns true when the `sysparm_collection_related_file` URL parameter is set.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true when the <code>sysparm_collection_related_file</code> URL parameter is set.</td>
</tr>
</tbody>
</table>

Scoped `ScopedRenderProperties - isRelatedList()`

Returns true when the `sys_is_related_list` URL-parameter is true. Returns false if the parameter is not present.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the URL parameter <code>sys_is_related_list</code> is true.</td>
</tr>
</tbody>
</table>

RequestAuthAPI - Scoped

The `RequestAuthAPI()` API provides methods to apply a signature to a REST request.

Generate outbound signing requests using these APIs in the following order:

1. `HttpRequestData`: Build the API request.
2. `AuthCredential`: Create a credential object or update an existing one. Use the credential to sign the request through the `RequestAuthAPI` class.
3. `RequestAuthAPI`: Sign the request and return an `HttpRequestAuthedData` object.
4. **HttpRequestAuthedData**: Get information about the signed request.

5. **GlideHTTPRequest**: Send the signed request.

Before using these APIs, you must configure an authentication algorithm to sign the request and associate it with the credential used to authenticate the request.

Use this API in scoped scripts with the `sn_auth` namespace identifier.

**RequestAuthAPI - RequestAuthAPI(Object data, Object credential)**

Instantiates a RequestAuthAPI object using a REST request and a credential.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Object</td>
<td>HttpRequestData object that contains the unsigned REST request. Use the HttpRequestData class to build this object. See HttpRequestData API.</td>
</tr>
<tr>
<td>credential</td>
<td>Object</td>
<td>Object that represents a record from the Credentials [discovery_credentials] table. Retrieve this object using the AuthCredential class. See AuthCredential API.</td>
</tr>
</tbody>
</table>

```javascript
// Define the HttpRequestData object
var endpoint= "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addHeader('x-amz-acl' , 'public-read' );

// Get a credential record
var credential = new
    sn_cc.StandardCredentialsProvider().getAuthCredentialByID("88772d0d40990010f8772fdd9ebc8075");

// Instantiate a RequestAuthAPI object
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
```
RequestAuthAPI - generateAuth()

Signs the HttpRequestData object and returns an HttpRequestAuthedData object to use to send in the REST request.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HttpRequestAuthedData</td>
<td>Object. Data to use to obtain information about the signed result. Use the methods in the HttpRequestAuthedData class to interact with the signed result. See HttpRequestAuthedData API.</td>
</tr>
</tbody>
</table>

This example creates a request and gets a credential record to pass to the RequestAuthAPI API. The generateAuth() method then signs the request.

```javascript
// Define the HttpRequest object
var endpoint = "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addHeader('x-amz-acl', 'public-read');

// Get a credential record
gs.info("date:" + httpRequestData.getDate());
var credential = new sn_cc.StandardCredentialsProvider().getAuthCredentialByID("5b61c16f73533300f662cff8af6a74bb");

// Create the RequestAuthAPI object and sign the request
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var output = signingAPI.generateAuth();
```
RequestAuthAPI - getAuthCredential()  
Returns the credential record used to sign the request.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuthCredential</td>
<td>Object. Data to use to obtain information about the credential record used to sign the request. Use the methods in the AuthCredential class to interact with the credential. See AuthCredential API.</td>
</tr>
</tbody>
</table>

// Define the HttpRequestData object
var endpoint = "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addHeader('x-amz-acl', 'public-read');

// Get a credential record
var credential = new sn_cc.StandardCredentialsProvider().getAuthCredentialByID("5b61c16f73533300f662cff8faf6a74b");

// Return the AuthCredential object
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var signingCredential = signingAPI.getAuthCredential();

RequestAuthAPI - getHttpRequestData()  
Returns the HttpRequestData object that was used to instantiate the RequestAuthAPI Class.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HttpRequestData</td>
<td>Object. Contains the unsigned REST request. Use the methods in the HttpRequestData class to interact with the request. See HttpRequestData API.</td>
</tr>
</tbody>
</table>

```javascript
// Define the HttpRequestData object
var endpoint = "https://third-party-endpoint";
var httpRequestData = new sn_auth.HttpRequestData();
httpRequestData.setEndpoint(endpoint);
httpRequestData.setService('s3');
httpRequestData.setRegion('us-east-1');
httpRequestData.setHttpMethod("PUT");
var content = "Action=SendMessage&MessageBody=This is a test message";
httpRequestData.setContent(content);
httpRequestData.addHeader('x-amz-acl', 'public-read');

// Get a credential record
var credential = new sn_cc.StandardCredentialsProvider().getAuthCredentialByID("88772d0d40990010f8772fdd9ebc8075");

// Return the HttpRequestData object
var signingAPI = new sn_auth.RequestAuthAPI(httpRequestData, credential);
var unsignedRequest = signingAPI.getHttpRequestData();
gs.info("Endpoint is: " + unsignedRequest.getEndpoint());
```

Output:

```
Endpoint is: https://third-party-endpoint
```

**RequestAuthAPI - resetAuthCredential()**

Generates a temporary, limited privilege token that you can use to provide trusted users with temporary security credentials to control third-party resources.
To use this method, the third-party service must include an API or SDK that processes and manages limited access tokens.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
// Creates a temporary token
function(authAPI) {
    var requestData = authAPI.getHttpRequestData();
    var credential = authAPI.getAuthCredential();
    credential.setAttribute('Action', 'aws_sessionToken');
    authAPI.resetAuthCredential();

    // Returns status information
    var status = credential.getAttribute('credential_reset_status');
}
```

**ResponseTemplate - Scoped, Global**

Provides methods for managing response templates.

Response templates are reusable messages that agents can copy to case or task forms. They provide quick and consistent messages to users and display standard chat response messages to requesters in Agent Chat. This API requires the Templated Responses plugin (com.sn_templated_snip), which is activated by default. This API is provided within the `sn_templated_snip` namespace.

For additional information on response templates, see [Response templates](#).

**ResponseTemplate - ResponseTemplate()**

Instantiates a new ResponseTemplate object.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

### Example

```python
new
    sn_templated_snip.ResponseTemplate().query("incident","ef4225a40a0b5700d0b8a790747812", 
    ",", 0, 0, true, ");
```

---

**ResponseTemplate - query(String tableName, String recordId, String searchTerm, Number limit, Number offset, Boolean includeEvaluatedBody, String errorFormat, Object opts)**

Returns all response templates from a specified table that match the passed in query criteria.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table to use to search the sn_templated_snip_note_template table to locate corresponding response templates. For example, incident or sn_hr_core_case.</td>
</tr>
<tr>
<td>recordId</td>
<td>String</td>
<td>Sys ID of the record to use to render the variables in the response template.</td>
</tr>
<tr>
<td>searchTerm</td>
<td>String</td>
<td>Optional. Text to use to filter the list of matching response templates. The method performs a CONTAINS search of this text on the name and body fields and a STARTS WITH search on the short name field. For example, if the search term is &quot;crash&quot;, the method returns any response template that matches the query criteria and has crash in the name or body or the short name starts with crash. Response templates with exact matches on short name appear first in the return results. All other returned response templates are sorted by name.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>limit</td>
<td>Number</td>
<td>Optional. Maximum number of response templates to return. Default: 50</td>
</tr>
<tr>
<td>offset</td>
<td>Number</td>
<td>Optional. For pagination, the index at which to start the search. Default: 0</td>
</tr>
</tbody>
</table>
| includeEvaluatedBody | Boolean    | Optional. Flag that indicates whether to render the template variables. Valid values:  
  • false: Do not render the template variables. The response contains the message variables. For example:  
    ```
    Please note that your case ${number} has been escalated to ${assignment_group}.
    ```  
  • true: Renders the template variables and returns `evaluated_response` in the return results. For example:  
    ```
    Please note that your case INC100001 has been escalated to Facilities.
    ```  
  Default: false |
| errorFormat        | String     | Optional. HTML formatting to use for errors. For example:  
  ```
  "<span style='color:#fff000'>${%s}</span>"
  ```  
  Default: "<span style='color:#ff0000'>${%s}</span>" |

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>opts</td>
<td>Object</td>
<td>Optional. Parameters to pass to the sn_templated_snip.response_template extension point. The format and content of these parameters are dependent on the implementation of the extension point. For additional information on extension points, see <a href="#">Using extension points to extend application functionality</a>.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Array of all templates that match the specified search criteria. Response templates with exact matches on short name appear first in the return results. All other returned response templates are sorted by name. Each node in the Array may contain the following parameters:</td>
</tr>
</tbody>
</table>

- **sys_id**: String. Unique identifier of the response template.
- **name**: String. Name of the response template.
- **short_name**: String. Short name of the response template.
- **body**: HTML. Body of the response template.
- **short_name_match**: Boolean. Flag that indicates whether an exact match occurred on the short name of the response template.
- **evaluated_response**: Array. Results of the template evaluation.
  - **success**: Boolean. Flag that indicates whether all variables rendered properly.
  - **evaluated_body**: HTML. Rendered response template body.
  - **error**: Array. Entry for each evaluation error that occurred.
    - **inAccessibleVariables**: String. Variables in the response template body that could not be resolved.
    - **unEvaluatedVariables**: String. Variables in the response template body that were not evaluated.
    - **message**: String. Error message. |
This example shows how to query for response templates associated with the incident table.

query("incident","ef4225a40a0a0b5700d0b8a790747812","",0,0,false,"");

Successful response:

```
[
  {
    "sys_id": "5fc1d65993003300a9bc1d1e867ffbf9c",
    "name": "Incident escalation",
    "short_name": "escalation",
    "template_body": "<p>Dear ${caller_id.first_name},</p>
    <p>Please note that your incident ${number} has been escalated to ${assignment_group}. An agent will be assigned on your case and will keep you updated. If you have more questions please reach out to our team.</p>
    <p>Regards,</p>
    <p>${sys_updated_by}</p>
  }
]
```

Same query with an error response.

query("incident","fe4225a40a0a0b5700d0b8a790747812","",0,0,false,"");

Error response:

```
[
  {
    "sys_id": "5fc1d65993003300a9bc1d1e867ffbf9c",
    "name": "Incident escalation",
    "short_name": "escalation",
    "template_body": "<p>Dear ${caller.first_name},</p>
    <p>Please note that your incident INC0000049 has been escalated to Hardware. An agent will be assigned on your case and will keep you updated. If you have more questions please reach out to our team.</p>
    <p>Regards,</p>
    <p>${sys_updated_by}</p>
    
    "evaluated_response": {
      "success": false,
      "error": {
        "unEvaluatedVariables": "caller.first_name",
        "message": "Cannot evaluate following variables: caller.first_name"
      },
      "evaluated_body": "<p>Dear <span style="color:#ff0000">${caller.first_name}</span></p>
    </span>,
    <p>Please note that your incident INC0000049 has been escalated to Hardware. An agent will be assigned on your case and will keep you updated. If you have more questions please reach out to our team.</p>
    <p>Regards,</p>
    <p>${sys_updated_by}</p>
  }
]
```
ResponseTemplate - render(String templateId, String tableName, String recordId, String errorFormat, Object opts)

Renders the HTML body of a specified response template.

During rendering, all variables are resolved using the information from the specified table and record. If variables cannot be resolved, or any other problem occurs during rendering, the method returns an error message in the results.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>templateId</td>
<td>String</td>
<td>Sys ID of the response template to render.</td>
</tr>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table to use when rendering the variables on the response template.</td>
</tr>
<tr>
<td>recordId</td>
<td>String</td>
<td>Sys ID of the record to use when rendering the variables on the response template. This record must be in the table specified by tableName.</td>
</tr>
<tr>
<td>errorFormat</td>
<td>String</td>
<td>Optional. HTML formatting to use for errors. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;&lt;span style='color:#ffff00'&gt;${%s}&lt;/span&gt;&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: &quot;&lt;span style='color:#ff0000'&gt;${%s}&lt;/span&gt;&quot;</td>
</tr>
<tr>
<td>opts</td>
<td>Object</td>
<td>Optional. Parameters to pass to the sn_templated_snip.response_template extension point. The format and content of these parameters are dependent on the implementation of the extension point. For additional information on extension points, see Using extension points to extend application functionality.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Results of the render.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• success</td>
<td>Flag that indicates whether the render was successful.</td>
</tr>
<tr>
<td>• evaluated_body</td>
<td>String. For success, rendered response template body. For error, response template body which includes both rendered and non-renderable variables.</td>
</tr>
<tr>
<td>• error</td>
<td>Object. Error message if render was unsuccessful.</td>
</tr>
<tr>
<td>◦ unEvaluatedVariables</td>
<td>Variables that could not be rendered.</td>
</tr>
<tr>
<td>◦ message</td>
<td>Error message.</td>
</tr>
</tbody>
</table>

This code example shows how to request a rendered response template for the incident table.

render("5fc1d65993003300a9bcl1d1e867ff9b9c","incident","ef4225a40a0a0b57000d0b8a790747812","")

Successful response:

{
  "success": true,
  "evaluated_body": "Dear Beth,
  
  Please note that your incident INC0000049 has been escalated to Hardware. An agent will be assigned on your case and will keep you updated. If you have more questions please reach out to our team.
  
  Regards,
  
  admin",
}

Same render request but returning an error response.

render("5fc1d65993003300a9bcl1d1e867ff9b9c","incident","ef4225a40a0a0b57000d0b8a790747812","")

Error response:

{
  "success": false,
  "error": {
    "unEvaluatedVariables": "caller.first_name",
    "message": "Cannot evaluate following variables: caller.first_name"
  },
  "evaluated_body": "Dear <span style='color:#ff0000'>${caller.first_name}</span>,
  
  Please note that your incident INC0000049 has been escalated to Hardware. An agent will be assigned on your
RESTAPIRequest - Scoped, Global

A RESTAPIRequest object allows you to access scripted REST API request details in scripts.

⚠️ **Note:** You cannot instantiate objects of this type. Objects of this type are created automatically and are accessible only in scripted REST API resource scripts.

RESTAPIRequest - body

The body of the request.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>RESTAPIRequestBody</td>
<td>The body of the request. You can access data from the body object using the RESTAPIRequestBody API.</td>
</tr>
</tbody>
</table>

```javascript
var requestBody = request.body // Returns instance of RESTAPIRequestBody
```

RESTAPIRequest - getHeader(String header)

Returns the value of a specific header from the web service request.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td>header</td>
<td>String</td>
</tr>
<tr>
<td>The name of the header, such as <code>accept</code> or <code>content-type</code>.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>String</td>
<td>The value of the specified header.</td>
</tr>
</tbody>
</table>
var acceptHeader = request.getHeader('accept');

**RESTAPIRequest - getSupportedResponseContentTypes()**

Get the content types specified in the request Accept header.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of string values where each string is a content type, such as application/json.</td>
</tr>
</tbody>
</table>

**RESTAPIRequest - headers**

All headers from the request.

### Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>headers</td>
<td>object</td>
<td>All headers from the request, and their values.</td>
</tr>
</tbody>
</table>

var headers = request.headers;
var acceptHeader = headers.Accept;
var myCustomHeader = headers.myCustom;
var specialHeader = headers['special - header'];

**RESTAPIRequest - pathParams**

The path parameters passed in the request URI.
Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pathParams</td>
<td>Object</td>
<td>The path parameters as a script object. Available path parameters depend on the web service configuration.</td>
</tr>
</tbody>
</table>

In this example, the scripted REST API endpoint follows this format: https://instance.service-now.com/api/now/myservice/{tableName}/{id}. The request being processed uses this URL: https://instance.service-now.com/api/now/myservice/myApp_table/1234.

```javascript
var pathParams = request.pathParams;
var tableName = pathParams.tableName; //’myApp_table’
var id = pathParams.id; //’1234’
```

**RESTAPIRequest - queryParams**

The query parameters from the web service request.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queryParams</td>
<td>Object</td>
<td>The query parameters from the web service request.</td>
</tr>
</tbody>
</table>

In this example, the request being processed uses this URL: https://<instance_rest_endpoint>?active=false&name=now. Note the active and name parameters.

```javascript
var queryParams = request.queryParams;
var isActiveQuery = queryParams.active; //false
var nameQueryVal = queryParams.name; //’now’
```

**RESTAPIRequest - queryString**

The entire query added to the endpoint URI.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queryString</td>
<td>String</td>
<td>The entire query for the request.</td>
</tr>
</tbody>
</table>
In this example, the request being processed uses this URL: https://<instance_rest_endpoint>?active=false&name=now. Note the query active=false&name=now.

```javascript
var query = request.queryString; //"active=false&name=now"
```

**RESTAPIRequest - uri**

The request URI, excluding domain information.

<table>
<thead>
<tr>
<th>Field</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri</td>
<td>String</td>
<td>The request URI, excluding domain information.</td>
<td></td>
</tr>
</tbody>
</table>

In this example, the request being processed uses this URL: https://instance.service-now.com/api/now/table/myTable?active=false&name=now.

```javascript
var query = request.uri; //"api/now/table/myTable"
```

**RESTAPIRequest - url**

The entire request URL.

<table>
<thead>
<tr>
<th>Field</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>The entire request URL.</td>
<td></td>
</tr>
</tbody>
</table>

In this example, the request being processed uses this URL: https://instance.service-now.com/api/now/table/myTable?active=false&name=now.

```javascript
var query =
request.url; //"https://instance.service-now.com/api/now/table/myTable?active=false&name=now"
```

**RESTAPIRequestBody - Scoped, Global**

A RESTAPIRequestBody object allows you to access the body content of a scripted REST API request in scripts.

The format of a RESTAPIRequestBody object may be JSON or XML, depending on the content-type header value from the request.
Note: You cannot instantiate objects of this type. Objects of this type are created automatically and are accessible only in scripted REST API resource scripts.

Single entry example-request-body in JSON format.

```
{
    "name": "user1",
    "id": 1234,
    "roles": [
    {
        "name": "admin"
    },
    {
        "name": "itil"
    }
    ]
}
```

Multiple entry example-request-body in JSON format.

```
[{  
    "name": "user1",
    "id": 1234,
    "roles": [  
    {
        "name": "admin"
    },  
    {
        "name": "itil"
    }
    ]
 },
{
    "name": "user2",
    "id": 9876,
    "roles": [  
    {
        "name": "admin"
    }
    ]
}]
```
Important: If the request body format is not of a json or xml subtype, use only the request body dataStream field to access the request body. Using request body data, dataString, nextEntry(), or hasNext() with a non-json or non-xml format results in a 500 error response.

RESTAPIRequestBody - data

The content of the request body.

⚠️ Note: REST web services consume data, datastream, and dataString via stream. Because streams are only consumed once, only one can be called. For example, if calling both data and dataString, the second call will return empty.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Object or Array</td>
<td>The request content. This can be a single object or an array of objects depending on the request.</td>
</tr>
</tbody>
</table>

```javascript
var entry;
var id;
var requestBody = request.body;
var requestData = requestBody.data; //May be an array or a single object
if (requestData instanceof Array) {
  entry = requestData[0].name; // 'user1'
  id = requestData[0].id; // '1234'
} else {
  entry = requestData.name; // 'user1'
  id = requestData.id; // '1234'
}
```

RESTAPIRequestBody - dataStream

The content of the request body, as a stream.

⚠️ Note: REST web services consume data, datastream, and dataString via stream. Because streams are only consumed once, only one can be called. For example, if calling both data and dataString, the second call will return empty.
### Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataStream</td>
<td>Object</td>
<td>The content of the request body. You can pass the stream to a separate API, such as to create an attachment from the request or forward the request to a different endpoint.</td>
</tr>
</tbody>
</table>

```javascript
var requestBody = request.body;
var requestStream = requestBody.dataStream;
```

### RESTAPIRequestBody - dataString

The content of the request body, as a String.

> **Note:** REST web services consume data, datastream, and dataString via stream. Because streams are only consumed once, only one can be called. For example, if calling both data and dataString, the second call will return empty.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataString</td>
<td>String</td>
<td>The content of the request body.</td>
</tr>
</tbody>
</table>

```javascript
var requestBody = request.body;
var requestString = requestBody.dataString;
```

### RESTAPIRequestBody - hasNext()

Determines if there are additional entries in the request body.

Use this method with the `nextEntry()` method to iterate over multiple request body entries.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>Flag indicating whether there are additional entries in the request body. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Additional entries are available. As <code>nextEntry()</code> is called, <code>hasNext()</code> continues to be true until the last matching record is loaded.</td>
</tr>
<tr>
<td></td>
<td>• false: No additional entries.</td>
</tr>
</tbody>
</table>

```javascript
var requestBody = request.body;
requestBody.hasNext(); // returns true if the request contains a single entry or multiple entries
requestBody.nextEntry(); // returns next available entry

// calling second time
requestBody.hasNext(); // returns false if the request contains a single entry, or true if the request contains multiple entries
```

**RESTAPIRequestBody - nextEntry()**

Retrieve one entry from the request body as a script object.

Use this method with the `hasNext()` method to iterate over multiple request body entries.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>A single entry from the request body.</td>
</tr>
</tbody>
</table>

```javascript
var requestBody = request.body;
var requestEntry = requestBody.nextEntry(); // returns available entry if there is only one entry, or the first entry if there are multiple.
```
```javascript
var name = requestEntry.name; // ‘user1’

// Calling second time
requestEntry = requestBody.nextEntry(); // returns undefined if there is only one entry, or
the second entry if there are multiple.
```

This example demonstrates using hasNext() with nextEntry().

```javascript
var requestBody = request.body;
while(requestBody.hasNext()){
    var entry = requestBody.nextEntry();
}
```

**RESTAPIResponse - Scoped, Global**

A RESTAPIResponse object allows you to build a RESTful response to a scripted REST API request.

> **Note:** You cannot instantiate objects of this type. Objects of this type are created automatically and are accessible only in scripted REST API resource scripts.

**RESTAPIResponse - getStreamWriter()**

Returns the ResponseStreamWriter for this response, allowing you to write directly to the response stream.

Set the content type and status code using the `setHeaders` and `setStatus` functions prior to calling the `getStreamWriter` function.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESTAPIResponseStream - Scoped, Global</td>
<td>The ResponseStreamWriter for this response. You can use this object to write directly to the response stream.</td>
</tr>
</tbody>
</table>
response.setContentType('application/json');
response.setStatus(200);
var writer = response.getWriter();

**RESTAPIResponse - setBody(Object body)**
Sets the body content to send in the web service response.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>Object</td>
<td>The response body, as a JavaScript object. The body content is automatically serialized to JSON or XML depending on the value of the <code>Accept</code> header passed in the request.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var body = {};
body.name = "incident";
body.number = "1234";
body.caller = {"id": "user1"};
response.setBody(body);
```

```javascript
var bodyArray = [];
var body = {};
body.name = "incident";
body.number = "1234";
body.caller = {"id": "user1"};
bodyArray.push(body);
response.setBody(bodyArray);
```

**RESTAPIResponse - setContentType(String contentType)**
Assigns a value to the Content-Type header in the web service response.
You must set a response content type before writing the response. The content type is set automatically for string responses, based on the request Accept header value.

Setting an invalid content type causes the response to default to JSON. Failing to set a content type results in a status code 500 error when sending a binary response.

See the W3 Content-Type header documentation for more information about this header.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contentType</td>
<td>String</td>
<td>The content type of the response body, such as application/json.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
response.setContentType('application/json');
```

**RESTAPIResponse - setError(Object error)**

Configures the REST response to return an error.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Object</td>
<td>Error object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to return an error from within a scripted REST resource.
var queryParams = request.queryParams;
var userId = String(queryParams.user_id || '');
var fileId = String(queryParams.file_id || '');
if (!userId || !fileId){
    response.setError(new sn_ws_err.BadRequestError('Missing required parameters.'));
    return;
}

RESTAPIResponse - setHeader(String header, String value)
Assign a value to a REST service response header.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>header</td>
<td>String</td>
<td>The header you want to set.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The value to assign the specified header.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

response.setHeader("Location","<URI>");

RESTAPIResponse - setHeaders(Object headers)
Sets the headers for the web service response.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>headers</td>
<td>Object</td>
<td>A JavaScript object listing each header and the value to assign that header.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```
var headers = {};
headers.X-Total-Count=100;
headers.Location='https://instance.service-now.com/<endpoint_to_resource>';
response.setHeaders(headers);
```

RESTAPIResponse - `setLocation(String location)`

Assigns a value to the Location header in the web service response.

See the [W3 Location header documentation](#) for more information about this header.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>String</td>
<td>An absolute URI to redirect the response recipient to.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

RESTAPIResponse - `setStatus(Number status)`

Sets the status code number for the web service response.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>Number</td>
<td>The status code to send in the response, such as 200 to indicate success. Passing a non-numerical value, such as a string, causes the status code to default to 0.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
response.setStatus(200);
```

RESTAPIResponseStream - Scoped, Global

A RESTAPIResponseStream object allows you to write directly to the scripted REST API response stream.

Use RESTAPIResponseStream methods to build web service APIs in the Scripted REST API feature.

ℹ️ **Note:** You cannot instantiate objects of this type. Objects of this type are created automatically and are accessible only in scripted REST API resource scripts.

RESTAPIResponseStream - writeStream(Object stream)

Write an input stream to the response stream.

You must set the content type and status code before calling the `writeStream()` method or the response will fail. You cannot modify these values after calling the `writeStream()` method.

ℹ️ **Note:**

It is the responsibility of the script author to obtain the stream from a third-party service.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stream</td>
<td>Object</td>
<td>An attachment or a response stream from a third-party service.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
The following example is for scoped applications:

```javascript
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
    response.setContentType('application/json');
    response.setStatus(200);

    var gsa = new GlideSysAttachment();
    var attachmentStream = new gsa.getContentStream(<sys_id of attachment>);
    var writer = response.getWriter();
    writer.writeStream(attachmentStream);

})(request, response);
```

The following example is for global applications:

```javascript
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
    response.setContentType('application/json');
    response.setStatus(200);

    var attachmentStream = new GlideSysAttachmentInputStream(<sys_id of attachment>);
    var writer = response.getWriter();
    writer.writeStream(attachmentStream);

})(request, response);
```

**RESTAPIResponseStream - writeString(String data)**

Write string data to the response stream.

You must set the content type and status code before calling the `writeString()` method or the response will fail. You cannot modify these values after calling the `writeString()` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>String</td>
<td>The string to add to the response data.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
response.setContentType('application/json');
response.setStatus(200);
var writer = response.getWriter();
var body = {
    name: user1,
    id: 1234,
    roles: [
        { name: admin },
        { name: itil }
    ]
}
writer.writeString(JSON.stringify(body));
```

RESTMessageV2 - Scoped, Global

The RESTMessageV2 API allows you to send outbound REST messages using JavaScript.

Use the RESTResponseV2 API to manage the response returned by the REST provider.

You can use this API in scoped applications, or within the global scope.

RESTMessageV2 -disableForcedVariableSubstitution()

Disables forced variable substitution in outbound REST messages.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**RESTMessageV2 - execute()**

Sends the REST message to the endpoint.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESTResponse</td>
<td>The response returned by the REST provider.</td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance.

```javascript
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); //Might throw exception if message doesn't exist or not visible due to scope.
var response = sm.execute(); //Might throw exception if http connection timed out or some issue with sending request itself because of encryption/decryption of password.
```

**RESTMessageV2 - executeAsync()**

Sends the REST message to the endpoint asynchronously. The instance does not wait for a response from the web service provider when making asynchronous calls.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>RESTResponse</td>
<td>The response returned by the REST provider.</td>
<td></td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance. When using `executeAsync`, consider processing the response body in a separate business rule to take advantage of the asynchronous call.

```javascript
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); //Might throw exception if message doesn't exist or not visible due to scope.
var response = sm.executeAsync(); //Might throw exception if http connection timed out or some issue with sending request itself because of encryption/decryption of password.
```

**RESTMessageV2 - getEndpoint()**

Gets the URL of the endpoint for the REST message.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The URL of the REST web service provider.</td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance.

```javascript
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); //Might throw exception if message doesn't exist or not visible due to scope.
var endpoint = sm.getEndpoint();
```

**RESTMessageV2 - getRequestBody()**

Gets the content of the REST message body.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>the REST message body.</td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance.

```javascript
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); // Might throw exception if message doesn't exist or not visible due to scope.
var body = sm.getRequestBody();
```

`RESTMessageV2 - getRequestHeader(String headerName)`

Gets the value for an HTTP header specified in the REST message.

By default, this method cannot return the value for a header set automatically by the system. To grant this method access to all headers, set the property `glide.http.log_debug` to true.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>headerName</td>
<td>String</td>
<td>Request header you want to get the value for.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The value of the specified header.</td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance.
```javascript
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); //Might throw exception if
message doesn't exist or not visible due to scope.
var header = sm.getRequestHeader("Accept");
```

**RESTMessageV2 - getRequestHeaders()**

Gets HTTP headers that were set by the REST client and the associated values.

This method does not return headers set automatically by the system. To configure this method to return all headers, set the property glide.http.log_debug to true.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>An Object that maps the name of each header to the associated value.</td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance.

```javascript
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); //Might throw exception if
message doesn't exist or not visible due to scope.
var headers = sm.getRequestHeaders();
```

**RESTMessageV2 - RESTMessageV2()**

Instantiates an empty RESTMessageV2 object.

When using an object instantiated this way, you must manually specify an HTTP method and endpoint.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
var sm = new sn_ws.RESTMessageV2();

**RESTMessageV2 - RESTMessageV2(String name, String methodName)**

Instantiates a RESTMessageV2 object using information from a REST message record.

You must have a REST message record defined before you can use this constructor.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the REST message record.</td>
</tr>
<tr>
<td>methodName</td>
<td>String</td>
<td>The name of the HTTP method to use, such as GET/get or PUT/put - case insensitive.</td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance.

```javascript
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); // Might throw exception if message doesn't exist or not visible due to scope.
```

**RESTMessageV2 - saveResponseBodyAsAttachment(String tableName, String recordSysId, String fileName)**

Configures the REST message to save the returned response body as an attachment record.

When you use this function with a REST message that is sent through a MID server, the MID server user must have any roles required to read and write attachment records, as well as any roles required to read and write records on the table specified in the `tableName` parameter.

The response body does not need to be a binary file to be saved as an attachment. Response bodies using text formats, such as JSON or XML can also be saved. If the instance fails to save the attachment, call `getErrorMessage()` on the related RESTResponseV2 object for error details.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Specify the table that contains the record you want to attach the saved file to.</td>
</tr>
<tr>
<td>recordSysId</td>
<td>String</td>
<td>Specify the sys_id of the record you want to attach the saved file to.</td>
</tr>
<tr>
<td>fileName</td>
<td>String</td>
<td>Specify the file name to give to the saved file.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

(function sampleRESTMessageV2() {
  try{
    var request = new sn_ws.RESTMessageV2();
    request.setHttpMethod('get');

    var attachment_sys_id = '<attachment_record_sys_id>',
    tablename = 'incident',
    recordSysId = '<incident_sys_id>',
    response,
    httpResponseStatus,
    filename = '<filename>';

    //endpoint - ServiceNow REST Attachment API
    request.setEndpoint('https://<instance_name>.service-now.com/api/now/attachment/' +
    attachment_sys_id + '/file');
    request.setBasicAuth('<username>', '<password>');

    //RESTMessageV2 - saveResponseBodyAsAttachment(String tableName, String recordSysId, String fileName)
    request.saveResponseBodyAsAttachment(tablename, recordSysId, filename);

    response = request.execute();
    httpResponseStatus = response.getStatusCode();

    gs.info(" http response status_code: " + httpErrorResponseStatus);
  }
})
catch(ex){
    var message  = ex.getMessage();
    gs.info(message);
}

RESTMessageV2 - saveResponseBodyAsAttachment(String tableName, String recordSysId, String fileName, String encryptContext)

Configures the REST message to save the returned response body as an encrypted attachment record.

When you use this function with a REST message that is sent through a MID server, the MID server user must have any roles required to read and write attachment records, as well as any roles required to read and write records on the table specified in the **tableName** parameter.

The response body does not need to be a binary file to be saved as an attachment. Response bodies using text formats, such as JSON or XML can also be saved. If the instance fails to save the attachment, call **getErrorMessage()** on the related RESTResponseV2 object for error details.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Specify the table that contains the record you want to attach the saved file to.</td>
</tr>
<tr>
<td>recordSysId</td>
<td>String</td>
<td>Specify the sys_id of the record you want to attach the saved file to.</td>
</tr>
<tr>
<td>fileName</td>
<td>String</td>
<td>Specify the file name to give to the saved file.</td>
</tr>
<tr>
<td>encryptContext</td>
<td>String</td>
<td>Specify the sys_id of an encryption context. The saved file is encrypted using this context.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
**RESTMessageV2 - setAuthenticationProfile(String type, String profileId)**

Sets the credentials for the REST message using an existing basic auth or OAuth 2.0 profile.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>The type of authentication profile to use. Valid values are 'basic' to use basic authentication, or 'oauth2' to use OAuth 2.0.</td>
</tr>
<tr>
<td>profileId</td>
<td>String</td>
<td>The sys_id of an authentication profile record. When using basic auth, specify the sys_id of a Basic Auth Configuration [sys_auth_profile_basic] record. When using OAuth 2.0, specify the sys_id of a OAuth Entity Profile [oauth_entity_profile] record.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance.

```javascript
var requestBody;
var responseBody;
var status;
var sm;
try{
    // Might throw exception if message doesn't exist or not visible due to scope.
    sm = new sn_ws.RESTMessageV2("<REST_message_record>", "get");

    //set auth profile to an OAuth 2.0 profile record.
    sm.setAuthenticationProfile('oauth2', '1234adsf123212131123qasdsf');

    sm.setStringParameter("symbol", "NOW");
    sm.setStringParameterNoEscape("xml_data","<data>test</data>");

    //In milliseconds. Wait at most 10 seconds for response from http request.
    sm.setHttpTimeout(10000);
    // Might throw exception if http connection timed out or some issue
} catch(exception) {
    status = sm.getStatus();
    console.log(sm.getResponseBody());
    console.log(sm.getErrorBody());
    console.log(sm.getErrorMsg());
    throw("Error occurred while request", exception);}
```
//with sending request itself because of encryption/decryption of password.
response = sm.execute();
responseBody = response.haveError() ? response.getErrorMessage() : response.getBody();
status = response.getStatusCode();
} catch(ex) {
responseBody = ex.getMessage();
status = '500';
} finally {
requestBody = sm ? sm.getRequestBody():null;
}

### RESTMessageV2 - setBasicAuth(String userName, String userPass)

Sets basic authentication headers for the REST message.

Setting security values using this method overrides basic authentication values defined for the REST message record.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userName</td>
<td>String</td>
<td>Username you want to use to authenticate the REST message.</td>
</tr>
<tr>
<td>userPass</td>
<td>String</td>
<td>Password for the specified user.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

In the following example, replace REST_message_record with the name of the REST message record from your instance.

```javascript
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); //Might throw exception if message doesn't exist or not visible due to scope.
sm.setBasicAuth("username","password");
```
RESTMessageV2 - setEccCorrelator(String correlator)

Associates outbound requests and the resulting response record in the ECC queue. This method only applies to REST messages sent through a MID Server.

The correlator provided populates the **Agent correlator** field on the ECC queue record for the response. Provide a unique correlator for each outbound request to associate the correct results in the ECC queue with the request when designing asynchronous automation through a MID Server.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>correlator</td>
<td>String</td>
<td>Unique identifier</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance.

```javascript
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); //Might throw exception if message doesn't exist or not visible due to scope.
sm.setEccCorrelator("unique_identifier");
```

RESTMessageV2 - setEccParameter(String name, String value)

Overrides a value from the database by writing to the REST message payload. This method only applies to REST messages sent through a MID Server.

Use this method when a value from the REST message in the database is invalid, such as when the endpoint URL is longer than the maximum REST endpoint field length. You can set only the endpoint URL using this method by passing source as the name parameter.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the parameter, such as source.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The value to assign to the specified parameter.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance.

```java
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); //Might throw exception if message doesn't exist or not visible due to scope.
sm.setEccParameter("source","http://very.long.endpoint.url");
```

**RESTMessageV2 - setEndpoint(String endpoint)**

Sets the endpoint for the REST message.

By default, the REST message uses the endpoint specified in the REST message record. Use this method to override this default. You must call this method when using the `RESTMessageV2 - RESTMessageV2()` constructor with no parameters.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>endpoint</td>
<td>String</td>
<td>The URL of the REST provider you want to interface with.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var sm = new sn_ws.RESTMessageV2();
sm.setEndpoint("http://web.service.endpoint");
```

**RESTMessageV2 - setHttpMethod(String method)**

Sets the HTTP method this REST message performs, such as GET or PUT.

You must set an HTTP method when using the `RESTMessageV2 - RESTMessageV2()` constructor with no parameters.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>method</td>
<td>String</td>
<td>HTTP method to perform.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var sm = new sn_ws.RESTMessageV2();
sm.setHttpMethod("post");

**RESTMessageV2 - setHttpTimeout(Number timeoutMs)**

Sets the amount of time the REST message waits for a response from the web service provider before the request times out.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeoutMs</td>
<td>Number</td>
<td>Amount of time, in milliseconds, before the call to the REST provider times out.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance.

var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); //Might throw exception if message doesn't exist or not visible due to scope.
sm.setHttpTimeout(6000);

**RESTMessageV2 - setLogLevel(String level)**

Sets the log level for this message and the corresponding response.
Setting a log level using the RESTMessageV2 API overrides the log level configured on the REST message record. This log level may not apply if the endpoint domain is excluded, or if the property `glide.outbound_http_log.override` is true. To view outbound web service logs, navigate to System Logs > Outbound HTTP Requests.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>level</td>
<td>String</td>
<td>The log level. Valid values are basic, elevated, and all.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var rm = new sn_ws.RESTMessageV2();
rm.setLogLevel('all');
```

**RESTMessageV2 - setMIDServer(String midServer)**

Configures the REST message to communicate through a MID Server.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>midServer</td>
<td>String</td>
<td>Name of the MID Server to use. Your instance must have an active MID Server with the specified name.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance.

```javascript
var sm = new sn_ws.RESTMessageV2("<REST_message_record>", "get"); // Might throw exception if message doesn't exist or not visible due to scope.
sm.setMIDServer("mid_server_name");
```
RESTMessageV2 - setMutualAuth(String profileName)

Sets the mutual authentication protocol profile for the REST message.

Setting a protocol profile using this method overrides the protocol profile selected for the REST message record.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>profileName</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

In the following example, replace *REST_message_record* with the name of the REST message record from your instance.

```java
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); //Might throw exception if message doesn't exist or not visible due to scope.
sm.setMutualAuth("mutual_auth_profile_name");
```

RESTMessageV2 - setQueryParameter(String name, String value)

Appends a parameter to the end of the request URL with the form name=value.

For example, the code

```java
setQueryParameter("sysparm_query","active=true^ORDERBYnumber^ORDERBYDESCcategory");
```

appends the text

```
sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory
```

to the request URL.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>value</td>
</tr>
</tbody>
</table>
var sm = new sn_ws.RESTMessageV2();
//Set up message, including endpoint and authentication
sm.setQueryParameter("sysparm_query","active=true^ORDERBYnumber^ORDERBYDESCcategory");

### RESTMessageV2 - setRequestBody(String body)

Sets the body content to send to the web service provider when using PUT or POST HTTP methods.

When you set the body content using this method, variables in the body are not substituted for parameters from the REST message function record. You must explicitly define all values within the REST message body.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body</td>
<td>String</td>
<td>Request body to send.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var sm = new sn_ws.RESTMessageV2("Update user","post"); //Might throw exception if message doesn't exist or not visible due to scope.
var body = "<Message body content>";
sm.setRequestBody(body);

### RESTMessageV2 - setRequestBodyFromAttachment(String attachmentSysId)

Sets the request body using an existing attachment record.

When you use this function with a REST message that is sent through a MID Server, the MID Server user must have any roles required to read attachment records.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attachmentSysId</td>
<td>String</td>
<td>Sys_id of the Attachment [sys_attachment] record you want to send in this REST message.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

(function sampleRESTMessageV2() {
    try {
        var request = new sn_ws.RESTMessageV2();
        request.setHttpMethod('post');
        request.setEndpoint('<web service endpoint URL>');
        request.setRequestBodyFromAttachment('<attachment sys_id>');

        var response = request.execute();
        var httpResponseStatus = response.getStatusCode();

        gs.info("http response status_code: "+ httpResponseStatus);
    }
    catch (ex) {
        var message = ex.getMessage();
        gs.info(message);
    }
})();

RESTMessageV2 - setRequestHeader(String name, String value)
Sets an HTTP header in the REST message to the specified value.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the header.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value to assign to the specified header.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance.

```java
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); //Might throw exception if message doesn't exist or not visible due to scope.
sm.setRequestHeader("Accept","Application/json");
```

**RESTMessageV2 - setRequestorProfile(String requestorContext, String requestorId)**

Overides the default requestor profile for the REST message in order to retrieve an OAuth access token associated with a different requestor.

This method applies only to REST messages configured to use OAuth 2.0 authentication. This method is optional and is unnecessary in most configurations.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestorContext</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>requestorId</td>
<td>String</td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**RESTMessageV2 - setStringParameter(String name, String value)**

Sets a REST message function variable with the specified name from the REST message record to the specified value.

XML reserved characters in the value are converted to the equivalent escaped characters. Use `setStringParameterNoEscape` to set a variable without escaping XML reserved characters.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the REST message variable. This parameter must be defined in the REST message record before you can assign a value to it.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value to assign the variable.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

In the following example, replace `REST_message_record` with the name of the REST message record from your instance.

```javascript
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); // Might throw exception if message doesn't exist or not visible due to scope.
sm.setStringParameter("s","NOW");
```

**RESTMessageV2 - setStringParameterNoEscape(String name, String value)**

Sets a REST message function variable with the specified name from the REST message record to the specified value.

This method is equivalent to `setStringParameter` but does not escape XML reserved characters.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the REST message variable. This parameter must be defined in the REST message record before you can assign a value to it.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value to assign the variable.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
In the following example, replace REST_message_record with the name of the REST message record from your instance.

```javascript
var sm = new sn_ws.RESTMessageV2("<REST_message_record>","get"); // Might throw exception if message doesn't exist or not visible due to scope.
sm.setStringParameterNoEscape("s","NOW");
```

**RESTResponseV2 - Scoped, Global**

The RESTResponseV2 API allows you to use the data returned by an outbound REST message in JavaScript code.

A RESTResponseV2 object is returned by the RESTMessageV2 methods `execute()` and `executeAsync()`.

You can use this API in scoped applications, or within the global scope.

**RESTResponseV2 - getAllHeaders()**

Returns all headers contained in the response, including any duplicate headers.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>List&lt;GlideHTTPHeader&gt;</td>
</tr>
</tbody>
</table>

The follow code example shows how to call the REST Table API and then lists out all of the headers found in the response.

```javascript
var request = new sn_ws.RESTMessageV2();
request.setEndpoint('http://<SN_Instance_host>/api/now/table/incident');
request.setHttpMethod('GET');
request.setRequestHeader("Accept", "application/json");
var res = request.execute();
if (res.getStatusCode() != 200) {
    gs.info('Bad Request');
}
```
var headers = res.getAllHeaders();
for(var i in headers){
    gs.info(headers[i].name + ': ' + headers[i].value);
}

Response headers

X-Is-Logged-In: true
X-Transaction-ID: e3003cb2db8c
X-Total-Count: 66
X-Content-Type-Options: nosniff
Pragma: no-store,no-cache
Cache-Control: no-cache,no-store,must-revalidate,max-age=-1
Expires: 0
Content-Type: application/json;charset=UTF-8
Date: Fri, 14 May 2021 16:05:10 GMT
Connection: close
Server: ServiceNow
JSESSIONID=5041870726ADADFAB2EE2BF8E7E01596; Path=/; HttpOnly; SameSite=None; Secure
Set-Cookie: glide_user=; Max-Age=0; Expires=Thu, 01-Jan-1970 00:00:10 GMT; Path=/; HttpOnly; SameSite=None; Secure
Set-Cookie: glide_user_session=; Max-Age=0; Expires=Thu, 01-Jan-1970 00:00:10 GMT; Path=/; HttpOnly; SameSite=None; Secure
Set-Cookie: glide_user_route=glide.f1aa2015b3fa5fa8dfe567386cde8fd5; Max-Age=2147483647; Expires=Wed, 01-Jun-2009 19:19:17 GMT; Path=/; HttpOnly; SameSite=None; Secure
Set-Cookie: glide_session_store=AF003CB2DB8CF8103DD9C39D139619C7; Max-Age=1800; Expires=Fri, 14-May-2021 16:35:10 GMT; Path=/; HttpOnly; SameSite=None; Secure
Set-Cookie: BIGipServerpool_ddershem=2643023626.45630.0000; path=/; HttpOnly; Secure; SameSite=None; Secure
Strict-Transport-Security: max-age=63072000; includeSubDomains

RESTResponseV2 - getBody()

Gets the content of the REST response body.

Use this function when you want to get the request body as text content. Do not use this method when saving the response as a binary attachment. If a RESTMessageV2 object called the saveResponseBodyAsAttachment(...) function, using getBody() on the associated RESTResponseV2 object will cause an error. When saving the response as an attachment, if the outbound REST message fails, call getErrorMessage() on the response to retrieve the body content.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>REST response body.</td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.RESTMessageV2("Yahoo Finance","get"); // Might throw exception if message doesn't exist or not visible due to scope.
var response = sm.execute();
var responseBody = response.getBody();
```

RESTResponseV2 - getCookies()

Returns all cookies included in the response.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Array of strings representing cookies. Iterate through the array to perform operations on each cookie.</td>
</tr>
</tbody>
</table>

Display individual cookies from the response.

```javascript
var cookies = response.getCookies();
for (var i = 0; i < cookies.length; i++) {
  gs.info('cookie: ' + cookies.get(i));
}
```

Output:

```
cookie: JSESSIONID=4135AA97A5D12DA22EF614AA2B0CAF8D8.node20; Path=/; Secure; HttpOnly
cookie: SABASESSIONID=370152970.36895.0000; path=/
```
**RESTResponseV2 - getErrorCode()**

Gets the numeric error code if there was an error during the REST transaction.

This error code is specific to the Now Platform, it is not an HTTP error code. Provide this error code if you require assistance from ServiceNow Customer Support.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Numeric error code, such as 1 for socket timeout.</td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.RESTMessageV2("Yahoo Finance","get"); //Might throw exception if message doesn't exist or not visible due to scope.
var response = sm.execute();
var errorCode = response.getErrorCode();
```

**RESTResponseV2 - getErrorMessage()**

Gets the error message if there was an error during the REST transaction.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Error message.</td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.RESTMessageV2("Yahoo Finance","get"); //Might throw exception if message doesn't exist or not visible due to scope.
var response = sm.execute();
var errorMsg = response.getErrorMessage();
```
**RESTResponseV2 - getHeader(String name)**

Gets the value for a specified header.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the header that you want the value for, such as Set-Cookie.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the specified header.</td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.RESTMessageV2("Yahoo Finance","get"); //Might throw exception if message
doesn't exist or not visible due to scope.
var response = sm.execute();
var headerVal = response.getHeader("Content-Type");
```

**RESTResponseV2 - getHeaders()**

Gets all headers returned in the REST response and the associated values.

⚠️ **Note:** If a header is present more than once in the response, such as a Set-Cookie header, this function returns only the last of the duplicate headers. To return all headers including duplicates, use the `getAllHeaders()` function.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object that maps the name of each header to the associated value.</td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.RESTMessageV2("Yahoo Finance","get"); //Might throw exception if message
doesn't exist or not visible due to scope.
```
```javascript
var response = sm.execute();
var headers = response.getHeaders();

var sm = new sn_ws.RESTMessageV2("Yahoo Finance","get"); // Might throw exception if message doesn't exist or not visible due to scope.
var response = sm.execute();
var queryString = response.getQueryString();
```

**RESTResponseV2 - getQueryString()**

Gets the fully-resolved query sent to the REST endpoint.

This query contains the endpoint URL as well as any values assigned to variables in the REST message. Use this method only with responses to direct requests. This method is not supported for requests sent asynchronously, or requests sent using a MID server.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Fully-resolved query.</td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.RESTMessageV2("Yahoo Finance","get"); // Might throw exception if message doesn't exist or not visible due to scope.
var response = sm.execute();
var queryString = response.getQueryString();
```

**RESTResponseV2 - getResponseAttachmentSysid()**

Gets the sys_id value of the attachment created from the response body content.

If the RESTMessageV2 object associated with this response called the `saveResponseBodyAsAttachment(...) function, use getResponseAttachmentSysid()` to get the sys_id of the created attachment record. Use this function when you want to perform additional operations with the new attachment record.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the new attachment record.</td>
</tr>
</tbody>
</table>

**RESTResponseV2 - getStatusCode()**

Gets the numeric HTTP status code returned by the REST provider.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Numeric status code returned by the REST provider, such as 200 for a successful response.</td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.RESTMessageV2("Yahoo Finance","get"); //Might throw exception if message doesn't exist or not visible due to scope.
var response = sm.execute();
var statusCode = response.getStatusCode();
```

**RESTResponseV2 - haveError()**

Indicates if there was an error during the REST transaction.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>Returns true if there was an error, false if there was no error.</td>
</tr>
</tbody>
</table>
var sm = new sn_ws.RESTMessageV2("Yahoo Finance","get"); //Might throw exception if message doesn't exist or not visible due to scope.
var response = sm.execute();
var error = response.haveError();

**RESTResponseV2 - waitForResponse(Number timeoutSecs)**

Set the amount of time the instance waits for a response from the web service provider.

This method overrides the property glide.rest.outbound.ecc_response.timeout for this REST response.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeoutSecs</td>
<td>Number</td>
<td>The amount of time, in seconds, to wait for this response.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var sm = new sn_ws.RESTMessageV2("Yahoo Finance","get"); //Might throw exception if message doesn't exist or not visible due to scope.
var response = sm.executeAsync();
response.waitForResponse(60);

**RoundingInterval - Global**

Handles all the details of rounding intervals for CPU speed and RAM size.

Use in any server-side Discovery script.

**RoundingInterval - getRoundedValue(Number value)**

Returns the value after applying the rounding interval.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Number</td>
<td>The value to round off</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The value after rounding</td>
</tr>
</tbody>
</table>

RoundingInterval - RoundingInterval(String type)

Creates an instance of the RoundingInterval class.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>Can be either CPU or RAM.</td>
</tr>
</tbody>
</table>

ScopedSessionDomain - Client

The ScopedSessionDomain API is a script include that contains client-side methods that provide functionality related to the current session domain.

This API is only available if the Domain Support - Domain Extensions Installer plugin (com.glide.domain.msp_extensions) plugin has been activated in the instance. In addition, the caller must have the admin role to access this API.

ScopedSessionDomain - getCurrentDomainID()

Returns the sys_id of the current domain for the logged-in user session.

The identifier that is returned depends on the domain type and the instantiation of that domain.

- If the user is configured in the global domain, and does not use the domain picker to switch domains, the method returns null.
- If the user uses the domain picker to switch to the global domain, the method returns the string "global".
- For all other domains, the method returns the sys_id of that domain.
To access this method from a client-side script, you must use GlideAjax() to invoke it. To invoke this method from a server-side script, use something similar to the following to instantiate the object and access the method.

```javascript
var ssg = new global.ScopedSessionDomain();
domainID = ssg.getCurrentDomainID();
```

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id of the session domain of the current logged-in user. This is the same information that appears in the domain picker.</td>
</tr>
</tbody>
</table>

This example shows how to call the `getCurrentDomainID()` method from a client-side script.

```javascript
function onLoad() {
  var ga = new GlideAjax("global.ScopedSessionDomain"); // Set the script include
  ga.addParam("sysparm_name", "getCurrentDomainID"); // Set the getCurrentDomainID method
  ga.getXML(getResponse);

  function getResponse(response) {
    var answer = response.responseXML.documentElement.getAttribute('answer');
    alert(answer); // Pops up the sys_id of the domain record
  }
}
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**ScriptableDataStream - Scoped, Global**

Provides methods to interact with a stream of data.

This class can only be used in a server-side script after retrieving a ScriptableDataStream object using one of these APIs:

- The `executeDataStreamAction()` method in the `FlowAPI` class. See `FlowAPI`.
- The `getDataStream()` method in the `ScriptableFlowRunnerResult` class. See `ScriptableFlowRunnerResult`.

After retrieving a ScriptableDataStream object, call the methods in this specific order:

1. Use the `hasNext()` method to determine whether there are more items in the data stream.
2. Use the `next()` method to access the next item in the stream.
3. Use the `getItemIndex()`, `getItemInPageIndex()`, and `getPageIndex()` methods to get information from the stream.
4. Use the `close()` method to close the stream.

⚠️ **Note:** Always wrap data stream logic in a `try/catch` block to catch errors. Always include a `finally` statement that ends with the `close()` method from the `ScriptableDataStream` class to close the data stream and prevent performance issues.

**ScriptableDataStream - close()**

Closes the connection to a data stream. Always call this method after performing any desired operations on a data stream.

You can only call this method on a ScriptableDataStream object returned from the `executeDataStreamAction()` method in the `FlowAPI` class. See `FlowAPI`.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

(function() {

try {
    // Execute Data Stream Action.
    var stream = sn_fd.FlowAPI.executeDataStreamAction('x_snc_my_scope.data_stream_name');
    // Process each item in the data stream
    while (stream.hasNext()) {
        // Get a single item from the data stream.
        var user = stream.next();

        // Only log the first item in each page
        if (stream.getItemInPageIndex() == 0) {
            gs.info('first user on page is ' + user.name);
        }
    }
} catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
} finally {
    stream.close();
}
})();

**ScriptableDataStream - getItemIndex()**

Returns the current index of an item in a data stream.

You can only call this method on a ScriptableDataStream object returned from the `executeDataStreamAction()` method in the FlowAPI class. See `FlowAPI`.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Current index of an item in a data stream using zero-based indexing.</td>
</tr>
</tbody>
</table>

(function() {

    try {

        // Execute Data Stream Action.
        var stream = sn_fd.FlowAPI.executeDataStreamAction('x_my_scope.data_stream_name');

        // Process each item in the data stream
        while (stream.hasNext()) {

            // Get a single item from the data stream.
            var User = stream.next();

            // Use the item. Example:
            // var now_GR = new GlideRecord(<table_name>);
            // now_GR.<field_name> = User.<field_name>;
            // now_GR.insert();

            // By default, this code snippet will terminate after 10 items.
            // Remove or modify this limit after your code has been tested.
            if (stream.getItemIndex() >= 9) {
                break;
            }
        } catch (ex) {
            var message = ex.getMessage();
            gs.error(message);
        } finally {
            stream.close();
        }
    }
})();

**ScriptableDataStream - getItemInPageIndex()**

Returns the current index of an item within the current page in a data stream.
You can only call this method on a ScriptableDataStream object returned from the `executeDataStreamAction()` method in the FlowAPI class. See `FlowAPI`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Current index of an item within the current page in the data stream using zero-based indexing.</td>
</tr>
</tbody>
</table>

```javascript
(function() {
  try {
    // Execute Data Stream Action.
    var stream = sn_fd.FlowAPI.executeDataStreamAction('x_snc_my_scope.data_stream_name');
    // Process each item in the data stream
    while (stream.hasNext()) {
      // Get a single item from the data stream.
      var user = stream.next();

      // Only log the first item in each page
      if (stream.getItemInPageIndex() == 0) {
        gs.info('first user on page is ' + user.name);
      }
    }
  } catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
  } finally {
    stream.close();
  }
})();
```

**ScriptableDataStream - getPageIndex()**

Returns the current index of a page in a data stream.
You can only call this method on a ScriptableDataStream object returned from the `executeDataStreamAction()` method in the FlowAPI class. See `FlowAPI`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Current index of a page in a data stream using zero-based indexing.</td>
</tr>
</tbody>
</table>

```javascript
(function() {
    try {

        // Execute Data Stream Action.
        var stream = sn_fd.FlowAPI.executeDataStreamAction('x_my_scope.data_stream_name');

        // Process each item in the data stream
        while (stream.hasNext()) {

            // Get a single item from the data stream.
            var item = stream.next();

            // Use the item.
            var now_GR = new GlideRecord('incident');
            now_GR.setValue('number', item.id);
            now_GR.setValue('short_description', item.name);
            now_GR.insert();

            // By default, this code snippet will terminate after 5 pages.
            // Remove or modify this limit after testing your code.
            if (stream.getPageIndex() >= 4) {
                break;
            }
        }
    } catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    } finally {
```
stream.close();
}
})

**ScriptableDataStream - hasNext()**

Returns true if there are more items in the data stream.

You can only call this method on a ScriptableDataStream object returned from the `executeDataStreamAction()` method in the FlowAPI class. See FlowAPI.

ℹ️ **Note:** By default, the instance waits for 600 seconds to retrieve a single page of data from a MID Server. If you encounter a timeout when running a Data Stream action through a MID Server, change this default by increasing the `datastream_alternative_env_fetch_page_timeout_seconds` system property.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean    | Flag that determines whether there are more items in the data stream. Values include:  
  - true: There are more items to iterate through in the data stream.  
  - false: There are no more items in the data stream. |

This example creates an incident record for each item returned in the data stream.

```javascript
(function() {
  try {
    // Execute Data Stream Action.
    var stream = sn_fd.FlowAPI.executeDataStreamAction('x_my_scope.data_stream_name');

    // Process each item in the data stream
    while (stream.hasNext()) {
```
// Get a single item from the data stream.
var item = stream.next();

// Use the item.
var now_GR = new GlideRecord('incident');
own_GR.setValue('number',item.id);
own_GR.setValue('short_description',item.name);
own_GR.insert();

// By default, this code snippet will terminate after 10 items.
// Remove or modify this limit after testing your code.
if (stream.getItemIndex() >= 9) {
    break;
}
}
}

}()

ScriptableDataStream - next()
Returns the next item in a data stream.

You can only call this method on a ScriptableDataStream object returned from the executeDataStreamAction() method in the FlowAPI class. See FlowAPI.

💡 Note: By default, the instance waits for 600 seconds to retrieve a single page of data from a MID Server. If you encounter a timeout when running a Data Stream action through a MID Server, change this default by increasing the datastream_alternative_env_fetch_page_timeout_seconds system property.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The next item in the data stream. This object contains the outputs defined by the Data Stream action. To view the Data Stream action outputs, navigate to the Outputs section of the Data Stream action in the Flow Designer interface.</td>
</tr>
</tbody>
</table>

This example creates an incident record for each item returned in the data stream.

```javascript
(function() {
    try {

        // Execute Data Stream Action.
        var stream = sn_fd.FlowAPI.executeDataStreamAction('x_my_scope.data_stream_name');

        // Process each item in the data stream
        while (stream.hasNext()) {

            // Get a single item from the data stream.
            var item = stream.next();

            // Use the item.
            var now_GR = new GlideRecord('incident');
            now_GR.setValue('number', item.id);
            now_GR.setValue('short_description', item.name);
            now_GR.insert();

            // By default, this code snippet will terminate after 10 items.
            // Remove or modify this limit after testing your code.
            if (stream.getItemIndex() >= 9) {
                break;
            }
        }
    }
}
```
**ScriptableFlowRunner - Scoped**

Create a builder object used to define parameters for flow, subflow, and action execution. You can specify a flow to execute in a particular domain. Start the flow, subflow, or action execution directly from the builder and view the results in a ScriptableFlowRunnerResult object.

Use these methods in your server-side scripts with the `sn_fd` namespace identifier.

**API call order**

Build and execute flows, subflows, and actions using these APIs in the following order:

1. **FlowAPI**: Creates a builder object
   
   Use `getRunner()` to instantiate the ScriptableFlowRunner builder object.

2. **ScriptableFlowRunner**: Specify Flow Designer content to run
   
   Use these methods in the following order to create the builder pattern:
   
   1. Use one of the methods `action()`, `datastream()`, `flow()`, or `subflow()` to specify what type of Flow Designer object to build.
   
   2. Use one or more methods such as `addInput()`, `inDomain()`, or `quick()` to specify execution parameters.
   
   3. Use the `run()` method to run the action, flow, or subflow with the provided parameters and return a ScriptableFlowRunnerResult object.

3. **ScriptableFlowRunnerResult**: Retrieve Flow Designer execution details
   
   Use one or more methods such as `getContextId()`, `getOutputs()`, and `getDomainId()` to view execution details.

**Example**

This example shows how to create a ScriptableFlowRunner builder object and uses it to execute an approval action on a specific record. A ScriptableFlowRunnerResult object captures the execution arguments and action outputs.

```javascript
(function() {
})();
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
try {

    var inputs = {};

    inputs['sys_id'] = '57af7aec73d423002728660c4cf6a71c'; // Pass the record's sys_id in as input.

    var result = sn_fd.FlowAPI.getRunner() // Create a ScriptableFlowRunner builder object.
    .action('global.markapproved') // Run the global scope action named markapproved.
    .inForeground()
    .inDomain('TOP/ACME') // Run the action from the TOP/ACME domain.
    .withInputs(inputs)
    .run(); // Run the action and return a FlowRunnerResult object.

    var contextId = result.getContextId(); // Retrieve the context ID from the result
    var dateRun = result.getDate();
    var domainUsed = result.getDomainId(); // Retrieve the Domain ID from the result.
    var flowName = result.getFlowObjectName();
    var flowObjectType = result.getFlowObjectType();

    var outputs = result.getOutputs(); // Retrieve any outputs from the action execution.
    var newApprovalStatus = outputs['approval']; // Echo back the approval status for verification.

} catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
}

)}();

**ScriptableFlowRunner - action(String scopedActionName)**

Identifies the scope and name of the action to execute.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopedActionName</td>
<td>String</td>
<td>Scope and name of the action to execute. For example, <code>global.actionName</code>.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableFlowRunnerBuilder</td>
<td>Builder object used to run a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

This example shows how to create a ScriptableFlowRunner builder object and uses it to execute an approval action on a specific record. A ScriptableFlowRunnerResult object captures the execution arguments and action outputs.

```javascript
(function() {
    try {

        var inputs = {};
        inputs['sys_id'] = '57af7aec73d423002728660c4cf6a71c'; // Pass the record’s sys_id in as input.

        var result = sn_fd.FlowAPI.getRunner() // Create a ScriptableFlowRunner builder object.
            .action('global.markapproved') // Run the global scope action named markapproved.
            .inForeground()
            .inDomain('TOP/ACME') // Run the action from the TOP/ACME domain.
            .withInputs(inputs)
            .run(); // Run the action and return a FlowRunnerResult object.

        var contextId = result.getContextId(); // Retrieve the context ID from the result
        var dateRun = result.getDate();
        var domainUsed = result.getDomainId(); // Retrieve the Domain ID from the result.
        var flowName = result.getFlowObjectName();
        var flowObjectType = result.getFlowObjectType();
    }
})();
```
var outputs = result.getOutputs(); // Retrieve any outputs from the action execution.
var newApprovalStatus = outputs['approval']; // Echo back the approval status for verification.

} catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
}

());

ScriptableFlowRunner - datastream(String scopedDatastreamName)
Identifies the scope and name of the data stream action to execute.

To learn more about data stream actions, see Data Stream actions and pagination.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopedDatastreamName</td>
<td>String</td>
<td>Scope and name of the Data Stream action to execute. For example, global.dataStreamActionName.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableFlowRunner</td>
<td>Builder object used to run a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

This example shows how to run a Data Stream action.

(function() {
    try {

        var result = sn_fd.FlowAPI.getRunner()
            .datastream('global.test_dsa')
            .inForeground()
            .run();

    } catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    }

})();
Output:

*** Script: FlowRunnerResult
Flow Object Name: global.test_dsa
Flow Object Type: datastream
Domain: null
Result Time: 2020-06-08 16:41:13
ContextId: null
Output count: 0

ScriptableFlowRunner - flow(String scopedFlowName)
Identifies the scope and name of the flow to execute.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopedFlowName</td>
<td>String</td>
<td>Scope and name of the flow to execute. For example, global.flowName.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableFlowRunner</td>
<td>Builder object used to run a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

This example shows how to run a flow that logs a message.

```javascript
(function() {
  try {
    var result = sn_fd.FlowAPI.getRunner()
    .flow('global.test_flow')
    .inForeground()
    .run();
  }

  catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
  }

  })();
```
gs.info(result.debug());

} catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
}

})

Output:

*** Script: FlowRunnerResult
Flow Object Name: global.test_flow
Flow Object Type: flow
Domain: null
Result Time: 2020-06-08 16:41:13
ContextId: null
Output count: 0

ScriptableFlowRunner - subflow(String scopedSubflowName)
Identifies the scope and name of the subflow to execute.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopedSubflowName</td>
<td>String</td>
<td>Scope and name of the subflow to execute. For example, global.subflowName.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableFlowRunner</td>
<td>Builder object used to run a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

(function() {
    try {

        var result = sn_fd.FlowAPI.getRunner()
            .subflow('global.test_subflow')
            .inForeground()
            .run();

    } catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    }

})();
gs.info(result);

}) catch (ex) {
  var message = ex.getMessage();
  gs.error(message);
}

})();

Output:

Flow Designer: Cloning a new session to run as as user id: [user_name] from original user session: [user_name]
Flow Designer: Message to log
Flow Designer: Reverting cloned session to original user session: [user_name]
*** Script: FlowRunnerResult
Flow Object Name: global.test_subflow
Flow Object Type: flow
Domain: null
Result Time: 2020-06-08 16:41:13
ContextId: null
Output count: 0

ScriptableFlowRunner - inForeground()

Runs the flow, subflow, or action synchronously. Script execution pauses while the flow object is running.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableFlowRunner</td>
<td>Builder object used to run a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

This example shows how to create a ScriptableFlowRunner builder object and uses it to execute an approval action on a specific record. A
ScriptableFlowRunnerResult object captures the execution arguments and action outputs.

```javascript
(function() {
  try {

    var inputs = {};

    inputs['sys_id'] = '57af7aec73d423002728660c4cf6a71c'; // Pass the record's sys_id in as input.

    var result = sn_fd.FlowAPI.getRunner() // Create a ScriptableFlowRunner builder object.
      .action('global.markapproved')        // Run the global scope action named markapproved.
      .inForeground()
      .inDomain('TOP/ACME')                 // Run the action from the TOP/ACME domain.
      .withInputs(inputs)
      .run();                               // Run the action and return a FlowRunnerResult object.

    var contextId = result.getContextId();  // Retrieve the context ID from the result
    var dateRun = result.getDate();
    var domainUsed = result.getDomainId(); // Retrieve the Domain ID from the result.
    var flowName = result.getFlowObjectName();
    var flowObjectType = result.getFlowObjectType();

    var outputs = result.getOutputs(); // Retrieve any outputs from the action execution.
    var newApprovalStatus = outputs['approval']; // Echo back the approval status for verification.

    } catch (ex) {
      var message = ex.getMessage();
      gs.error(message);
    }
  }()
});
```
ScriptableFlowRunner - inBackground()

Runs the flow, subflow, or action asynchronously. Once the flow object starts running, script execution resumes immediately.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableFlowRunner</td>
<td>Builder object used to run a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

This example shows how to run a flow asynchronously.

```javascript
(function() {
    try {

        var result = sn_fd.FlowAPI.getRunner()
            .flow('global.test_flow')
            .inBackground()
            .run();

        gs.info(result.debug());

    } catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    }

})(function);
```

Output:

Flow Designer: Cloning a new session to run as as user id: [user_name] from original user session: [user_name]
Flow Designer: Message to log
Flow Designer: Reverting cloned session to original user session: [user_name]
*** Script: FlowRunnerResult
Flow Object Name: global.test_flow

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
ScriptableFlowRunner - inDomain(String domainId)

Runs the flow, subflow, or action in the specified domain. Checks to ensure the domain exists and is available.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainId</td>
<td>String</td>
<td>The sys_id or name for the domain of execution for the flow.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableFlowRunner</td>
<td>Builder object used to run a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

This example shows how to create a ScriptableFlowRunner builder object and uses it to execute an approval action on a specific record. A ScriptableFlowRunnerResult object captures the execution arguments and action outputs.

```
(function() {
    try {
        var inputs = {};

        inputs['sys_id'] = '57af7aec73d423002728660c4cf6a71c';  // Pass the record’s sys_id as input.

        var result = sn_fd.FlowAPI.getRunner()  // Create a ScriptableFlowRunner builder object.
            .action('global.markapproved')        // Run the global scope action named markapproved.
            .inForeground()
    }
}
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
inDomain('TOP/ACME') // Run the action from the TOP/ACME domain.

.withInputs(inputs)
.run(); // Run the action and return a FlowRunnerResult object.

var contextId = result.getContextId(); // Retrieve the context ID from the result
var dateRun = result.getDate();
var domainUsed = result.getDomainId(); // Retrieve the Domain ID from the result.
var flowName = result.getFlowObjectName();
var flowObjectType = result.getFlowObjectType();

var outputs = result.getOutputs(); // Retrieve any outputs from the action execution.
var newApprovalStatus = outputs['approval'] // Echo back the approval status for verification.

} catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
}

ScriptableFlowRunner - withInputs(Map inputs)
Adds a collection of inputs. If a name in one of the name-value pairs already exists, the new value replaces the pre-existing value.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputs</td>
<td>Object</td>
<td>Map object containing the name-value pairs that define inputs for the flow, subflow, or action.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableFlowRunner</td>
<td>Builder object used to run a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

Creates an input object and passes the value to the withInputs() method.
(function() {
  try {

    var inputs = {}; 

    inputs['sys_id'] = '57af7aec73d423002728660c4cf6a71c'; // Pass the record’s sys_id in 
    as input.

    var result = sn_fd.FlowAPI.getRunner() // Create a ScriptableFlowRunner builder 
    object.
      .action('global.markapproved') // Run the global scope action named 
    markapproved.
      .inForeground()
      .inDomain('TOP/ACME') // Run the action from the TOP/ACME domain.
      .withInputs(inputs)
      .run(); // Run the action and return a FlowRunnerResult 
    object.

    var contextId = result.getContextId(); // Retrieve the context ID from the result 
    var dateRun = result.getDate();
    var domainUsed = result.getDomainId(); // Retrieve the Domain ID from the result.
    var flowName = result.getFlowObjectName();
    var flowObjectType = result.getFlowObjectType();

    var outputs = result.getOutputs(); // Retrieve any outputs from the action 
    execution.
    var newApprovalStatus = outputs['approval']; // Echo back the approval status for 
    verification.

  } catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
  }
})();

**ScriptableFlowRunner - addInput(String name, Object value)**

Adds a single input. If the name passed as an argument already exists as a 
separate input, the new value replaces the pre-existing value.
This method adds a single input. To create an object and add multiple inputs, use the `withInputs()` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the input for the flow, subflow, or action.</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>The value of the input for the flow, subflow, or action.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ScriptableFlowRunner</code></td>
<td>Builder object used to run a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

This example runs an action that takes a single input called `table_name`.

```javascript
(function() {
  try {

    var result = sn_fd.FlowAPI.getRunner()
      .action('global.test_action')
      .addInput('table_name', 'incident')
      .inForeground()
      .run();

    gs.info(result.debug());

  } catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
  }

})();
```

### Output:

```
Flow Designer: TableName
*** Script: FlowRunnerResult
Flow Object Name: global.test_action
Flow Object Type: action
```
**ScriptableFlowRunner - timeout(Number timeout)**

Sets a timeout for a flow, subflow, or action execution.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeout</td>
<td>Number</td>
<td>Timeout in milliseconds.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableFlowRunner</td>
<td>Builder object used to run a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

This example shows how to run a flow and sets a timeout for two minutes.

```javascript
(function() {
    try {

        var result = sn_fd.FlowAPI.getRunner()
            .flow('global.test_flow')
            .inForeground()
            .timeout(120000)
            .run();

        gs.info(result.debug());

    } catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    }

})();
```

**Output:**

```
Flow Designer: Cloning a new session to run as as user id: [user_name] from original user session: [user_name]
```
Flow Designer: Reverting cloned session to original user session: [user_name]

*** Script: FlowRunnerResult
Flow Object Name: global.test_flow
Flow Object Type: flow
Domain: null
Result Time: 2020-06-08 18:22:35
ContextId: null
Output count: 0

ScriptableFlowRunner - withConnectionAliasOverride(String aliasName, String overrideName)

Overrides the Connections and Credentials alias associated with the flow, action, or subflow.

To learn more about overriding a Connections and Credentials alias, see Supporting multiple connections.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aliasName</td>
<td>String</td>
<td>The name of the alias to override.</td>
</tr>
<tr>
<td>overrideName</td>
<td>String</td>
<td>The name of the alias to use when running the flow, subflow, or action.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableFlowRunner</td>
<td>Builder object used to run a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

This example shows how to run a flow using a different alias than the default associated with the flow.

```javascript
(function() {
  try {

    var result = sn_fd.FlowAPI.getRunner()
      .flow('global.test_flow')
      .withConnectionAliasOverride('sn_original_alias.spoke', 'x_new_alias.spoke')
      .inForeground()
      .run();

```
ScriptableFlowRunner - quick()

Runs a flow, subflow, or action from a server-side script synchronously or asynchronously without creating execution details or other related records. Improves performance by eliminating record-keeping overhead. Use this API to increase the speed of high-volume processing, for example multiple executions per second, in a production environment.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableFlowRunner</td>
<td>Builder object used to run a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

This example shows how to run a flow without creating any related records.

```javascript
(function() {
  try {
    var result = sn_fd.FlowAPI.getRunner()
      .flow('global.test_flow')
      .inForeground()
      .quick()
      .run();

    gs.info(result);
  }

  catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
  }
})();
```
Output:

Flow Designer: Message to log.
*** Script: [object FlowRunnerResult]

**ScriptableFlowRunner - run()**

Runs the flow, subflow, or action with the specified parameters.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScriptableFlowRunnerResult</td>
<td>Object containing the execution details of a Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

This example shows how to create a ScriptableFlowRunner builder object and uses it to execute an approval action on a specific record. A ScriptableFlowRunnerResult object captures the execution arguments and action outputs.

```javascript
(function() {
  try {
    var inputs = {};

    inputs['sys_id'] = '57af7aec73d423002728660c4cf6a71c';  // Pass the record’s sys_id in as input.

    var result = sn_fd.FlowAPI.getRunner()  // Create a ScriptableFlowRunner builder object.
    .action('global.markapproved')        // Run the global scope action named markapproved.
    .inForeground()
```
inDomain('TOP/ACME') // Run the action from the TOP/ACME domain.

.withInputs(inputs)
.run(); // Run the action and return a FlowRunnerResult object.

var contextId = result.getContextId(); // Retrieve the context ID from the result
var dateRun = result.getDate();
var domainUsed = result.getDomainId(); // Retrieve the Domain ID from the result.
var flowName = result.getFlowObjectName();
var flowObjectType = result.getFlowObjectType();

var outputs = result.getOutputs(); // Retrieve any outputs from the action execution.
var newApprovalStatus = outputs['approval']; // Echo back the approval status for verification.

} catch (ex) {
  var message = ex.getMessage();
  gs.error(message);
}
}}();

**ScriptableFlowRunnerResult - Scoped**

Captures the result of using ScriptableFlowRunner to execute a flow, subflow, or action. Includes data such as the context ID, domain, and any outputs from the flow execution.

Use these methods in your server-side scripts with the `sn_fd` namespace identifier.

**API call Order**

Build and execute flows, subflows, and actions using these APIs in the following order:

1. **FlowAPI**: Creates a builder object
   
   Use `getRunner()` to instantiate the ScriptableFlowRunner builder object.

2. **ScriptableFlowRunner**: Specify Flow Designer content to run
   
   Use these methods in the following order to create the builder pattern:
1. Use one of the methods `action()`, `datastream()`, `flow()`, or `subflow()` to specify what type of Flow Designer object to build.

2. Use one or more methods such as `addInput()`, `inDomain()`, or `quick()` to specify execution parameters.

3. Use the `run()` method to run the action, flow, or subflow with the provided parameters and return a `ScriptableFlowRunnerResult` object.

**3. ScriptableFlowRunnerResult: Retrieve Flow Designer execution details**

Use one or more methods such as `getContextId()`, `getOutputs()`, and `getDomainId()` to view execution details.

**Example**

This example shows how to create a `ScriptableFlowRunner` builder object and uses it to execute an approval action on a specific record. A `ScriptableFlowRunnerResult` object captures the execution arguments and action outputs.

```javascript
(function() {
  try {

    var inputs = {};

    inputs['sys_id'] = '57af7aec73d423002728660c4cf6a71c';  // Pass the record’s sys_id in as input.

    var result = sn_fd.FlowAPI.getRunner()  // Create a ScriptableFlowRunner builder object.
      .action('global.markapproved')        // Run the global scope action named markapproved.
      .inForeground()
      .inDomain('TOP/ACME')                 // Run the action from the TOP/ACME domain.
      .withInputs(inputs)
      .run();                               // Run the action and return a FlowRunnerResult object.

    var contextId = result.getContextId();  // Retrieve the context ID from the result
    var dateRun = result.getDate();
    var domainUsed = result.getDomainId();  // Retrieve the Domain ID from the result.
    var flowName = result.getFlowObjectName();
  }
});
```
var flowObjectType = result.getFlowObjectType();

var outputs = result.getOutputs(); // Retrieve any outputs from the action execution.
var newApprovalStatus = outputs['approval']; // Echo back the approval status for verification.

} catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
}

());

**ScriptableFlowRunnerResult - getOutputs()**

Returns the outputs of a completed Flow Designer action, flow, or subflow.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing the output of a completed Flow Designer action, flow, or subflow.</td>
</tr>
</tbody>
</table>

This example shows how to retrieve the outputs from a Flow Designer action, flow, or subflow run with the `ScriptableFlowRunner` API.

(function() {
    try {

        var result = sn_fd.FlowAPI.getRunner()
            .action('global.test_action')
            .inForeground()
            .timeout(12000)
            .run();

        gs.info(result.getOutputs());
    }

})();
Output:
Flow Designer: Warning. This is an important log message.

**ScriptableFlowRunnerResult - getFlowObjectType()**

Returns the type of Flow Designer object run.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>FlowObjectType</td>
</tr>
</tbody>
</table>

This example shows how to retrieve the flow object type from the ScriptableFlowRunnerResult API.

```javascript
(function() {
  try {

    var result = sn_fd.FlowAPI.getRunner()
      .flow('global.test_flow')
      .inForeground()
      .timeout(12000)
      .run();

    gs.info(result.getFlowObjectType());

  } catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
  }

})();
```
ScriptableFlowRunnerResult - getFlowObjectName()

Returns the scope and internal name of the Flow Designer action, flow, or subflow run.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The scope and internal name of the Flow Designer action, flow, or subflow run. For example, <code>global.emailflow</code>.</td>
</tr>
</tbody>
</table>

This example shows how to retrieve the name of the flow, subflow, or action name from a ScriptableFlowRunnerResult object.

```javascript
(function() {
    try {

        var result = sn_fd.FlowAPI.getRunner()
            .flow('global.test_flow')
            .inForeground()
            .timeout(12000)
            .run();

        gs.info(result.getFlowObjectName());

    } catch (ex) {
        var message = ex.getMessage();
        gs.error(message);
    }

})();
```
Output:

```
global.test_flow
```

**ScriptableFlowRunnerResult - getDomainId()**

Returns the sys_id of the domain that the Flow Designer action, flow, or subflow ran in.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_id of the domain that the Flow Designer action, flow, or subflow ran in.</td>
</tr>
</tbody>
</table>

This example shows how to retrieve a domain ID from a ScriptableFlowRunnerResult object.

```
(function() {
  try {
    var result = sn_fd.FlowAPI.getRunner()
      .flow('global.test_flow')
      .inForeground()
      .inDomain('TOP/ACME')
      .timeout(12000)
      .run();

    gs.info(result.getDomainId());
  }
  catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
  }

})();
```

Output:

```
4ecead85a4da1110598d0c7d6bf75554
```
### ScriptableFlowRunnerResult - `getDate()`

Returns the date and time when a Flow Designer action, flow, or subflow ran as a GlideDateTime object.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDateTime</td>
<td>The execution date and time for the flow, subflow, or action.</td>
</tr>
</tbody>
</table>

This example shows how to retrieve the date and time of a flow execution from a ScriptableFlowRunnerResult object.

```javascript
(function() {
  try {
    var result = sn_fd.FlowAPI.getRunner()
    .flow('global.test_flow')
    .inForeground()
    .timeout(12000)
    .run();

    gs.info(result.getDate());
  }
  catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
  }

})();
```

**Output:**

```
2020-05-22 18:45:42
```

### ScriptableFlowRunnerResult - `getDataStream()`

Returns the stream of data from a data stream action.
If the `datastream()` method was used in the `ScriptableFlowRunner` builder class, this returns the stream of data as a `ScriptableDataStream` object. Use the `ScriptableDataStream` class to iterate over items in the stream. See `ScriptableDataStream`.

For more information about data stream actions, see Data Stream actions and pagination.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ScriptableDataStream</code></td>
<td>A <code>ScriptableDataStream</code> object you can use to iterate through items in a data stream. Use the methods in the <code>ScriptableDataStream</code> class to interact with this object. See <code>ScriptableDataStream</code>.</td>
</tr>
</tbody>
</table>

This example shows how to retrieve a data stream from a `ScriptableFlowRunnerResult` object.

```javascript
var datastream = result.getDataStream();
```

### `ScriptableFlowRunnerResult - getContextId()`

Returns the context ID of the flow, subflow, or action.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>String</code></td>
<td>The <code>sys_id</code> of the Flow Designer execution details record for the action, flow, or subflow.</td>
</tr>
</tbody>
</table>
This example shows how to retrieve a context ID from a ScriptableFlowRunnerResult object.

```javascript
var contextId = result.getContextId();
```

Output:

```
4ecead85c4da1110598d0c7d6bf73554
```

**ScriptableFlowRunnerResult - debug()**

Returns information about the executed flow, subflow, or action, including the context ID, domain ID, and execution outputs.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String | Execution details about the Flow Designer action, flow, or subflow run.  
- flow object name: Name of the flow, subflow, or action.  
- flow object type: Flow, subflow, action, or datastream action.  
- domain ID: ID of the domain that the flow, subflow, or action ran in.  
- result time: Amount of time it took to run.  
- context ID: Sys_id of the Flow Designer execution details record for the action, flow, or subflow.  
- output count: Number of action or subflow outputs. |

This example shows how to retrieve information about the executed flow, subflow, or action from the ScriptableFlowRunnerResult object.

```javascript
(function() {
  try {
    
    var result = sn_fd.FlowAPI.getRunner()
    .flow('global.test_flow')
  }

  catch (err) {
    console.log(err);
  }
}
```
.inForeground()
.timeout(12000)
.run();

gs.info(result.debug());

} catch (ex) {
  var message = ex.getMessage();
  gs.error(message);
}

})();

Output:

*** Script: FlowRunnerResult
Flow Object Name: global.test_flow
Flow Object Type: flow
Domain: null
Result Time: 2020-06-08 18:28:41
ContextId: null
Output count: 0

ScriptLoader - Client
Provides the ability to load scripts asynchronously.

You can use the ScriptLoader API in client-side scripts for a platform/desktop UI using ListV2 and ListV3 APIs. It is not available for Service Portal, Mobile, or Agent Workspace.

You access the ScriptLoader methods by using the global object ScriptLoader.

ScriptLoader - getScripts(String filePath, Function callback)
Gets scripts asynchronously.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filePath</td>
<td>String</td>
<td>Path, including the file name, that contains one or more scripts.</td>
</tr>
<tr>
<td>callback</td>
<td>Function</td>
<td>Function to call after the scripts have been loaded. This callback function should not have arguments.</td>
</tr>
</tbody>
</table>
This example shows how load a utility script and then run the `getDepartment()` function.

```javascript
// Client script to load a utility script and run the getDepartment callback function
ScriptLoader.getScripts('sn_ui_script_util.Utilities.jsdbx', getDepartment);

function getDepartment() {
  var req = sn_ui_script_util.Utilities.rest('json');
  req.addParam("sysparm_query", "sys_id=" + newValue);
  req.addParam("sysparm_fields", "department");
  req.addParam("sysparm_display_value", true);
  req.success(updateNotes);
  req.get("/api/now/table/sys_user");
}

function updateNotes(data) {
  g_form.setValue("work_notes", data.result[0].department.display_value);
}
```

**ScriptLoader - getScripts(Array scripts, Function callback)**

Loads scripts asynchronously.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scripts</td>
<td>Array</td>
<td>Array of scripts to load.</td>
</tr>
<tr>
<td>callback</td>
<td>Function</td>
<td>Function to call when the scripts have been loaded. The callback function must not have any arguments.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
SentimentAnalyser - Scoped

The SentimentAnalyser API performs sentiment analysis on a string value.

The Sentiment Analysis API should be used in a script that is treated as an admin-executing script. For example, we should use the Sentiment Analysis API in Script Action or Scheduled Job.

To use this class in a scoped application, use the sn_nlp_sentiment namespace identifier. The Sentiment Analysis plugin (com.snc.sentiment_analysis) must be enabled to access the SentimentAnalyser API.

SentimentAnalyser - SentimentAnalyser()

Creates an instance of the SentimentAnalyser class with the default connector configuration that is used for sentiment analysis.

```javascript
var sa = new sn_nlp_sentiment.SentimentAnalyser();
```

SentimentAnalyser - SentimentAnalyser(GlideRecord configGR)

Creates an instance of the SentimentAnalyser class with the specified connector configuration that is used for sentiment analysis.

```javascript
var sa = new sn_nlp_sentiment.SentimentAnalyser(configGR);
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>configGR</td>
<td>GlideRecord</td>
<td>GlideRecord object of a connector configuration.</td>
</tr>
</tbody>
</table>
SentimentAnalyser - analyze(String inputText)
Performs sentiment analysis on the specified text.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputText</td>
<td>String</td>
<td>Text on which sentiment analysis should be performed.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSON object</td>
<td>Result of the sentiment analysis specifying the status, score, normalised score, sys_id of the relevant connector configuration, and error message.</td>
</tr>
</tbody>
</table>

```javascript
var sa = new sn_nlp_sentiment.SentimentAnalyser();
var result = sa.analyze("Example string");
```

Output:

```json
{"status": "Success", "score": "0.7", "normalizedScore": "0.7", "connectorConfig": "10932aa773101300734e234fff6a777", "errorMessage": ""}
```

SentimentAnalyser - analyzeWithLanguage(String inputText, String language)
Performs sentiment analysis on a specified text and language.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputText</td>
<td>String</td>
<td>Text on which to perform sentiment analysis.</td>
</tr>
<tr>
<td>language</td>
<td>String</td>
<td>Language for the input text. This can very for different sentiment services.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSON object</td>
<td>Result of the sentiment analysis specifying the status, score, normalized score, sys_id of the relevant connector configuration, and error message.</td>
</tr>
</tbody>
</table>

```javascript
var sa = new sn_nlp_sentiment.SentimentAnalyser();
var result = sa.analyze("Example string", "en");
```

Output:

```javascript
{"status": "Success", "score": "0.7", "normalizedScore": "0.7", "connectorConfig": "10932aa773101300734e234fff6a777", errorMessage:""}
```

**SentimentAnalyser - analyzeMultiple(Array inputTextArray)**

Performs sentiment analysis on an array of strings.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputTextArray</td>
<td>Array</td>
<td>Array of text (string) on which to perform sentiment analysis.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSON Array</td>
<td>An array that gives the result of the sentiment analysis performed on multiple texts specifying the status, score, normalized score, sys_id of the relevant connector configuration, and error message.</td>
</tr>
</tbody>
</table>

```javascript
var sa = new sn_nlp_sentiment.SentimentAnalyser();
var result = sa.analyzeMultiple ("["Example string1","Example string2"]");
```

Output:

```javascript
[{"text": "I am happy","result": {"status": "Success", "score": "0.7", "normalizedScore": "0.7", "connectorConfig": "10932aa773101300734e234fff6a777", errorMessage:""}},{"text": "I am not happy","result": {"status": "Success", "score": "-0.7", "normalizedScore": "-0.7", "connectorConfig": "10932aa773101300734e234fff6a777", errorMessage:""}}]
```
SentimentAnalyser - analyzeMultipleWithLanguage(Array inputTextArray, String language)

Performs sentiment analysis on an array of strings in the specified language.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputTextArray</td>
<td>Array</td>
<td>Array of text (string) on which to perform sentiment analysis.</td>
</tr>
<tr>
<td>language</td>
<td>String</td>
<td>Language for the input text. This can vary for different sentiment services.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSON</td>
<td>An array with the result of the sentiment analysis performed on multiple texts of the mentioned language, specifying the status, score, normalized score, sys_id of the relevant connector configuration, and error message.</td>
</tr>
</tbody>
</table>

```javascript
var sa = new sn_nlp_sentiment.SentimentAnalyser();
var result = sa.analyzeMultipleWithLanguage ("Example string1","Example string2", "en");
```

Output:

```json
[{
  "text": "I am happy",
  "result": {Success", "score": "0.7", "normalizedScore": "0.7",
    "connectorConfig": "10932aa773101300734e234fff6a777", "errorMessage":""}},
{
  "text": "I am not happy",
  "result": {Success", "score": "-0.7", "normalizedScore": "-0.7",
    "connectorConfig": "10932aa773101300734e234fff6a777", "errorMessage":""}}
```

SentimentAnalyser - getConnectorByName(String connectorName)

Returns the GlideRecord of the specified connector configuration.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectorName</td>
<td>String</td>
<td>Name of the connector configuration.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>GlideRecord of the specified connector configuration.</td>
</tr>
</tbody>
</table>

```javascript
var sa = new sn_nlp_sentiment.SentimentAnalyser();
var connector = sa.getConnectorByName("xxx");
```

Output:

GlideRecord object of the connector configuration with name "xxx", null if no connector is named as "xxx".

**SentimentAnalyser - getDefaultConnector()**

Returns the GlideRecord of the default connector configuration.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>GlideRecord of the default connector configuration.</td>
</tr>
</tbody>
</table>

```javascript
var sa = new sn_nlp_sentiment.SentimentAnalyser();
var defaultConnector = sa.getDefaultConnector();
```

**SerialNumberManager - Global**

Manages the serial numbers for discovery devices.
Use with a discovery script where you need to manage serial numbers.

**SerialNumberManager - add(Number sType, Number value)**

Adds the specified serial number.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sType</td>
<td>Number</td>
<td>The serial number type</td>
</tr>
<tr>
<td>value</td>
<td>Number</td>
<td>The serial number</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### SerialNumberManager - get()

Returns the serial number.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The serial number</td>
</tr>
</tbody>
</table>

### SerialNumberManager - getSerialsForCIData()

Returns serial number information for CI Data, including the type, serial number, and validity.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array with the serial number information in the following order: serial number type, serial number, and validity flag.</td>
</tr>
</tbody>
</table>

**SerialNumberManager - isValid(Number value)**

Checks if the number is a valid serial number.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Number</td>
<td>The number to check</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the number is valid; otherwise, false.</td>
</tr>
</tbody>
</table>

**ServiceRegistryQuery - Global**

Maps Shazzam service registry query results against a port probe. Child classes provide specifics on mapping.

Use this API during the discovery scanning phase.

**ServiceRegistryQuery - findForPortProbe(PortProbe portProbe)**

Parses an array of IDs for the given port probe.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>portProbe</td>
<td>PortProbe</td>
<td>The port probe</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
ServiceRegistryQuery - get(Object source)
Retrieves a registry service by id, regardless of child class.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>Object</td>
<td>The registry service GlideRecord or sys_id.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The registry service</td>
</tr>
</tbody>
</table>

ServiceRegistryQuery - getShazzamQuerierClassname()
Returns the Shazzam! querier class name.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The class name</td>
</tr>
</tbody>
</table>

ServiceRegistryQuery - toPortProbeMap()
Retrieves the port probe XML and creates a map.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The port probe map</td>
</tr>
</tbody>
</table>

**ServiceRegistryQuery - toPortProbeXml()**

Creates XML output for Shazzam! port probes.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The XML output for the probe.</td>
</tr>
</tbody>
</table>

**ServiceRegistryQuery - ServiceRegistryQuery(Object source)**

Creates an instance of the ServiceRegistryQuery class.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>Object</td>
<td>A GlideRecord or sys_id of the registry service.</td>
</tr>
</tbody>
</table>

**ShazzamLaunch - Global**

Handles the details of launching a Shazzam probe for a given discovery.

Use this API to launch a Shazzam probe during the discovery scanning phase.

**ShazzamLaunch - addPortProbe(String midPPs, String portProbes, IPIncludeExcludeCollection ipiec)**

Adds a single port probe to our MID services collection.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>midPPs</td>
<td>String</td>
<td>A hashmap with two properties.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• midServer: a MIDServer instance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• portProbes: a hashmap (by port probe name) of DiscoveryPortProbe instances.</td>
</tr>
<tr>
<td>portProbes</td>
<td>String</td>
<td>A hashmap (by MIDServer name) that contains these properties.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• range: An IPMetaCollection containing the ranges to be probed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• midserver: A MIDServer instance for the MID server to be probed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• portProbes: a hashmap (by port probe name) of DiscoveryPortProbe instances to be probed.</td>
</tr>
<tr>
<td>ipiec</td>
<td>IPIncludeExcludeCollection</td>
<td>The range to be probed.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**ShazzamLaunch - addRange(IPCollection ipc, String portProbes, Boolean samePhase)**

Adds the specified range and this discovery’s port probes to scan (as defined by behavior) to the collection of MID servers and port probes that need to be probed.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipc</td>
<td>IPCollection</td>
<td>The range to be discovered</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| portProbes | String | A hashmap (by MID server name) of hashmaps containing these properties.  
• range: an IPMetaCollection containing the ranges to be probed.  
• midServer: a Mid server instance for the MID server to be probed.  
• portProbes: a hashmap (by port probe name) of DiscoveryPortProbe instances to be probed. |
| samePhase  | Boolean| True if this should use the same phase as is currently running. |

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### ShazzamLaunch - fireShazzamProbe(String portProbe, String ip, String sensor)

Fires a Shazzam probe for the MID server and services in the given hash map.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| portProbe  | String | A hashmap containing these properties.  
• range: an IPMetaCollection containing the ranges to be probed.  
• midServer: a MIDServer instance for the MID server to be probed. |
<p>| ip         | String | A comma-separated list of IPs to fire probes for. |
| sensor     | String | The sys_id of the sensor that launched this probe (network discovery only). |</p>
<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

**ShazzamLaunch - launch(String ip, Boolean samePhase, String sensor)**

Launches Shazzam probes as necessary, depending on the current phase of each behavior used.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip</td>
<td>String</td>
<td>A comma-separated list of IPs to fire probes for.</td>
</tr>
<tr>
<td>samePhase</td>
<td>Boolean</td>
<td>True if this launch should use the same phase as is currently running.</td>
</tr>
<tr>
<td>sensor</td>
<td>String</td>
<td>The sys_id of the sensor that launched this (only in network discovery).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>Boolean</td>
</tr>
</tbody>
</table>

**ShazzamLaunch - makeProbeSpec(String portProbe)**

Makes an XML port probe spec for the given port probe hashmap.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>portProbe</td>
<td>String</td>
<td>A hashmap containing these properties.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• range: an IPMetaCollection containing the ranges to be probed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• midServer: a MIDSserver instance for the MID server to be probed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• portProbes: a hashmap (by port probe name) of DiscoveryPortProbe instances to be probed.</td>
</tr>
</tbody>
</table>
### ShazzamLaunch - ShazzamLaunch(DiscoveryStatus status, DiscoverySchedule schedule)

Creates an instance of the ShazzamLaunch class.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>DiscoveryStatus</td>
<td>The DiscoveryStatus instance for this discovery.</td>
</tr>
<tr>
<td>schedule</td>
<td>DiscoverySchedule</td>
<td>The DiscoverySchedule instance for this discovery.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### ShazzamResult - Global

Encapsulates a Shazzam result.

Use this API during the discovery scanning phase.

### ShazzamResult - active

Property that shows if the IP address is active

#### Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>Boolean</td>
<td>True if this IP address is active, which means at least one port is open.</td>
</tr>
</tbody>
</table>
ShazzamResult - alive

Property is true if this IP address is alive, which means no ports are open, but at least one responded.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alive</td>
<td>Boolean</td>
<td>True if the IP address is alive, meaning no ports are open, but at least one port responded.</td>
</tr>
</tbody>
</table>

ShazzamResult - domainName

The Windows domain name for the IP.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainName</td>
<td>String</td>
<td>The Windows domain name for the IP if one was resolved by the scanner.</td>
</tr>
</tbody>
</table>

ShazzamResult - hostName

Host name for the IP.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hostName</td>
<td>String</td>
<td>Host name of the IP if one was resolved by the scanner.</td>
</tr>
</tbody>
</table>

ShazzamResult - ip

IP address of the results.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip</td>
<td>String</td>
<td>IP address of the results</td>
</tr>
</tbody>
</table>
ShazzamResult - scanners
A JavaScript array of ShazzamScanner instances.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scanners</td>
<td>Array</td>
<td>Array of ShazzamScanner instances</td>
</tr>
</tbody>
</table>

ShazzamScanner - Global
Encapsulates a Shazzam scanner.
Use this API during the discovery scanning phase.

ShazzamScanner - contents
A hash map of any nodes contained within the scanner, by name.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contents</td>
<td>Hash map</td>
<td>A hash map of any nodes contained within the scanner, by name.</td>
</tr>
</tbody>
</table>

ShazzamScanner - domainName
The Windows domain name, if one was resolved by the scanner.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainName</td>
<td>String</td>
<td>The Windows domain name, if one was resolved by the scanner.</td>
</tr>
</tbody>
</table>

ShazzamScanner - hostName
The IP hostname, if one was resolved by the scanner.
<table>
<thead>
<tr>
<th>Field</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>hostName</td>
<td>String</td>
<td>The IP hostname, if one was resolved by the scanner.</td>
</tr>
<tr>
<td>ShazzamScanner - name</td>
<td>name</td>
<td>String</td>
<td>The scanner name.</td>
</tr>
<tr>
<td>Field</td>
<td>name</td>
<td>String</td>
<td>The scanner name.</td>
</tr>
<tr>
<td>ShazzamScanner - port</td>
<td>port</td>
<td>String</td>
<td>The port being scanned.</td>
</tr>
<tr>
<td>Field</td>
<td>port</td>
<td>String</td>
<td>The port being scanned.</td>
</tr>
<tr>
<td>ShazzamScanner - portProbe</td>
<td>portProbe</td>
<td>String</td>
<td>The port probe name.</td>
</tr>
<tr>
<td>Field</td>
<td>portProbe</td>
<td>String</td>
<td>The port probe name.</td>
</tr>
<tr>
<td>ShazzamScanner - protocol</td>
<td>protocol</td>
<td></td>
<td>The scanner protocol.</td>
</tr>
<tr>
<td>Field</td>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>--------</td>
<td>-------</td>
<td>------------------------------</td>
</tr>
<tr>
<td></td>
<td>protocol</td>
<td>String</td>
<td>The scanner protocol.</td>
</tr>
</tbody>
</table>

**ShazzamScanner - result**

The scanner result.

<table>
<thead>
<tr>
<th>Field</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>result</td>
<td>String</td>
<td>The scanner result.</td>
</tr>
</tbody>
</table>

**ShazzamScanner - service**

The scanner service name.

<table>
<thead>
<tr>
<th>Field</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>service</td>
<td>String</td>
<td>The scanner service name.</td>
</tr>
</tbody>
</table>

**SimilaritySolution - Global**

Scriptable object used in Predictive Intelligence stores.

The `SimilaritySolution` API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the `sn_ml` namespace.

The solution setup-to-training flow is as follows:

1. **Build an encoder using the** `Encoder` **API.**
2. **Use the** `constructor` **to create a similarity solution object.**
3. **Add the solution object to the similarity solution store using the**
   `SimilaritySolutionStore - add()` **method.**
4. **Train the solution using the** `submitTrainingJob()` **method.** This creates a version of the object that you can manage using the `SimilaritySolutionVersion` API.
5. **Get predictions using the** `SimilaritySolutionVersion - predict()` **method.**

**Note:** This API runs with full privileges. To restrict user access, include an access control mechanism in the script.
For usage guidelines, refer to Using ML APIs.

**SimilaritySolution - SimilaritySolution(Object config)**

Creates a similarity solution.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>Object</td>
<td>JavaScript object containing configuration properties of the solution.</td>
</tr>
<tr>
<td>config.domainName</td>
<td>String</td>
<td>Optional. Domain name associated with this dataset. See Domain separation and Predictive Intelligence. Default: Current domain, for example, &quot;global&quot;.</td>
</tr>
<tr>
<td>config.encoder</td>
<td>Object</td>
<td>Trained encoder object to assign to this solution. See Encoder - Encoder(Object config).</td>
</tr>
<tr>
<td>config.label</td>
<td>String</td>
<td>Identifies the prediction task.</td>
</tr>
<tr>
<td>config.lookupDataset</td>
<td>Object</td>
<td>Name of the DatasetDefinition to use as the lookup set.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>config.minRowCount</td>
<td>String</td>
<td>Optional. Minimum number of records required in the dataset for training.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: 10000</td>
</tr>
<tr>
<td>config.processingLanguage</td>
<td>String</td>
<td>Optional. Processing language in two-letter ISO 639-1 language code format.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: &quot;en&quot;</td>
</tr>
<tr>
<td>config.stopwords</td>
<td>Array</td>
<td>Optional. Preset list of strings that the system automatically generates based on the language property setting. For details, see Create a custom stopwords list. Default: English Stopwords</td>
</tr>
<tr>
<td>config.testDataset</td>
<td>Object</td>
<td>Name of the DatasetDefinition to scan for similarities with lookupDataset results.</td>
</tr>
<tr>
<td>config.trainingFrequency</td>
<td>String</td>
<td>Optional. The frequency to retrain the model. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_30_days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_60_days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_90_days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_120_days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_180_days</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• run_once</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: run_once</td>
</tr>
</tbody>
</table>
| config.updateFrequency   |         | The frequency at which the model for the solution definition must be rebuilt.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• do_not_update</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_1_day</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_1_hour</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_6_hours</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_12_hours</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_1_minute</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_15_minutes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• every_30_minutes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: do_not_update</td>
</tr>
</tbody>
</table>

The following example shows how to create an object and add it to the SimilaritySolution store.

```javascript
var incidentData = new sn_ml.DatasetDefinition({
    'tableName': 'incident',
    'fieldNames': ['category', 'short_description']
});

var kbData = new sn_ml.DatasetDefinition({
    'tableName': 'kb_knowledge',
    'fieldNames': ['short_description'],
    'encodedQuery': 'active=true'
});

var encoder = sn_ml.EncoderStore.get('GloVe');

var mySolution = new sn_ml.SimilaritySolution({
    'label': "similarity solution",
    'lookupDataset': kbData,
    'testDataset': incidentData,
    'encoder': encoder
});

// add solution
var solutionName = sn_ml.SimilaritySolutionStore.add(mySolution);
```

**SimilaritySolution - cancelTrainingJob()**

Cancels a job for a solution object that has been submitted for training.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to cancel an existing training job.

```javascript
var mySolution = sn_ml.SimilaritySolutionStore.get('ml_sn_global_global_similarity');
mySolution.cancelTrainingJob();
```

**SimilaritySolution - getActiveVersion()**

Gets the active `SimilaritySolutionVersion` object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Active <code>SimilaritySolutionVersion</code> object.</td>
</tr>
</tbody>
</table>

The following example shows how to get an active `SimilaritySolution` version from the store and return its training status.

```javascript
var mlSolution = sn_ml.SimilaritySolutionStore.get('ml_x_snc_global_global_similarity');
gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getStatus()), null, 2));
```

Output:

```json
{
  "state": "solution_complete",
  "percentComplete": "100",
}
SimilaritySolution - getAllVersions()

Gets all versions of a SimilaritySolution object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Existing versions of a solution object. See also SimilaritySolutionVersion API.</td>
</tr>
</tbody>
</table>

The following example shows how to get all SimilaritySolution version objects and call the getVersionNumber() and getStatus() solution version methods on them.

```javascript
var mlSolution = sn_ml.SimilaritySolutionStore.get('ml_x_snc_global_global_Similarity');
var mlSolutionVersions = mlSolution.getAllVersions();
for (i = 0; i < mlSolutionVersions.length; i++) {
    gs.print("Version "+ mlSolutionVersions[i].getVersionNumber() + " Status: " +
               mlSolutionVersions[i].getStatus() +"\n");
}
```

Output:

Version 3 Status:
  ("state":"solution_complete","percentComplete":"100","hasJobEnded":"true")

Version 2 Status:
  ("state":"solution_complete","percentComplete":"100","hasJobEnded":"true")

Version 1 Status: ("state":"solution_cancelled","percentComplete":"0","hasJobEnded":"true")

SimilaritySolution - getLatestVersion()

Gets the latest version of a solution.
The following example shows how to get the latest version of a solution and return its training status.

```javascript
var mlSolution = sn_ml.SimilaritySolutionStore.get('ml_x_snc_global_global_Similarity');

gs.print(JSON.stringify(JSON.parse(mlSolution.getLatestVersion().getStatus()), null, 2));
```

Output:

```
{
    "state": "solution_complete",
    "percentComplete": "100",
    "hasJobEnded": "true"
}
```

**SimilaritySolution - getName()**

Gets the name of the object to use for interaction with the store.

The following example shows how to update `SimilaritySolution` dataset information and print the name of the object.
// Update solution
var myIncidentData = new sn_ml.DatasetDefinition({
  'tableName': 'incident',
  'fieldNames': ['category', 'short_description', 'priority'],
  'encodedQuery': 'activeANYTHING'
});

var eligibleFields = JSON.parse(myIncidentData.getEligibleFields('Similarity'));

var mySimilarity = new sn_ml.SimilaritySolution({
  'label': "my Similarity solution",
  'dataset': myIncidentData,
  'inputFieldNames': eligibleFields['eligibleInputFieldNames'],
  'predictedFieldName': 'category'
});

// update solution
sn_ml.SimilaritySolutionStore.update('ml_x_snc_global_global_my_solution_definition_4', mySimilarity);

// print solution name
gs.print('Solution Name: '+mySimilarity.getName());

Output:

Solution Name: ml_x_snc_global_global_my_solution_definition_4

SimilaritySolution - getProperties()

Gets solution object properties.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Contents of the Dataset and SimilaritySolution object details in SimilaritySolutionStore.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>`{</td>
<td></td>
</tr>
<tr>
<td>&quot;domainName&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;encoder&quot;: Object,</td>
<td></td>
</tr>
<tr>
<td>&quot;label&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;lookupDatasetProperties&quot;: Object,</td>
<td></td>
</tr>
<tr>
<td>&quot;name&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;processingLanguage&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;scope&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;stopwords&quot;: [Array],</td>
<td></td>
</tr>
<tr>
<td>&quot;testDatasetProperties&quot;: Object,</td>
<td></td>
</tr>
<tr>
<td>&quot;trainingFrequency&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;updateFrequency&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
</tbody>
</table>

**<Object>.domainName**

Domain name associated with this dataset. See [Domain separation and Predictive Intelligence](#).

Data type: String.

**<Object>.encoder**

Encoder object assigned to this solution. See [Encoder](#).

Data type: Object.

**<Object>.label**

Identifies the prediction task.
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.lookupDatasetProperties</td>
<td>Details of the object used as lookup set.</td>
</tr>
<tr>
<td>&lt;Object&gt;.lookupDatasetProperties.tableName</td>
<td>Name of the table for the dataset. For example, &quot;tableName&quot;: &quot;Incident&quot;.</td>
</tr>
<tr>
<td>&lt;Object&gt;.lookupDatasetProperties.fieldNames</td>
<td>List of field names from the specified table as strings. For example, &quot;fieldNames&quot;: [&quot;short_description&quot;, &quot;priority&quot;].</td>
</tr>
<tr>
<td>&lt;Object&gt;.lookupDatasetProperties.fieldNames.fieldDetails</td>
<td>List of JavaScript objects that specify field properties.</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.lookupDatasetProperties.fieldNames.fieldDetails.&lt;object&gt;.name</td>
<td>Name of the field defining the type of information to restrict this dataset to.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.lookupDatasetProperties.fieldDetails.&lt;object&gt;.type</td>
<td>Machine-learning field type.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.lookupDatasetProperties.fieldDetails.encodedQuery</td>
<td>Encoded query string in standard Glide format.</td>
</tr>
<tr>
<td></td>
<td>See <a href="https://glide.snowservice.com/help/encoded-query-strings">Encoded query strings</a>.</td>
</tr>
<tr>
<td>&lt;Object&gt;.name</td>
<td>System-assigned name</td>
</tr>
<tr>
<td>&lt;Object&gt;.processingLanguage</td>
<td>Processing language in two-letter ISO 639-1 language code.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.scope</td>
<td>Object scope.</td>
</tr>
<tr>
<td></td>
<td>Currently the only valid value is <code>global</code>.</td>
</tr>
<tr>
<td>&lt;Object&gt;.stopwords</td>
<td>Optional. Preset list of strings that the system automatically generates based on the <code>language</code> property.</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;.testDatasetProperties</td>
<td>Details of the DatasetDefinition() object used to search for similarities between results searched in this model and results found in lookupDataset. Data type: Object.</td>
</tr>
<tr>
<td>&lt;Object&gt;.testDatasetProperties.tableName</td>
<td>Name of the table for the dataset. For example, &quot;tableName&quot;: &quot;Incident&quot;. Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.testDatasetProperties.fieldNames</td>
<td>List of field names from the specified table as strings. For example, &quot;fieldNames&quot; : [&quot;short_description&quot;, &quot;priority&quot;]. Data type: Array.</td>
</tr>
<tr>
<td>&lt;Object&gt;.testDatasetProperties.fieldNames.fieldDetails</td>
<td>List of JavaScript objects that specify field properties.</td>
</tr>
</tbody>
</table>
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>```json</td>
<td>[</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td>Data type: Array.</td>
</tr>
<tr>
<td>&lt;Object&gt;.testDatasetProperties.fieldNames.fieldDetails.&lt;object&gt;.name</td>
<td>Name of the field defining the type of information to restrict this dataset to.</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.testDatasetProperties.fieldDetails.&lt;object&gt;.type</td>
<td>Machine-learning field type.</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.testDatasetProperties.fieldDetails.encodedQuery</td>
<td>Encoded query string in standard Glide format. See Encoded query strings.</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
<tr>
<td>&lt;Object&gt;.trainingFrequency</td>
<td>The frequency to retrain the model. Possible values: every_30_days, every_60_days, every_90_days, every_120_days, every_180_days, run_once</td>
</tr>
<tr>
<td></td>
<td>Default: run_once.</td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| `<Object>.updateFrequency` | The frequency at which the model for the solution definition must be rebuilt. Possible values:  
  - `do_not_update`
  - `every_1_day`
  - `every_1_hour`
  - `every_6_hours`
  - `every_12_hours`
  - `every_1_minute`
  - `every_15_minutes`
  - `every_30_minutes`

Default: `do_not_update`  
Datatype: String |

The following example gets properties of a solution object in the store.

```javascript
var mySolution =  
sn_ml.SimilaritySolutionStore.get('ml_sn_global_global_similarity_solution');

gs.print(JSON.stringify(JSON.parse(mySolution.getProperties()), null, 2));
```

Output:

```json
*** Script: {
  "domainName": "global",
  "encoderProperties": {
    "datasetsProperties": [],
    "name": "wordCorpusA"
  },
  "label": "similarity",
  "lookupDatasetProperties": {
    "tableName": "incident",
    "fieldNames": [  
```
SimilaritySolution - getVersion(String version)

Gets a solution by provided version number.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Existing version number of a solution.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Specified version of the SimilaritySolution() object on which you can call</td>
</tr>
<tr>
<td></td>
<td>SimilaritySolutionVersion API methods.</td>
</tr>
</tbody>
</table>

The following example shows how to get the training status of a solution by version number.

```javascript
var mlSolution = sn_ml.SimilaritySolutionStore.get('ml_x_snc_global_global_similarity');

gs.print(JSON.stringify(JSON.parse(mlSolution.getVersion('1').getStatus()), null, 2));
```

Output:
SimilaritySolution - setActiveVersion(String version)
Activates a specified version of a solution in the store.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Name of the SimilaritySolution() object version to activate. Activating this version deactivates any other version.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to activate a solution version in the store.

```
sn_ml.SimilaritySolution.setActiveVersion("ml_incident_categorization");
```

SimilaritySolution - submitTrainingJob()
Submits a training job.

ℹ️ Note: Before running this method, you must first add a solution to the store using the SimilaritySolutionStore - add() method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>SimilaritySolutionVersion object corresponding to the SimilaritySolution being trained.</td>
</tr>
</tbody>
</table>

The following example shows how to create a dataset, apply it to a solution, add the solution to a store, and submit the training job.

```javascript
// Create a dataset
var myData = new sn_ml.DatasetDefinition({
    'tableName': 'incident',
    'fieldNames': ['assignment_group', 'short_description', 'description'],
    'encodedQuery': 'activeANYTHING'
});

// Create a solution
var mySolution = new sn_ml.SimilaritySolution({
    'label': "my solution definition",
    'dataset': myData,
    'predictedFieldName': 'assignment_group',
    'inputFieldNames': ['short_description']
});

// Add the solution to the store to later be able to retrieve it.
var my_unique_name = sn_ml.SimilaritySolutionStore.add(mySolution);

// Train the solution - this is a long running job
var mySimilarityVersion = mySolution.submitTrainingJob();
```

**SimilaritySolutionStore - Global**

Enables storing and retrieving solutions.

The `SimilaritySolutionStore` API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the `sn_ml` namespace.

**SimilaritySolutionStore - add(Object mlSolution)**

Adds a new solution object to the store and returns a unique name.
Note: Label values do not need to be unique. For example, if you run this method with the same label 10 times, this method adds 10 different uniquely-named objects to the store.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mlSolution</td>
<td>SimilaritySolution</td>
<td>object to add to the store.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>System-generated solution name.</td>
</tr>
</tbody>
</table>

The following example shows how to add a solution to the store. Use to run the training job after adding it to the store.

```javascript
// Create a dataset
var myData = new sn_ml.DatasetDefinition({
  'tableName': 'incident',
  'fieldNames': ['assignment_group', 'short_description', 'description'],
  'encodedQuery': 'activeANYTHING'
});

// Create a solution
var mySolution = new sn_ml.SimilaritySolution({
  'label': "my solution definition",
  'dataset': myData,
  'predictedFieldName': 'assignment_group',
  'inputFieldNames': ['short_description']
});

// Add the solution to the store to later be able to retrieve it.
var my_unique_name = sn_ml.SimilaritySolutionStore.add(mySolution);
```

**SimilaritySolutionStore - deleteObject(String name)**

Removes a specified solution object from the store.
The following example shows how to delete a solution from the store.

```
sn_ml.SimilaritySolutionStore.deleteObject("ml_sn_global_global_solution");
```

**SimilaritySolutionStore - get(String name)**

Gets a solution object from the store.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of a solution in the store.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>SimilaritySolution object. Returns an error if the object does not exist.</td>
</tr>
</tbody>
</table>

The following example shows how to get a solution object from the store using the `get()` method and view its training status using the `SimilaritySolution - getActiveVersion()` and `SimilaritySolutionVersion - getStatus()` methods.

```
// Get status
var mlSolution = sn_ml.SimilaritySolutionStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getStatus(), null, 2)));
```

Output:

```
{
  "state":"solution_complete",
```
SimiliarSolutionStore - getAllNames(Object options)

Gets the names of all solution definition records in the store.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>Options for restricting results within the specified properties.</td>
</tr>
<tr>
<td>options.label</td>
<td>String</td>
<td>Optional. Label of your solution object.</td>
</tr>
<tr>
<td>options.domainName</td>
<td>String</td>
<td>Optional. Name of the domain for your solution object. Refer to Domain separation and Predictive Intelligence.</td>
</tr>
<tr>
<td>options.scope</td>
<td>String</td>
<td>Optional. Name of an application scope for your solution object.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>List of strings representing solution object names in the store.</td>
</tr>
</tbody>
</table>

In the following example, the `getAllNames()` method returns a list of all names in the store.

```javascript
gs.print(JSON.stringify(JSON.parse(sn_ml.SimilaritySolutionStore.getAllNames()), null, 2));
```

**Output:**

```
[
  "ml_x_snc_global_global_my_solution_definition_3",
  "ml_incident_assignment",
  "ml_x_snc_global_global_my_solution_definition",
]```
In the following example, the `getAllNames()` method returns only names associated with values set in the `options` parameter.

```javascript
var options = {
    'label' : 'my solution definition',
    'domainName' : 'global',
    'scope' : 'global'
};
var solNames = sn_ml.SimilaritySolutionStore.getAllNames(options);
gs.print(JSON.stringify(JSON.parse(solNames), null, 2));
```

**Output:**

```
[ "ml_x_snc_global_global_my_solution_definition"
]
```

---

**SimilaritySolutionStore - update(String name, Object mlSolution)**

Replaces an existing object in the store with the object passed as a parameter. The object name provided must be empty or match.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the solution to update.</td>
</tr>
<tr>
<td>mlSolution</td>
<td>SimilaritySolution()</td>
<td>object properties to update.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to update a solution object in the store.

```javascript
var solutionUpdate = new sn_ml.SimilaritySolution({
    'label': 'my solution definition',
    'dataset' : myData,
    'predictedFieldName' : 'assignment_group',
});
```
**SimilaritySolutionVersion - Global**

Scriptable object used in Predictive Intelligence stores.

The **SimilaritySolutionVersion** API requires the Predictive Intelligence plugin (com.glide.platform_ml) and is provided within the **sn_ml** namespace.

This API is used for working with solution versions based on **SimilaritySolution** API objects in the **SimilaritySolution store**.

The system creates a solution version each time you train a solution definition. Most versions are created during scheduled solution training.

Methods in this API are accessible using the following **SimilaritySolution** methods:

- **getActiveVersion()**
- **getAllVersions()**
- **getLatestVersion()**
- **getVersion()**

**SimilaritySolutionVersion - getProperties()**

Gets solution object properties and version number.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Object</td>
</tr>
</tbody>
</table>

```javascript
sn_ml.SimilaritySolutionStore.update('ml_sn_global_global_incident_service', solutionUpdate);
```
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|      | "isActive": "String",
|      | "label": "String",
|      | "lookupDatasetProperties": {Object},
|      | "name": "String",
|      | "predictedFieldName": "String",
|      | "processingLanguage": "String",
|      | "scope": "String",
|      | "stopwords": [Array],
|      | "testDatasetProperties": {Object},
|      | "trainingFrequency": "String",
|      | "updateFrequency": "String",
|      | "versionNumber": "String"

 responseObject.domainName

- Domain name associated with this dataset. See [Domain separation and Predictive Intelligence](#).
- Data type: String.

responseObject.encoder

- Encoder object assigned to this solution. See [Encoder - Encoder(Object config)](#).
- Data type: Object.

responseObject.isActive

- Flag that indicates whether this version is active. Valid values:
  - true: Version is active.
  - false: Version is not active.
- Data type: String

responseObject.label

- Identifies the prediction task.

```json
{
  "label": "my first prediction"
}
```

- Data type: String.

responseObject.lookupDatasetProperties

- Details of the [DatasetDefinition()](#) object used as the lookup set.

```json
{
  "encodedQuery": "String",
```
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
|      | "fieldDetails": [Array],  
|      | "fieldNames": [Array],  
|      | "tableName": "String"
|      | }  

Data type: Object.

<Object>.lookupDatasetProperties.tableName  
Name of the table for the dataset. For example,  
"tableName" : "Incident".
Data type: String.

<Object>.lookupDatasetProperties.fieldNames  
List of field names from the specified table as strings.  
For example, "fieldNames" : ["short_description",  
"priority"].
Data type: Array.

<Object>.lookupDatasetProperties.fieldDetails  
List of field properties.

[  
|  
|  "name": "String",  
|  "type": "String"  
|}

Data type: Array.

<Object>.lookupDatasetProperties.fieldDetails.<object>.name  
Name of the field defining the type of information to  
restrict this dataset to.
Data type: String.

<Object>.lookupDatasetProperties.fieldDetails.<object>.type  
Machine-learning field type.
Data type: String.

<Object>.lookupDatasetProperties.encodedQuery  
Encoded query string in standard Glide format. See  
Encoded query strings.
Data type: String.

<Object>.name  
System-assigned name.
Data type: String.

<Object>.predictedFieldName  
Identifies a field to be trained for predictability.
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String.</td>
<td></td>
</tr>
<tr>
<td><code>&lt;Object&gt;.processingLanguage</code></td>
<td>Processing language in two-letter ISO 639-1 language code format. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.scope</code></td>
<td>Object scope. Currently the only valid value is <code>global</code>. Data type: String.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.stopwords</code></td>
<td>Optional. Preset list of strings that the system automatically generates based on the <code>language</code> property setting. For details, see Create a custom stopwords list. Data type: Array.</td>
</tr>
<tr>
<td><code>&lt;Object&gt;.testDatasetProperties</code></td>
<td>Described the DatasetDefinition() object used to retrieve similarities between results searched in this model and results found in the lookupDataset.</td>
</tr>
<tr>
<td></td>
<td>`{</td>
</tr>
<tr>
<td></td>
<td>&quot;encodedQuery&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;fieldDetails&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;fieldName&quot;: [Array],</td>
</tr>
</tbody>
</table>
| | "tableName": "String"
| } |
| Data type: Object. | |
| `<Object>.testDatasetProperties.tableName` | Name of the table for the dataset. For example, "tableName" : "Incident". Data type: String. |
| `<Object>.testDatasetProperties.fieldNames` | List of field names from the specified table as strings. For example, "fieldName" : ["short_description", "priority"]. Data type: Array. |
| `<Object>.testDatasetProperties.fieldDetails` | List of objects that specify field properties. |
| | `[ |
| | { |
| | "name": "String", |
| | "type": "String"
| } |

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Data type: Array.</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;Object&gt;.testDatasetProperties.fieldDetails&lt;object&gt;.name</code></td>
<td>Name of the field defining the type of information to restrict this dataset to.</td>
</tr>
<tr>
<td><code>Data type: String.</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;Object&gt;.testDatasetProperties.fieldDetails&lt;object&gt;.type</code></td>
<td>Machine-learning field type.</td>
</tr>
<tr>
<td><code>Data type: String.</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;Object&gt;.testDatasetProperties.encodedQuery</code></td>
<td>Encoded query string in standard Glide format. See Encoded query strings.</td>
</tr>
<tr>
<td><code>Data type: String.</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;Object&gt;.trainingFrequency</code></td>
<td>Frequency to retrain the model. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• every_30_days</td>
</tr>
<tr>
<td></td>
<td>• every_60_days</td>
</tr>
<tr>
<td></td>
<td>• every_90_days</td>
</tr>
<tr>
<td></td>
<td>• every_120_days</td>
</tr>
<tr>
<td></td>
<td>• every_180_days</td>
</tr>
<tr>
<td></td>
<td>• run_once</td>
</tr>
<tr>
<td></td>
<td>Default: run_once</td>
</tr>
<tr>
<td><code>Data type: String.</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;Object&gt;.updateFrequency</code></td>
<td>Frequency at which the model for the solution definition must be rebuilt. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• do_not_update</td>
</tr>
<tr>
<td></td>
<td>• every_1_day</td>
</tr>
<tr>
<td></td>
<td>• every_1_hour</td>
</tr>
<tr>
<td></td>
<td>• every_6_hours</td>
</tr>
<tr>
<td></td>
<td>• every_12_hours</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• every_1_minute</td>
</tr>
<tr>
<td></td>
<td>• every_15_minutes</td>
</tr>
<tr>
<td></td>
<td>• every_30_minutes</td>
</tr>
<tr>
<td>Default: do_not_update</td>
<td>Datatype: String</td>
</tr>
</tbody>
</table>

<Object>.versionNumber

Version number of the SimilaritySolution object. Data type: String.

The following example gets properties of the active object version in the store.

```javascript
// Get properties
var mlSolution = sn_ml.SimilaritySolutionStore.get('ml_incident_categorization');

gs.print(JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getProperties()), null, 2));
```

Output:

```json
*** Script: {
    "domainName": "global",
    "encoderProperties": {
        "datasetsProperties": [],
        "name": "wordCorpusA"
    },
    "isActive": "true",
    "label": "similarity",
    "lookupDatasetProperties": {
        "encodedQuery": "",
        "fieldNames": [
            "short_description"
        ],
        "tableName": "incident"
    },
    "name": "ml_x_snc_global_global_similarity",
    "processingLanguage": "en",
    "stopwords": [
        "Default English Stopwords"
    ],
```
SimilaritySolutionVersion - getStatus(Boolean includeDetails)

Gets training completion status.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>includeDetails</td>
<td>Boolean</td>
<td>Flag that indicates whether to return status details. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Return additional details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Do not return additional details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: False</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>JavaScript object containing training status information for a SimilaritySolution object.</td>
</tr>
<tr>
<td>&lt;Object&gt;.state</td>
<td>Training completion state. If the training job reaches a terminal state, the job does not leave that state. If the state is terminal, the hasJobEnded property is set to true. Possible values:</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• fetching_files_for_training</td>
</tr>
<tr>
<td></td>
<td>• preparing_data</td>
</tr>
<tr>
<td></td>
<td>• retry</td>
</tr>
<tr>
<td></td>
<td>• solution_cancelled (terminal)</td>
</tr>
<tr>
<td></td>
<td>• solution_complete (terminal)</td>
</tr>
<tr>
<td></td>
<td>• solution_error (terminal)</td>
</tr>
<tr>
<td></td>
<td>• solutionIncomplete</td>
</tr>
<tr>
<td></td>
<td>• training_request_received</td>
</tr>
<tr>
<td></td>
<td>• training_request_timed_out (terminal)</td>
</tr>
<tr>
<td></td>
<td>• training_solution</td>
</tr>
<tr>
<td></td>
<td>• uploading_solution</td>
</tr>
<tr>
<td></td>
<td>• waiting_for_training</td>
</tr>
</tbody>
</table>

Data type: String

```
<Object>.hasJobEnded
```

Field that indicates whether training is complete.

Valid values:
- true: Training is complete.
- false: Training is incomplete.

Data type: Boolean value as a String

```
<Object>.percentComplete
```

Number between zero and 100 representing training percent complete. If the completion percentage is less than 100, the job might be in a terminal state. For example, if training times out.

Data type: Number as a String

```
<Object>.details
```

Object containing a list of additional training details.

Data type: Object

The following example shows a successful result with training complete.

```javascript
// Get status
var mlSolution = sn_ml.SimilaritySolutionStore.get('ml_incident_categorization');
```
Output:

```javascript
{
  "state": "solution_complete",
  "percentComplete": "100",
  "hasJobEnded": "true",
  "details": {"stepLabel": "Solution Complete"} // This information is only returned if getActiveVersion().getStatus(true);
}
```

The following example shows an unsuccessful result with training complete.

```javascript
// Get status
var solutionName = 'ml_x_snc_global_global_similarity_solution';
var mlSolution = sn_ml.SimilaritySolutionStore.get(solutionName);
var trainingStatus = JSON.parse(mlSolution.getLatestVersion().getStatus());

gs.print(JSON.stringify(trainingStatus, null, 2));
```

Output:

```javascript
{
  "state": "solution_error",
  "percentComplete": "100",
  "hasJobEnded": "true"
}
```

**SimilaritySolutionVersion - getVersionNumber()**

Gets the version number of a solution object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Version number.</td>
</tr>
</tbody>
</table>
The following example shows how to get a version number.

```javascript
// Get version number
var mlSolution = sn_ml.SimilaritySolutionStore.get('ml_incident_categorization');

gs.print("Version number:
  "+JSON.stringify(JSON.parse(mlSolution.getActiveVersion().getVersionNumber()), null, 2));
```

Output:

Version number: 1

**SimilaritySolutionVersion** - predict(Object input, Object options)

Gets the input data for a prediction.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>Object</td>
<td>GlideRecord or array of JSON objects containing field names and values as key-value pairs.</td>
</tr>
<tr>
<td>options</td>
<td>Object</td>
<td>Optional values for filtering prediction results.</td>
</tr>
<tr>
<td>options.apply_threshold</td>
<td>Boolean</td>
<td>Flag that indicates whether to check the threshold value for the solution and apply it to the result set. Valid values:</td>
</tr>
<tr>
<td>options.top_n</td>
<td>Number</td>
<td>If provided, returns the top results, up to the specified number of predictions.</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Object</td>
<td>JSON object containing the prediction results sorted by sys_id or record_number.</td>
<td><code>{&lt;identifier&gt;: [Array]}</code></td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;</td>
<td>List of objects with details for each prediction result.</td>
<td>&lt;identifier&gt;: [</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;.&lt;object&gt;.confidence</td>
<td>Value of the confidence associated with the prediction. For example, 53.84.</td>
<td>Data type: Number</td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;.&lt;object&gt;.predictedSysId</td>
<td>The sys_id of the predicted value. Results can be from any table on which information is being predicted.</td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;.&lt;object&gt;.predictedValue</td>
<td>Value representing the prediction result.</td>
<td>Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.&lt;identifier&gt;.&lt;object&gt;.threshold</td>
<td>Value of the configured threshold associated with the prediction.</td>
<td>Data type: Number</td>
</tr>
</tbody>
</table>

The following example shows how to display prediction results for a `predict()` method that takes a GlideRecord by sys_id for input and includes optional parameters to restrict to top three results and exclude the threshold value.

```javascript
var mlSolution = sn_ml.SimilaritySolutionStore.get('ml_incident_categorization');

// single GlideRecord input
```
var input = new GlideRecord("incident");
input.get("<sys_id>");

// configure optional parameters
var options = {};
options.top_n = 3;
options.apply_threshold = false;

var results = mlSolution.getVersion(1).predict(input, options);
// pretty print JSON results
gs.print(JSON.stringify(JSON.parse(results), null, 2));

{  
    "<sys_id/now_GR>": [
        {
            "confidence": 62.10782320780268,
            "threshold": 20.36,
            "predictedValue": "Clone Issues",
            "predictedSysId": ""
        },
        {
            "confidence": 6.945237375770391,
            "threshold": 16.63,
            "predictedValue": "Instance Administration",
            "predictedSysId": ""
        },
        {
            "confidence": 5.321061076300759,
            "threshold": 23.7,
            "predictedValue": "Administration",
            "predictedSysId": ""
        }
    ]
}

The following example shows how to display prediction results for a predict() method that takes an array of field names as key-value pairs for input and includes optional parameters to restrict to top three results and exclude the threshold value.

var mlSolution = sn_ml.SimilaritySolutionStore.get("ml_incident_categorization");
// key-value pairs input
var input = [{"short_description":"my email is not working"}, {short_description:"need help with password"}];
// configure optional parameters
var options = {};
options.top_n = 3;
options.apply_threshold = false;
var results = mlSolution.predict(input, options);
// pretty print JSON results
gs.print(JSON.stringify(JSON.parse(results), null, 2));

{
  "1": [
    {
      "confidence": 37.5023032262591,
      "threshold": 10.72,
      "predictedValue": "Authentication",
      "predictedSysId": ""
    },
  ],
  "2": [
    {
      "confidence": 99,
      "threshold": 17.77,
      "predictedValue": "Email",
      "predictedSysId": ""
    },
    {
      "confidence": 2.8773826570713514,
      "threshold": 10.72,
      "predictedValue": "Authentication",
      "predictedSysId": ""
    },
    {
      "confidence": 2.8773826570713514,
      "threshold": 10.72,
      "predictedValue": "Authentication",
      "predictedSysId": ""
    }
  ]
}
SkillDeterminationUtils - Scoped

Provides utility methods to get or set work item skill data.

The SkillDeterminationUtils API requires the Skill Determination (com.snc.skill_determination) plugin and is provided within the sn_skill_rule namespace.

SkillDeterminationUtils - assignSkillsToWorkItem(Array skills, GlideRecord now_GR)

Assigns an array of active skill objects to a work item.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>skills</td>
<td>Array</td>
<td>Array of active skill objects to be assigned.</td>
</tr>
<tr>
<td>skill.skillName</td>
<td>String</td>
<td>Name of the skill.</td>
</tr>
<tr>
<td>skill.skillSysId</td>
<td>String</td>
<td>Sys ID of the skill.</td>
</tr>
<tr>
<td>skill.mandatory</td>
<td>Boolean</td>
<td>True if the skill is mandatory, false otherwise.</td>
</tr>
<tr>
<td>skill.skillLevelName</td>
<td>String</td>
<td>Optional skill level name.</td>
</tr>
<tr>
<td>skill.skillLevelSysId</td>
<td>String</td>
<td>Optional skill level Sys ID.</td>
</tr>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>GlideRecord of the work item on which to assign skills.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
The examples below create an Interaction record using the table name (interaction) and show how to assign skills to the interaction work item, creating a record in the Interaction Skills [interaction_m2m_skill] table.

Use case 1: Use returned results from determineWorkItemSkills() to assign skills to a work item from the Interaction table.

```javascript
// Create an interaction record
var now_GR = new GlideRecord('interaction');
now_GR.insert();

var util = new sn_skill_rule.SkillDeterminationUtils();

// Get interaction skills interaction
// and specify if any are mandatory
var result = util.determineWorkItemSkills(now_GR);

// Assign skills determined by result and
// assign them to the interaction work item
util.assignSkillsToWorkItem(result, now_GR);
```

Use case 2: Manually insert skills array. In this case, assign IT skills to the Interaction work item.

```javascript
var now_GR = new GlideRecord('interaction');
now_GR.insert();

var util = new sn_skill_rule.SkillDeterminationUtils();

var skills = [
    {"skillSysId":"2eb1c2029f100200a3bc1471367fcfe4", "skillName":"IT",
    "mandatory":true, "skillLevelName":"", "skillLevelSysId":null}
];

util.assignSkillsToWorkItem(skills, now_GR);
```

**SkillDeterminationUtils - determineWorkItemSkills(GlideRecord now_GR)**

Gets skills for a specified work item, indicates if the skills are mandatory, and lists any skill levels.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>GlideRecord of a work item from any interaction or task table extension.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>One or more skill objects.</td>
</tr>
<tr>
<td>• <code>skillSysId</code>: String. Sys ID of the skill from the Skills [cmn_skill] table.</td>
<td></td>
</tr>
<tr>
<td>• <code>skillName</code>: String. Name of the skill.</td>
<td></td>
</tr>
<tr>
<td>• <code>mandatory</code>: Boolean. True if mandatory, false otherwise.</td>
<td></td>
</tr>
<tr>
<td>• <code>skillLevelName</code>: If skill exists, name of the skill level.</td>
<td></td>
</tr>
<tr>
<td>• <code>skillLevelSysId</code>: If skill exists, Sys ID of the skill level from the Skill Levels [cmn_skill_level] table.</td>
<td></td>
</tr>
</tbody>
</table>

The following script creates an interaction record, gets skills for the interaction work item, and indicates if the skills are mandatory.

```
// Create an interaction record
var now_GR = new GlideRecord('interaction');
now_GR.insert();

var util = new sn_skill_rule.SkillDeterminationUtils();

var result = util.determineWorkItemSkills(now_GR);
gs.info(JSON.stringify(result));
```

Output:

```
[{
"skillSysId":"6c0f025c7f672300a8b1bdc8adfa917f",
"skillName":"Premier Support Certified",
"mandatory":false,"skillLevelName":"
"skillLevelSysId":null}
```

**SkillDeterminationUtils - SkillDeterminationUtils()**

Constructor to create an instance of SkillDeterminationUtils.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
var util = new sn_skill_rule.SkillDeterminationUtils();
```
SLARepair - Global

The SLARepair API first deletes the existing SLAs and then recreates them from each task’s history.

SLARepair - repairByFilter(String filter, String sourceTable)

Repair the task SLAs associated with the passed-in filter and source table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>string</td>
<td>Specify an encoded query that is used to retrieve a set of records from the source table.</td>
</tr>
<tr>
<td>sourceTable</td>
<td>string</td>
<td>Specify the name of a table that is (or extends) contract_sla, task_sla, or task.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Repair SLAs for problems created last month with a priority of 2.

```javascript
var now_GR = new GlideRecord("problem");
now_GR.addQuery("sys_created_on", "ON", "Last Month@javascript:gs.beginningOfLastMonth()@javascript:gs.endOfLastMonth()";)
now_GR.addQuery("priority", "2");
now_GR.query();

var repair = new SLARepair();
while (now_GR.next())
  repair.repairByFilter(now_GR.getEncodedQuery(), now_GR.getRecordClassName());
```

SLARepair - repairByGlideRecord(GlideRecord now_GR)

Repair the task SLAs associated with the passed in GlideRecord.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>now_GR</td>
<td>GlideRecord</td>
<td>Specify a GlideRecord for a table that is (or extends) contract_sla, task_sla, or task.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Repair SLAs for problems created last month with a priority of 2.

```javascript
var now_GR = new GlideRecord("problem");
now_GR.addQuery("sys_created_on", "ON", "Last Month@javascript:gs.beginningOfLastMonth()@javascript:gs.endOfLastMonth()";)
now_GR.addQuery("priority", "2");
now_GR.query();

var repair = new SLARepair();
while (now_GR.next())
   repair.repairByGlideRecord(now_GR);
```

**SLARepair - repairBySysId(String sysId, String sourceTable)**

Repair the task SLAs associated with the passed in sys_id and source table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysId</td>
<td>string</td>
<td>Specify the ID of a table that is (or extends) contract_sla, task_sla, or task.</td>
</tr>
<tr>
<td>sourceTable</td>
<td>string</td>
<td>Specify the name of a table that is (or extends) contract_sla, task_sla, or task.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
Find problems created last month with a priority of 2

```javascript
var now_GR = new GlideRecord("problem");
now_GR.addQuery("sys_created_on", "ON", "Last Month@javascript:gs.beginningOfLastMonth()@javascript:gs.endOfLastMonth()" paranoia="true");
now_GR.addQuery("priority", "2");
now_GR.query();
```

```javascript
var repair = new SLARepair();
while (now_GR.next())
  repair.repairBySysId(now_GR.sys_id + ",", now_GR.getRecordClassName());
```

**SLARepair - setAuditEnabled(Boolean onOrOff)**

Enables or disables auditing when running a repair.

By default, auditing is set to the value in the property `com.snc.sla.repair.audit`. You can override this with passing in true to enable or false to disable auditing.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOrOff</td>
<td>Boolean</td>
<td>Determines whether to enable or disable auditing.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>this</td>
<td>A self-reference to allow for method chaining.</td>
</tr>
</tbody>
</table>

```javascript
var builder = new SLARepair();
builder.setAuditEnabled(true);
```

**SLARepair - setRunWorkflow(Boolean onOrOff)**

Enables or disables running a workflow for each of the Task SLA records being repaired.

By default, when a Task SLA is repaired the workflow will be run during the repair process. To override this, you can pass in false to disable running of the workflow or true to enable it.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOrOff</td>
<td>Boolean</td>
<td>Determines whether to enable or disable workflow.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>this</td>
<td>A self-reference to allow for method chaining.</td>
</tr>
</tbody>
</table>

```javascript
var repair = new SLARepair();
repair.setRunWorkflow(false);
```

### SLARepair - setValidateOnly(Boolean onOrOff)

Validates the repair request.

If false is passed in, the task SLAs will be repaired. If true is passed in, calls to repair will not alter any task SLAs but only validate the supplied parameters and generate a count of records to be repaired.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onOrOff</td>
<td>Boolean</td>
<td>Determines whether to enable or disable validation.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>this</td>
<td>A self-reference to allow for method chaining.</td>
</tr>
</tbody>
</table>

```javascript
var repair = new SLARepair();
repair.setValidateOnly(true);
```

### SLPServiceRegistryQuery - Global

Maps Shazzam! SLP query results to port probes.

Use this API during the discovery scanning phase.
SLPServiceRegistryQuery - getShazzamQuerierClassname()

Returns the Shazzam querier class name.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The class name</td>
</tr>
</tbody>
</table>

SLPServiceRegistryQuery - toPortProbeMap()

Creates a scalar map for port probes to use in XML output to Shazzam!.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The scalar map</td>
</tr>
</tbody>
</table>

SMSPreferenceHandler - Scoped, Global

The SMSPreferenceHandler API provides methods to manage SMS configurations. This API is implemented as the script include SMSPreferenceHandlerSNC and resides in the `sn_sms_pref` namespace. To use this API you must install the Notify plugin (com.snc.notify) which requires a separate subscription. You can activate this plugin if you have the admin role.

SMSPreferenceHandler - getConfig()

Returns the preferences configuration record for the current telephony service provider.
The method queries all preference configuration records from the SMS Preference Configuration [sn_sms_pref_configuration] table for the current telephony service provider, sorts the results by the Order field (lowest to highest), and then returns the first record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>Preference configuration record for the current telephony service provider. If no record exists, then null.</td>
</tr>
</tbody>
</table>

**Example**

```javascript
var handler = new sn_sms_pref.SMSPreferenceHandler('TwilioDirect'); // Pass the provider name
var config = handler.getConfig();
```

**SMSPreferenceHandler - SMSPreferenceHandler(String provider)**

Instantiates the SMSPreferenceHandler script include object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>provider</td>
<td>String</td>
<td>Name of the telephony service provider.</td>
</tr>
</tbody>
</table>

```javascript
var handler = new sn_sms_pref.SMSPreferenceHandler('TwilioDirect'); // Pass the provider name
var config = handler.getConfig();
```

**SNAnalytics - Client**

The SNAnalytics API provides methods to push custom analytics data (events, pages, and user properties) to the User Experience Analytics for Service Portal dashboard.
User Experience Analytics for Service Portal provides dashboard views for monitoring the key performance indicators (KPIs) of web applications built on Service Portal. You can use these insights to optimize your portal. For example, User Experience Analytics tracks when a user orders a catalog item or views a knowledge article. You can use this data to infer which items or articles are the most popular among users.

To access this API, the Service Portal Analytics (com.glide.service-portal.analytics) plugin must be activated on the instance. In addition, within your application, you must import the snAnalytics Angular service, such as:

```html
<client_script><![CDATA[function($rootScope, $scope, $window, $timeout, spUtil, $sce, spModal, $uibModal, $location, cabrillo, snAnalytics)
</client_script>]

For additional information, see User Experience Analytics for Service Portal.

**SNAnalytics - addEvent(Object payload)**

Stores the specified event information in the analytics data store.

Events are actions performed by a user, such as clicking a button or submitting a form. Call this method within your web-page widget whenever you want to capture a user action. These events then automatically appear on the associated user session timeline and User Experience Analytics dashboard.

The following is an example of a payload passed in an `addEvent()` call:

```javascript
var payload = {}; 
payload.name = "Manage Account";
payload.data = {}; 
payload.data["Function Name"] = c.data.function_name; 
payload.data["User Type"] = c.data.user_type; 
payload.data["Company"] = c.data.company_name; 
snAnalytics.addEvent(payload);
```

The screen capture below shows the information that appears on the analytics dashboard for the event "Manage Account." The timeline at the top represents the number of times that the event occurred. The pie charts below the
timeline reflect the properties that were captured in the `addEvent()` call.

If you want to capture when users access a web page, use the `SNAnalytics-startPage(String name, String description)` method instead of this method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>payload</td>
<td>Object</td>
<td>Event to store in the analytics data store. Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;payload&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;data&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;name&quot;: String</td>
</tr>
<tr>
<td>payload.data</td>
<td>Array</td>
<td>Optional. Name-value pairs of custom event properties. These properties can be any values that you want to track and see on the analytics</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dashboard. They appear under the associated event timeline on the analytics dashboard. The <strong>Page Id</strong> property always appears first on</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the dashboard for all base system events, and all other properties are sorted alphabetically. If no properties are required for an event,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>only an event timeline appears on the analytics dashboard. Properties can be added at a later time. Default value: Null</td>
</tr>
<tr>
<td>payload.name</td>
<td>String</td>
<td>Descriptive name of the event. Special characters are not allowed.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum length:</td>
<td></td>
<td>The length of the event name and value cannot exceed 300 bytes.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to call the addEvent() method during initialization of a widget.

```javascript
function initialize() {
  c.options.glyph = c.options.glyph || 'search';
  c.options.title = c.options.title || c.data.searchMsg;
  c.options.color = c.options.color || "default";
  c.searchTerm = c.data.q;
  c.searchQuery = "";
  c.pageID = $scope.page && $scope.page.id;
  c.showSuggestions = c.data.searchTypeBehavior === "suggestions" &&
                      c.data.isSuggestionsEnabled === "true";
  c.suggestionsLimit = c.options.limit || "";
  c.latitude = null;
  c.longitude = null;
  c.isLocationTrackerDisabled = c.data.isLocationTrackerDisabled === "true";
  c.isTypeAheadEnabled = c.data.isTypeAheadEnabled === "true";

  c.sendAnalytics = function(type){
    var payload = {};
    payload.name = "Initiate Search";
    payload.data = {};
    payload.data["Keyword"] = (type == 'User Entered' ? c.searchTerm : c.searchQuery);
    payload.data["Type"] = type;
    snAnalytics.addEvent(payload);
  }
}
```

**SNAnalytics - appendToUserProperty(String name, String value)**

Appends the specified string to the specified user string property in the analytics data store.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| name   | String or String[][] | Name of the property to append the specified string to. Special characters are not allowed.  

**Note:** The associated property must be a string or string[].

Maximum length: The length of the property name and property value cannot exceed 300 bytes.

| value  | String       | Value to append to the string property.                                    |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to add television to the `tags` property.

```javascript
snAnalytics.setUserProperties({
    level: 7,
    lastPurchase: new Date(),
    lastPurchaseId: '41563cd2-1666-4855-8c0d-b9ca778aed23',
    isPremium: true,
    tags: ['chair', 'table'],
});

// Append television to the tags property (now 'tags' will have 'chair', 'table', and 'television')
snAnalytics.appendToUserProperty('tags', 'television');
```

**SNAnalytics - incUserProperty(String name, Number value)**

Increments or decrements the specified user property value with the specified number value in the analytics data store.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the property to increment. Value is case-sensitive.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Number</td>
<td>Amount to increment the property by. If you enter a negative number, the value is decremented.</td>
</tr>
</tbody>
</table>

**Note:** The associated property must be a number.

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to increment the property `Grace days` by 5.

```javascript
snAnalytics.incUserProperty('Grace days', 5)
```

**SNAntics - removeUserProperty(String name)**

Removes the specified property for the current user from the analytics data store.

In addition, the property no longer appears on the analytics dashboard.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the property to remove. Value is case-sensitive.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to remove the `IsAdmin` property.

```javascript
snAnalytics.removeUserProperty('IsAdmin');
```
SNAalytics - setUserProperties(Object properties)

Sets the specified properties with the specified values for the current user in the analytics data store.

These properties are saved in the analytics data store and appear on the user session details page as illustrated below. If a property already exists in the analytics data store, the current value is overwritten with the new value.

Parameter Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>properties</td>
<td>Object</td>
<td>Object that contains the name-value pairs of the user properties to set, such as:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{ level: 7, lastPurchase: new Date(), lastPurchaseId: '41563cd2-1666-4855-8c0d-b9ca778aed23', isPremium: true, tags: ['chair', 'table'], }</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>string[], or null</td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set multiple properties for the current user.

```javascript
snAnalytics.setUserProperties({
    level: 7,
    lastPurchase: new Date(),
    lastPurchaseId: '41563cd2-1666-4855-8c0d-b9ca778aed23',
    isPremium: true,
    tags: ['chair', 'table'],
});
```

**SNAnalytics - setUserProperty(String name, UserProperty value)**

Sets the specified property with the specified value for the current user in the analytics data store.

These properties are saved in the analytics data store and appear on the user session details page as illustrated below. If a property already exists in the analytics data store, the current value is overwritten with the new value.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the property to update. This name appears as the label for the property. For example, in the prior screenshot, Domain, Instance Name, Company, Role, and User Type are all name parameters. Special characters are not allowed. Maximum length: The length of the property name and property value cannot exceed 300 bytes.</td>
</tr>
<tr>
<td>value</td>
<td>UserProperty</td>
<td>Value to set in the specified property. This value can be a string, boolean, number, date, string[], or null.</td>
</tr>
</tbody>
</table>
The following example shows how to set the property `Company`.

```java
snAnalytics.setUserProperty('Company', "ABC Company")
```

### SNAnalytics - startPage(String name, String description)

Saves the name and description of a page in the analytics data store.

This information appears in the user session timeline and on the analytics dashboard. Call this method within your custom widgets to track the pages visited by a user. You can also use this method to track user navigation within an individual page. For more information, see [Use User Experience Analytics](#).

> **Note:** In general, portal pages are automatically tagged with this tracking capability. Use this method for custom scenarios, such as a single page custom widget in a wizard scenario.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Descriptive name of the page or page section. Special characters are not allowed.</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>Optional. Description of the page to appear in the timeline and analytics dashboard. Default: name parameter value</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to call the `startPage()` method.

```javascript
snAnalytics.startPage('login_view', 'Login');
```

**SNEventSenderProvider - Global**

The `SNEventSenderProvider` API provides the method to instantiate an `IEventSender` instance.

Use this API when writing scripts that will run on MID Servers to instantiate the object to send an event to your ServiceNow instance.

Use the `IEventSender` API to send the event to the ServiceNow instance. Use the `Event` API to add/update fields within an event.

You must activate the Event Management (com.glideapp.itom.snac) plugin before attempting to access this API. The Event Management plugin requires a separate subscription and must be activated by ServiceNow personnel. This plugin includes demo data and activates related plugins if they are not already active.

For additional information on event management, see Event Management.

**EventSender - getEventSender()**

Returns an `IEventSender` object to use to send events from a MID Server to a ServiceNow instance.

You must call this method before calling any `IEventSender` API methods.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEventSender</td>
<td>IEventSender object to use to send an event to a ServiceNow instance.</td>
</tr>
</tbody>
</table>

```javascript
var eventSender = SNEventSenderProvider.getEventSender();
```

### SNMPGatherDocParts - Global

Handles gathering the parts of an SNMP response document for JavaScript SNMP sensors.

Use with a discovery script when you need to gather the parts of an SNMP response document.

#### SNMPGatherDocParts - gather(String doc, Object probe)

Returns the given XML document with any data gathered by prior SNMP probes.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc</td>
<td>String</td>
<td>The XML document for this sensor</td>
</tr>
<tr>
<td>probe</td>
<td>Object</td>
<td>The probe instance</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The XML document with the gathered data.</td>
</tr>
</tbody>
</table>

### SNMPNetworkInterfaces - Global

Creates network interfaces for JavaScript SNMP sensors.

Use with a discovery script when you need to create network interfaces for JavaScript SNMP sensors.
SNMPNetworkInterfaces process(String cmdb_ci, String doc, Boolean isComputer, String agentName, String ecc_sys_id)

Analyzes the SNMP data and reconciles the discovered NICs to the CMDB’s NICs.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmdb_ci</td>
<td>String</td>
<td>The sys_id of the CI</td>
</tr>
<tr>
<td>doc</td>
<td>String</td>
<td>The SNMP data in XML form</td>
</tr>
<tr>
<td>isComputer</td>
<td>Boolean</td>
<td>If true, this is a computer; otherwise, false.</td>
</tr>
<tr>
<td>agentName</td>
<td>String</td>
<td>Name of the MID server</td>
</tr>
<tr>
<td>ecc_sys_id</td>
<td>String</td>
<td>The sys_id of the ECC queue record with the input.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

SNMPResponse - Global

Wraps an SNMP payload response instance with methods to safely and easily retrieve SNMP singleton fields or tables.

Use to retrieve SNMP singleton fields or tables.

SNMPResponse - buildIndex(Object cur, String name, String extName)

Builds an index to all SNMP entities in the current object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cur</td>
<td>Object</td>
<td>The current object</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The index name</td>
</tr>
<tr>
<td>extName</td>
<td>String</td>
<td>The index extension</td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SNMPResponse - buildReferences()**

Builds references to actual instances for all identified indexes (in the discovery_snmp_ref table) that can be resolved.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

**SNMPResponse - getAmbiguousOIDs(String refBase, String refLeaf)**

Returns an array of OIDs that match the given ref_base and which have a defined, non-null ref_leaf.

The last character of ref_base may be a question mark; in that case, if the parent node is a table then all table entries are returned.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>refBase</td>
<td>String</td>
<td>The reference base</td>
</tr>
<tr>
<td>refLeaf</td>
<td>String</td>
<td>The reference leaf</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Array</td>
</tr>
</tbody>
</table>
**SNMPResponse - getOIDInt(String name)**

Returns the integer contained in the OID object with the given MIB name.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The MIB name</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The integer contained in the OID object, or zero if the object could not be found or is empty.</td>
</tr>
</tbody>
</table>

**SNMPResponse - getOIDObj(String name)**

Retrieves the OID object with the given MIB name from the response.

This method does not work with numeric OID strings, only MIB names.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The MIB name</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OID object</td>
<td>The OID object with the given MIB name, or null if the object could not be found or if any table entries are encountered while walking down the named elements.</td>
</tr>
</tbody>
</table>

**SNMPResponse - getOIDTable(String parentName, String entryName)**

Returns a hashmap of table entries under the given parent MIB name, with the given entry MIB names.
### SNMPResponse - getOIDText(String name)

Returns the text contained in the OID object with the given MIB name.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The MIB name</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The text contained in the OID object. Returns null if the object could not be found, or if it has no text.</td>
</tr>
</tbody>
</table>

### SNMPResponse - normalize(Object cur)

Resolves SNMP tables into hashmaps and OID values into primitives.

All SNMP numeric types convert to a JavaScript number. An SNMP null type converts to a JavaScript null. An SNMP IP Address value converts to a Java IPAddressV4 instance. All other values convert to a JavaScript string.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cur</td>
<td>Object</td>
<td>The SNMP table or OID value.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

SOAPMessageV2 - Scoped, Global

The SOAPMessageV2 API allows you to send an outbound SOAP message using JavaScript.

Use the SOAPResponseV2 API to manage the response returned by the SOAP provider.

You can use this API in scoped applications, or within the global scope.

See also:
- Direct SOAPMessageV2 example
- Recordless SOAPMessageV2 example
- Asynchronous SOAPMessageV2 example
- SOAPMessageV2 MID server example

SOAPMessageV2 - execute()

Sends the SOAP message to the endpoint.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOAPResponseV2 - Scoped, Global</td>
<td>Response returned by the SOAP provider.</td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); // Might throw exception if message doesn't exist or not visible due to scope.
var response = sm.execute(); // Might throw exception if http connection timed out or some issue with sending request itself because of encryption/decryption of password.
```
SOAPMessageV2 - executeAsync()

Sends the SOAP message to the ECC queue.

SOAP messages in the ECC queue are processed by the SOAPClient business rule.

By default, this business rule does not run asynchronously. To configure this business rule to run asynchronously, set the When value to Async and add current.update() to the end of the Script. The instance does not wait for a response from the web service provider when sending a message through the ECC queue.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOAPResponseV2 - Scoped, Global</td>
<td>Response returned by the SOAP provider.</td>
</tr>
</tbody>
</table>

Note: Attempting to use the SOAP response object before the response has been processed may result in a timeout error.

```javascript
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); //Might throw exception if message doesn't exist or not visible due to scope.
var response = sm.executeAsync();
```

SOAPMessageV2 - getEndpoint()

Gets the endpoint for the SOAP message.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>URL of the SOAP web service provider.</td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); // Might throw exception if message doesn't exist or not visible due to scope.
var endpoint = sm.getEndpoint();
```

**SOAPMessageV2 - getRequestBody()**

Returns the content of the SOAP message body.

ℹ️ **Note:** Before calling the `getRequestBody()` method, you must call the `execute()` method to obtain the response object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>SOAP message body.</td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.SOAPMessageV2("StockQuote","StockQuoteSoap.GetQuote"); // Might throw exception if message doesn't exist or not visible due to scope.
var response = sm.execute();
var requestBody = response.getRequestBody();
```

**SOAPMessageV2 - getRequestMethod(String headerName)**

Gets the value for an HTTP header specified by the SOAP client.

By default, this method cannot return the value for a header set automatically by the system. To grant this method access to all headers, set the property `glide.http.log_debug` to true.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>headerName</td>
<td>String</td>
<td>Request header you want to get the value for.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the specified header.</td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); // Might throw exception if message doesn't exist or not visible due to scope.
var header = sm.getRequestHeader("Accept");
```

### SOAPMessageV2 - getRequestHeaders()

Gets HTTP headers that were set by the SOAP client and the associated values.

This method does not return headers set automatically by the system. To configure this method to return all headers, set the property glide.http.log_debug to true.

```javascript
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); // Might throw exception if message doesn't exist or not visible due to scope.
var requestHeaders = sm.getRequestHeaders();
```

### SOAPMessageV2 - setBasicAuth(String userName, String userPass)

Sets basic authentication headers for the SOAP message.
Setting basic authentication headers using this method overrides basic authentication values defined in the SOAP message record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userName</td>
<td>String</td>
<td>Username to use when authenticating the SOAP message.</td>
</tr>
<tr>
<td>userPass</td>
<td>String</td>
<td>Password for the specified user.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); //Might throw exception if message doesn't exist or not visible due to scope.
sm.setBasicAuth("username","password");
```

**SOAPMessageV2 - setEccCorrelator(String correlator)**

Associates outbound requests and the resulting response record in the ECC queue.

This method only applies to SOAP messages sent through a MID Server. The correlator provided populates the **Agent correlator** field on the ECC queue record for the response. Provide a unique correlator for each outbound request to associate the correct results in the ECC queue with the request when designing asynchronous automation through a MID Server.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>correlator</td>
<td>String</td>
<td>Unique identifier</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); // Might throw exception if message doesn't exist or not visible due to scope.
sm.setEccCorrelator("unique_id");

SOAPMessageV2 - setEccParameter(String name, String value)

Overrides a value from the database by writing to the SOAP message payload.

This method only applies to SOAP messages sent through a MID Server. Use this method when a value from the SOAP message in the database is invalid, such as when the endpoint URL is longer than the maximum SOAP endpoint field length.

These are valid values for the name parameter.
- **source**: The endpoint URL.
- **name**: The SOAP message function to run.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the ECC parameter.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value to assign to the specified ECC parameter.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); // Might throw exception if message doesn't exist or not visible due to scope.
sm.setEccParameter("source","http://very.long.endpoint");

SOAPMessageV2 - setEndpoint(String endpoint)

Sets the endpoint for the SOAP message.

By default, the SOAP message uses the endpoint specified in the SOAP message record. Use this method to override the default. You must call this method when using the SOAPMessageV2() constructor with no parameters.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>endpoint</td>
<td>String</td>
<td>URL of the SOAP web service provider you want to interface with.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.SOAPMessageV2();
sm.setEndpoint("http://web.service.endpoint");
```

**SOAPMessageV2 - setHttpTimeout(Number timeoutMs)**

Sets the amount of time the SOAP message waits for a response from the web service provider before the request times out.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeoutMs</td>
<td>Number</td>
<td>Amount of time to wait for a response from the web service provider, in milliseconds.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); // Might throw exception if message doesn't exist or not visible due to scope.
sm.setHttpTimeout(6000);
```

**SOAPMessageV2 - setLogLevel(String level)**

Sets the log level for this message and the corresponding response.

Setting a log level using the SOAPMessageV2 API overrides the log level configured on the SOAP message record. This log level may
not apply if the endpoint domain is excluded, or if the property `glide.outbound_http_log.override` is true. To view outbound web service logs, navigate to **System Logs > Outbound HTTP Requests**.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>level</td>
<td>String</td>
<td>The log level. Valid values are basic, elevated, and all.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**SOAPMessageV2 - setMIDServer(String midServerName)**

Configures the SOAP message to be sent through a MID Server.

By default, the SOAP message uses the MID Server specified in the SOAP message function record. Use this method to override the default.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>midServerName</td>
<td>String</td>
<td>Name of the MID Server you want to send the SOAP message through. Your instance must have an active MID Server with the specified name.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**SOAPMessageV2 - setMutualAuth(String profileName)**

Sets the mutual authentication protocol profile for the SOAP message.

Setting a protocol profile using this method overrides the protocol profile selected for the SOAP message record.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>profileName</td>
<td>String</td>
<td>Name of the protocol profile to use for mutual authentication.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); //Might throw exception if message doesn't exist or not visible due to scope.
sm.setMutualAuth("auth_profile_name");
```

### SOAPMessageV2 - setRequestBody(String requestBody)

Sets the body content to send to the web service provider.

When you set the body content using this method, variables in the body are not substituted for parameters from the SOAP message function record. You must explicitly define all values within the SOAP message body.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestBody</td>
<td>String</td>
<td>Body of the SOAP message.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); //Might throw exception if message doesn't exist or not visible due to scope.
var body = "<SOAP message body>";
sm.setRequestBody(body);
```
**SOAPMessageV2 - setRequestHeader(String headerName, String headerValue)**

Sets an HTTP header in the SOAP message to the specified value.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>headerName</td>
<td>String</td>
<td>Name of the header.</td>
</tr>
<tr>
<td>headerValue</td>
<td>String</td>
<td>Value to assign to the specified header.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); // Might throw exception if message doesn't exist or not visible due to scope.
sm.setRequestHeader("Accept","Application/json");
```

**SOAPMessageV2 - setSOAPAction(String soapAction)**

Defines the SOAP action this SOAP message performs.

The WSDL for your web service provider lists SOAP actions you can perform. You must call this method when using the `SOAPMessageV2()` constructor with no parameters.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>soapAction</td>
<td>String</td>
<td>SOAP action this SOAP message performs.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var sm = new sn_ws.SOAPMessageV2();
sm.setSOAPAction("GetQuote");
```
//construct SOAP message by specifying endpoint and auth
sm.execute();

SOAPMessageV2 - setStringParameter(String name, String value)
Sets a variable with the specified name from the SOAP message record to the specified value.

XML reserved characters in the value are converted to the equivalent escaped characters.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the SOAP message variable.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value to assign to the specified variable.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); //Might throw exception if message doesn't exist or not visible due to scope.
sm.setStringParameter("symbol","NOW");
```

SOAPMessageV2 - setStringParameterNoEscape(String name, String value)
Sets a variable with the specified name from the SOAP message record to the specified value.

This method is equivalent to setStringParameter but does not escape XML reserved characters.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the SOAP message variable.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value to assign to the specified variable.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); //Might throw exception if message doesn't exist or not visible due to scope.
sm.setStringParameterNoEscape("symbol","NOW");
```

SOAPMessageV2 - setWSSecurity(String keystoreId, String keystoreAlias, String keystorePassword, String certificateId)

Sets web service security values for the SOAP message.

Setting security values using this method overwrites web service security values defined for the SOAP message record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>keystoreId</td>
<td>String</td>
<td>Sys_id of the Java or PKCS12 key store to use.</td>
</tr>
<tr>
<td>keystoreAlias</td>
<td>String</td>
<td>Alias that identifies the public and private keys.</td>
</tr>
<tr>
<td>keystorePassword</td>
<td>String</td>
<td>Password assigned to the key store record.</td>
</tr>
<tr>
<td>certificateId</td>
<td>String</td>
<td>Sys_id of the trusted server certificate.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); //Might throw exception if message doesn't exist or not visible due to scope.
sm.setWSSecurity("70d65e074f3812001f6eac118110c71a","Quote keys","UXr82cqX75z7MaSa+EyjGA==","ba969a074f3812001f6eac118110c76d");
```

SOAPMessageV2 - SOAPMessageV2()

Instantiates an empty SOAPMessageV2 object.
When using an object instantiated this way, you must manually specify a SOAP action and endpoint.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.SOAPMessageV2();
```

**SOAPMessageV2 - SOAPMessageV2(String soapMessage, String soapFunction)**

Instantiates a SOAPMessageV2 object from a SOAP message record and a function associated with that record.

Values such as the endpoint, authentication, or MID Server settings from the SOAP message record apply to this object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>soapMessage</td>
<td>String</td>
<td>SOAP message record you want to use as the base for this object.</td>
</tr>
<tr>
<td>soapFunction</td>
<td>String</td>
<td>SOAP function you want to execute. Available SOAP functions depend on the WSDL supplied by the web service provider.</td>
</tr>
</tbody>
</table>

```javascript
var sm = new sn_ws.SOAPMessageV2("StockQuote","GetQuote"); //Might throw exception if message doesn't exist or not visible due to scope.
```

**SOAPResponseV2 - Scoped, Global**

The SOAPResponseV2 API allows you to use the data returned by an outbound SOAP message in JavaScript code.

A SOAPResponseV2 object is returned by the SOAPMessageV2 methods `execute()` and `executeAsync()`.

You can use this API in scoped applications, or within the global scope.

See also:

- Direct SOAPMessageV2 example
- Recordless SOAPMessageV2 example
• Asynchronous SOAPMessageV2 example
• SOAPMessageV2 MID server example

SOAPResponseV2 - getAllHeaders()
Returns all headers contained in the response, including any duplicate headers.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>List&lt;GlideHTTPHeader&gt;</td>
<td>List of headers contained in the response. Each header is represented as a GlideHTTPHeader object which contains the header name and value.</td>
</tr>
</tbody>
</table>

```javascript
var r = new sn_ws.SOAPMessageV2('<A SOAP message>', 'get');
var response = r.execute();
var headers = response.getAllHeaders();
for(var i in headers){
    gs.info(headers[i].name + ': ' + headers[i].value);
}
```

SOAPResponseV2 - getBody()
Gets the content of the SOAP response body.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>SOAP response body.</td>
</tr>
</tbody>
</table>
var body = response.getBody();

SOAPResponseV2 - get_cookies()

Returns all cookies included in the response.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Array of strings representing cookies. Iterate through the array to perform operations on each cookie.</td>
</tr>
</tbody>
</table>

Display individual cookies from the response.

```javascript
var cookies = response.getCookies();
for (var i = 0; i < cookies.length; i++) {
    gs.info('cookie: ' + cookies.get(i));
}
```

Output:

```
cookie: JSESSIONID=4135AA97A5D12DA22EF614AA2B0CAF8.node20; Path=/; Secure; HttpOnly
cookie: SABASESSIONID=370152970.36895.0000; path=/
```

SOAPResponseV2 - get_error_code()

Gets the numeric error code if there was an error during the SOAP transaction.

This error code is specific to the Now Platform, it is not an HTTP error code. Provide this error code if you require assistance from Customer Service and Support.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Numeric error code, such as 1 for a socket timeout.</td>
</tr>
</tbody>
</table>

```javascript
var errorCode = response.getErrorCode();
```

**SOAPResponseV2 - getErrorMessage()**

Gets the error message if there was an error during the SOAP transaction.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Error message</td>
</tr>
</tbody>
</table>

```javascript
var errorMsg = response.getErrorMessage();
```

**SOAPResponseV2 - getHeader(String name)**

Gets the value for a specified HTTP header.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the header that you want the value for, such as Set-Cookie.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the specified header.</td>
</tr>
</tbody>
</table>

```javascript
var headerVal = response.getHeader("Accept");
```
SOAPResponseV2 - getHeaders()

Gets all HTTP headers returned in the SOAP response and the associated values.

ℹ️ **Note:** If a header is present more than once in the response, such as a Set-Cookie header, this function returns only the last of the duplicate headers. To return all headers including duplicates, use the `getAllHeaders()` function.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object that maps the name of each header to the associated value.</td>
</tr>
</tbody>
</table>

```javascript
var headers = response.getHeaders();
```

SOAPResponseV2 - getStatusCode()

Gets the numeric HTTP status code returned by the SOAP provider.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Numeric status code returned by the SOAP provider, such as 200 for a successful response.</td>
</tr>
</tbody>
</table>

```javascript
var statusCode = response.getStatusCode();
```

SOAPResponseV2 - haveError()

Indicates if there was an error during the SOAP transaction.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>Returns true if there was an error, false if there was no error.</td>
</tr>
</tbody>
</table>

```javascript
var error = response.haveError();
```

### SOAPResponseV2 - waitForResponse(Number timeoutSecs)

Sets the amount of time the instance waits for a response from the web service provider.

This method overrides the property glide.soap.outbound.ecc_response.timeout for this SOAP response.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeoutSecs</td>
<td>Number</td>
<td>Amount of time, in seconds, to wait for this response.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
response.waitForResponse(60);
```

### spAriaUtil - Client

Show messages on a screen reader.

spAriaUtil is an angular service that you can use in Service Portal widget client scripts.
spAriaUtil - sendLiveMessage(String message)

Announce a message to a screen reader.

The sendLiveMessage() method injects text into an aria-live region on the page. The default setting for an aria-live region is assertive, which means that messages are announced immediately. This can annoy and confuse users if used too frequently.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The message to be shown.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
function(spAriaUtil) {
  /* widget controller */
  spAriaUtil.sendLiveMessage('Hello world!');
}
```

spContextManager - Client

Make data from a Service Portal widget available to other applications and services in a Service Portal page. For example, pass widget data to Agent Chat when it opens in a Service Portal page.

spContextManager is an AngularJS service that you can use in Service Portal widget client scripts.

Keys passed to this API are unique per page. For example, if the 'agent-chat' key is already initialized by another widget on the page through the addContext() method, you must use the updateContextForKey() method to update the key's data. Available keys include:

- **agent-chat**: Sends widget data to Agent Chat when it opens in a Service Portal page.

For more information about passing data to Agent Chat, see Configure Agent Chat in Service Portal.
**spContextManager - addContext(String key, Object context)**

Initializes a key and adds widget data as the value. For example, add data to the 'agent-chat' key to make it available to Agent Chat.

Use this method the first time data is added to a specific key on a Service Portal page. If the key is already used by another widget on the page, use the `updateContextForKey()` method instead.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the key to send the data. Available keys include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>agent-chat</strong>: Sends widget data to Agent Chat when it opens in a Service Portal page.</td>
</tr>
<tr>
<td>context</td>
<td>Object</td>
<td>Widget data in JSON format to send to the application or service specified in the key parameter. For example, <code>{approval_count': 5}</code>.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Pass `approval_count` to Agent Chat. When a user initiates an Agent Chat conversation from the Service Portal homepage, the system appends `&sysparm_approval_count=5` to the Agent Chat iframe URL.

```javascript
def function ($scope, spContextManager) {
    spContextManager.addContext('agent-chat', {
        'approval_count': 5
    });
}
```

**spContextManager - getContext()**

Returns each key and associated data object defined by any widget on the page.

Using this method may affect performance. Use this method to understand which keys are initialized on the page and to get their current values. If you
know which key you need to access, use the `getContextForKey()` method instead.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Each key and associated data object defined on the page.</td>
</tr>
</tbody>
</table>

```javascript
function ($scope, spContextManager) {
  spContextManager.getContext();
}
```

### `spContextManager - getContextForKey(String key, Boolean returnPromise)`

Returns the widget data associated with a key.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the key to get context from. Available keys include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>agent-chat</strong>: Sends widget data to Agent Chat when it opens in a Service Portal page.</td>
</tr>
<tr>
<td>returnPromise</td>
<td>Boolean</td>
<td>Flag that determines whether to return the data associated with a key as a promise or an object. Values include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• True: return the data as a promise. Use this option if another widget on the page uses the <code>addContext()</code> method to initialize the same key. Returning a promise prevents returning an undefined object when the key is not yet initialized.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• False: returns an object containing the data associated with the key.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Promise</td>
<td>If <code>returnPromise</code> is true, returns a promise that is fulfilled when another widget on the page initializes the key.</td>
</tr>
<tr>
<td>Object</td>
<td>If <code>returnPromise</code> is false, returns an object containing the data associated with the key. For example, <code>{approval_count: 5}</code>.</td>
</tr>
</tbody>
</table>

Pass `approval_count` to Agent Chat. When a user initiates an Agent Chat conversation from the Service Portal homepage, the system appends `&sysparm_approval_count=5` to the Agent Chat iframe URL.

```javascript
function ($scope, spContextManager) {
  spContextManager.getContextForKey('agent-chat', true).then(function(context) {
    context = context || {};
    context.approval_count = 5;
    spContextManager.updateContextForKey('agent-chat', context);
  });
}
```

**spContextManager - updateContextForKey(String key, Object context)**

Sends data to an existing key. For example, if another widget on the page uses the 'agent-chat' key to pass data to the Agent Chat configuration, you must update the context of the key rather than using the `addContext()` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>String</td>
<td>Name of the key to send the data. Available keys include:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>agent-chat</strong>: Sends widget data to Agent Chat when it opens in a Service Portal page.</td>
</tr>
<tr>
<td>context</td>
<td>Object</td>
<td>Widget data in JSON format to send to the application or service specified in the key parameter. For example, <code>{approval_count: 5}</code>.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Pass `approval_count` to Agent Chat. When a user initiates an Agent Chat conversation from the Service Portal homepage, the system appends `&sysparm_approval_count=5` to the Agent Chat iframe URL.

```javascript
function ($scope, spContextManager) {
    spContextManager.getContextForKey('agent-chat', true).then(function(context) {
        context = context || {};
        context.approval_count = 5;
        spContextManager.updateContextForKey('agent-chat', context);
    });
}
```

**spModal - Client**

Show alerts, prompts, and confirmation dialogs in Service Portal widgets. The SPModal class is available in Service Portal client scripts.

You can use `spModal.open()` to display a widget in a modal dialog. The spModal class is a lightweight wrapper for Angular UI bootstrap's `$uibModal`.

**spModal - alert(String message).then(fn)**

Displays an alert.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The message to display.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>The promise contains a single argument that contains true or false.</td>
</tr>
</tbody>
</table>

```html
// HTML template
<button ng-click="c.onAlert()" class="btn btn-default">
    Alert
</button>
```
// Client script
function(spModal) {
    var c = this;
    c.onAlert = function () {
        spModal.alert('How do you feel today?').then(function (answer) {
            c.simple = answer;
        });
    }
}

spModal - confirm(String message).then(fn)
Displays a confirmation message.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The message to display.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>The promise contains a single argument that contains true or false.</td>
</tr>
</tbody>
</table>

// HTML template
<button ng-click="c.onConfirm()" class="btn btn-default"> Confirm </button>
<span>{{c.confirmed}}</span>

// Client script
function(spModal) {
    var c = this;
    c.onConfirm = function () {
        c.confirmed = "asking";
        spModal.confirm("Can you confirm or deny this?").then(function(confirmed) {
            c.confirmed = confirmed; // true or false
        })
    }
}

Confirm with HTML message
// HTML template
<button ng-click="c.onConfirmEx()" class="btn btn-default">
  Confirm - HTML message
</button>
<span>{{c.confirmed}}</span>

// Client script
function(spModal) {
  var c = this;
  // more control, passing options
  c.onConfirmEx = function() {
    c.confirmed = "asking";
    var warn = '<i class="fa fa-warning" aria-hidden="true"></i>';
    spModal.open({
      title: 'Delete this Thing?',
      message: warn + ' Are you <b>sure</b> you want to do that?'
    }).then(function(confirmed) {
      c.confirmed = confirmed;
    })
  }
}

**spModal - open(Object options).then(fn)**

Opens a modal window using the specified options.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Object</td>
<td>An object that may have these properties.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• title - a string that can be HTML that goes in the header. The default is empty.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• message - a string that can be HTML that goes in the header. The default is empty.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• buttons - an array that contains the buttons to show on the dialog. The default is Cancel and OK.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• input - a Boolean. When true shows an input field on the dialog. The default is false.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• value - a string containing the value of the input field. The default is empty.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• widget</td>
<td>string</td>
<td>a string that identifies the widget ID or sys_id to embed in the dialog. The default is empty.</td>
</tr>
<tr>
<td>• widgetInput</td>
<td>object</td>
<td>an object to send the embedded widget as input. The default is null.</td>
</tr>
<tr>
<td>• shared</td>
<td>object</td>
<td>a client-side object to share data with the embedded widget client script.</td>
</tr>
<tr>
<td>• size</td>
<td>string</td>
<td>a string indicating the size of the window. Can be 'sm' or 'lg'. The default is empty.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example of a prompt with a label

```html
//HTML template
<button ng-click="c.onOpen()" class="btn btn-default">
    Prompt with label
</button>
<div ng-show="c.name">
    You answered <span>{{c.name}}</span>
</div>
```

```javascript
//Client code
function(spModal) {
    var c = this;
    c.onOpen = function() {
        //ask the user for a string
        spModal.open({
            title: 'Give me a name',
            message: 'What would you like to name it?',
            input: true,
            value: c.name
        }).then(function(name) {
            c.name = name;
        })
    }
}
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example of agree with custom buttons.

//HTML template
<button ng-click="c.onAgree()" class="btn btn-default">
  Agree
</button>
<div ng-show="c.agree">
  You answered {{c.agree}}
</div>

//Client script
function(spModal) {
  var c = this;
  c.onAgree = function() {
    // ask the user for a string
    // note embedded html in message
    var h = '<h4>Apple likes people to agree to lots of stuff</h4>'
    // Line feeds added to the following lines for presentation formatting.
    var m = 'Your use of Apple software or hardware products is based on the software license and other terms and conditions in effect for the product at the time of purchase. Your agreement to these terms is required to install or use the product.'
    spModal.open({
      title: 'Do you agree?',
      message: h + m,
      buttons: [
        {label:'✘ ${No}', cancel: true},
        {label:'✔ ${Yes}', primary: true}
      ]
    }).then(function() {
      c.agree = 'yes';
    }, function() {
      c.agree = 'no';
    })
  }
}

Example of embedded widget

//HTML template
<button ng-click="c.onWidget('widget-cool-clock')" class="btn btn-default">
  Cool Clock
</button>
//Client script
function(spModal) {
    var c = this;
    c.onWidget = function(widgetId, widgetInput) {
        spModal.open({
            title: 'Displaying widget ' + widgetId,
            widget: widgetId,
            widgetInput: widgetInput || {}
        }).then(function(){
            console.log('widget dismissed');
        })
    }
}

spModal - prompt(String message, String default).then(fn)
Displays a prompt for user input.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The message to display.</td>
</tr>
<tr>
<td>default</td>
<td>String</td>
<td>A default value to use if the user does not provide a response.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The promise contains the user's response, or the default value if the user does not enter a response.</td>
</tr>
</tbody>
</table>

//HTML template
<button ng-click="c.onPrompt()" class="btn btn-default">
    Prompt
</button>
<div ng-show="c.name">
    You answered <span>{{c.name}}</span>
</div>
// Client script
function(spModal) {
    var c = this;
    c.onPrompt = function() {
        spModal.prompt("Your name please", c.name).then(function(name) {
            c.name = name;
        });
    }
}

SPScriptedFacet - Scoped

Define facet items, filters, or mapped queries for a facets object.

The SPScriptedFacet API can only be used in a facet generation script in a Service Portal search source. The facet generation script is only visible when Is scripted source is selected.

There is no constructor for this class. Instead, use the createFacet() or createMultiChoiceFacet() methods of the SPScriptedFacetService class to generate a facets object.

Scoped SPScriptedFacet - addFacetItem(String label, Object valueObj)

Adds facet items or mapped queries to a facets object.

Before adding facet items to a facets object, create the facets object using the createFacet() or createMultiChoiceFacet() methods of the SPScriptedFacetService class.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>String</td>
<td>The display label for the facet item or mapped query.</td>
</tr>
<tr>
<td>valueObj</td>
<td>Object</td>
<td>The facet item or mapped query for the facet. Can only contain types String, Number, Boolean, and Double.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
```javascript
function createMultiChoiceFacetFromKB(label, variable, values, orderBy){
    if(!values || values.length == 0)
        return;
    var kbFacet = facetService.createMultiChoiceFacet(label, variable);
    var kbMap = {};
    for(var i in values) {
        var key = values[i].label;
        var value = values[i].id;
        var parent = new KBParentData().getParentCategory(value, key);
        key = parent.join(' > ');
        kbMap[key] = value;
    }
    var labels = Object.keys(kbMap).sort();
    for (i = 0; i < labels.length; i++) {
        var currentLabel = labels[i];
        kbFacet.addFacetItem(currentLabel, kbMap[currentLabel]);
        //adds facet search options for each kbMap entry
    }
}
```

**SPScriptedFacetService - Scoped**

Generate a multi choice or single choice facets object for an advanced search source.

The `SPScriptedFacetService` API can only be used in a facet generation script in a Service Portal search source. The facet generation script is only visible when **Is scripted source** is selected.

There is no constructor for this class. Instead, use the `createFacet()` or `createMultiChoiceFacet()` methods to generate a facets object.

**Scoped SPScriptedFacetService - createFacet(String label, String id)**

Creates a single choice facets object.

After creating the facets object, add facet items or mapped queries to the facet using the `addFacetItem()` method of the `SPScriptedFacet` class.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>String</td>
<td>Label for the facet.</td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>ID for the facet.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Single choice facets object.</td>
</tr>
<tr>
<td></td>
<td>Returns an error when:</td>
</tr>
<tr>
<td></td>
<td>• A duplicate label or ID is found.</td>
</tr>
<tr>
<td></td>
<td>• A label or ID is not defined.</td>
</tr>
</tbody>
</table>

Can only be used in a facet generation script in a Service Portal advanced search source.

```javascript
var stateFacet = facetService.createFacet("State", "state");
```

Scoped SPScriptedFacetService - createMultiChoiceFacet(String label, String id)

Creates a multi choice facets object.

After creating the facets object, add facet items or mapped queries to the facet using the `addFacetItem()` method of the SPScriptedFacet class.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>String</td>
<td>Label for the facet.</td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>ID for the facet.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Multi choice facets object.</td>
</tr>
<tr>
<td></td>
<td>Returns an error when:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• A duplicate label or ID is found.</td>
</tr>
<tr>
<td></td>
<td>• A label or ID is not defined.</td>
</tr>
</tbody>
</table>

```javascript
function createMultiChoiceFacetFromKB(label, variable, values, orderBy){
    if(!values || values.length == 0)
        return;

    var kbFacet = facetService.createMultiChoiceFacet(label, variable);
    var kbMap = {};
    for(var i in values) {
        var key = values[i].label;
        var value = values[i].id;
        var parent = new KBParentData().getParentCategory(value, key);
        key = parent.join(' > ');
        kbMap[key] = value;
    }

    var labels = Object.keys(kbMap).sort();
    for (i = 0; i < labels.length; i++) {
        var currentLabel = labels[i];
        kbFacet.addFacetItem(currentLabel, kbMap[currentLabel]);
        //adds facet search options for each kbMap entry
    }
}
```

**spUtil - Client**

Utility methods to perform common functions in a Service Portal widget client script.

**spUtil - addErrorMessage(String message)**

Displays a notification error message.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Error message to display.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

spUtil.addErrorMessage("There has been an error processing your request")

spUtil - addInfoMessage(String message)
Displays a notification info message.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Message to display.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

spUtil.addInfoMessage("Your order has been placed")

spUtil - addTrivialMessage(String message)
Displays a trivial notification message.

Trivial messages disappear after a short period of time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Message to display.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
spUtil.addTrivialMessage("Thanks for your order")

**spUtil - createUid()**

Create a unique ID.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>A unique 32 character id.</td>
</tr>
</tbody>
</table>

**spUtil - get(String widgetId Object data)**

Returns a widget model by ID or sys_id.

Use this method to embed a widget model in a widget client script. The callback function returns the full widget model.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>widgetId</td>
<td>String</td>
<td>Widget ID or sys_id of the widget to embed.</td>
</tr>
<tr>
<td>data</td>
<td>Object</td>
<td>(Optional) Name/value pairs of parameters to pass to the widget model.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Model of the embedded widget.</td>
</tr>
</tbody>
</table>

Without data passed

```javascript
spUtil.get("widget-cool-clock").then(function(response) {
  c.coolClock = response;
});
```
With data passed

```javascript
spUtil.get('pps-list-modal', {title: c.data.editAllocations,
    table: 'resource_allocation',
    queryString: 'GROUPBYuser^resource_plan=' + c.data.sysId,
    view: 'resource_portal_allocations' }).then(function(response) {
    var formModal = response;
    c.allocationListModal = response;
});
```

**spUtil - getHost()**

Get complete host domain.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The complete host domain, for example <code>hi.servicenow.com</code></td>
</tr>
</tbody>
</table>

**spUtil - getHeaders()**

Retrieve all headers to be used for API calls.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>All headers to be used for API calls.</td>
</tr>
</tbody>
</table>
spUtil - getPreference(String preference, Function callback)
Execute callback with User Preference response by passing Preference name.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>preference</td>
<td>String</td>
<td>The name of the preference.</td>
</tr>
<tr>
<td>callback</td>
<td>Function</td>
<td>Define the callback function.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

spUtil - getURL()
Get current service portal url information.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The current service portal url.</td>
</tr>
</tbody>
</table>

spUtil - format(String template, Object data)
Formats a string as an alternative to string concatenation.

Use this method to build a string with variables.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>template</td>
<td>String</td>
<td>String template with values for substitution.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Object</td>
<td>Object containing variables for substitution.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A formatted string.</td>
</tr>
</tbody>
</table>

```
spUtil.format('An error occurred: {error} when loading {widget}', {error: '404', widget: 'sp-widget'})
```

Output:

'An error occurred: 404 when loading sp-widget'

**spUtil - isMobile()**

Check if current client is a mobile device.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the current client is a mobile device; otherwise, false</td>
</tr>
</tbody>
</table>

**spUtil - parseAttributes(String attributes)**

Brief description of the method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>String</td>
<td>String containing comma separated attributes, such as the Attributes field of a dictionary record.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Array object containing the parsed attributes.</td>
</tr>
</tbody>
</table>

[Working to add example script with example output]

**spUtil - recordWatch(Object $scope, String table, String filter, Function callback)**

Watches for updates to a table or filter and returns the value from the callback function.

Allows a widget developer to respond to table updates in real-time. For instance, by using `recordWatch()`, the Simple List widget can listen for changes to its data table. If records are added, removed, or updated, the widget updates automatically.

ℹ️ **Note:** When passing the `$scope` argument into the `recordWatch()` function, inject `$scope` into the parameters of your client script function.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$scope</td>
<td>Object</td>
<td>Scope of the data object updated by the callback function.</td>
</tr>
<tr>
<td>table</td>
<td>String</td>
<td>Watched table.</td>
</tr>
<tr>
<td>filter</td>
<td>String</td>
<td>Filter for fields to watch.</td>
</tr>
<tr>
<td>callback</td>
<td>Function</td>
<td>Optional. Parameter to define the callback function.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Promise</td>
<td>Return value of the callback function.</td>
</tr>
</tbody>
</table>

```javascript
//A simple recordWatch function.
spUtil.recordWatch($scope, "live_profile", "sys_id=" + liveProfileId);

//In a widget client script
function(spUtil, $scope) {
    /* widget controller */
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
var c = this;

// Registers a listener on the incident table with the filter active=true,
// meaning that whenever something changes on that table with that filter,
// the callback function is executed.
// The callback function takes a single parameter 'response', which contains
// the property 'data'. The 'data' property contains information about the changed
// record.
spUtil.recordWatch($scope, "incident", "active=true", function(response) {

    // Returns the data inserted or updated on the table
    console.log(response.data);

});

spUtil - refresh(Object $scope)

Calls the server and replaces the current options and data with the server
response.

Calling spUtil.refresh() is similar to calling server.refresh(). However, when you
call spUtil.refresh(), you can define the $scope object.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>$scope</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Object</td>
</tr>
</tbody>
</table>

spUtil - scrollTo(String selector, Number time)

Scroll to element with specified selector, over specified period of time.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>selector</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>Number</td>
<td>The time, in milliseconds, taken to scroll to the specified selector.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

spUtil - setBreadCrumb()

Update the header breadcrumbs.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$scope</td>
<td>Object</td>
<td>The scope defined for the table.</td>
</tr>
<tr>
<td>breadcrumbs</td>
<td>Array</td>
<td>An array of conditions used to create the breadcrumb filter.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

spUtil - setPreference(String preference, String value)

Set a user preference.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>preference</td>
<td>String</td>
<td>The preference name</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>The preference value</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**spUtil - setSearchPage(String searchPage)**

Update the search page.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>searchPage</td>
<td>String</td>
<td>The name of the search page.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**spUtil - update(Object $scope)**

Updates the data object on the server within a given scope.

This method is similar to `server.update()`, but includes a `$scope` parameter that defines the scope to pass over.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$scope</td>
<td>Object</td>
<td>The scope defined for the update.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The updated data object.</td>
</tr>
</tbody>
</table>

**StandardCredentialsProvider - Scoped, Global**

Use the `StandardCredentialsProvider` API to retrieve credential information.
You can use this API in scoped applications, or within the global scope. In scoped scripts, use the `sn_cc` namespace identifier.

This API provides methods to retrieve credential information by sys_id and by specified credential attributes.

```javascript
//Get a single credential
var provider = new sn_cc.StandardCredentialsProvider();
var credential = provider.getCredentialByID("f43c6d40a0a0b5700c77f9bf387afe3");
var userName = credential.getAttribute("user_name");
var password = credential.getAttribute("password");
//using getAttribute for new keys in extended tables, for example
//cloud management credential has the "user_public_key" attribute
var userPublicPublicKey = credential.getAttribute("user_public_key");

//Get a list of SSH credentials
var provider = new sn_cc.StandardCredentialsProvider();
var credentials = provider.getCredentials(['ssh']);
for (var i = 0; i < credentials.length; i++) {
    var credential = credentials[i];
    gs.info(credential.getAttribute("name"));
}
```

**StandardCredentialsProvider - StandardCredentialsProvider()**

Instantiates a credentials provider object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var provider = new sn_cc.StandardCredentialsProvider();
var credentials = provider.getCredentialByID("ef43c6d40a0a0b5700c77f9bf387afe3");
```

**StandardCredentialsProvider - getCredentials(String types, String tags)**

Returns an array of all credentials that match the specified types and tags.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tags</td>
<td>String</td>
<td>Optional. Comma-separated list of tag names. For example, &quot;ssh, jdbc&quot;. Examples of valid calls:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• var credentials = provider.getCredentials(null, null);</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• var credentials = provider.getCredentials([&quot;ssh&quot;], &quot;&quot;);</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• var credentials = provider.getCredentials(new ArrayList&lt;String&gt;(), &quot;ssh, admin&quot;);</td>
</tr>
<tr>
<td>types</td>
<td>Array</td>
<td>Optional. Credential type names. For example, [&quot;ssh&quot;, &quot;windows&quot;] Note: If types is null or empty, any match returns a credential. If types is specified, the credentials whose type matches one of the types is returned.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StandardCredential</td>
<td>Credential record object.</td>
</tr>
</tbody>
</table>

This code example shows how to get the provider credentials for "ssh" credential types.

```java
var provider = new sn_cc.StandardCredentialsProvider();
var credentials = provider.getCredentials(["ssh"]);
```

This code example shows how to get the provider credentials for "ssh" and "windows" credential types that have tags of "admin".

```java
var provider = new sn_cc.StandardCredentialsProvider();
ArrayList<String> types = new ArrayList<>();
types.add("ssh");
types.add("windows");
JSONArray jsonArray = provider.getCredentials(types, "admin");
```

**StandardCredentialsProvider - getCredentialBysys_id(String sys_id)**

Returns the credential record object identified by the specified sys_id.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StandardCredential</td>
<td>Credential record object.</td>
</tr>
</tbody>
</table>

```plaintext
var provider = new sn_cc.StandardCredentialsProvider();
var credentials = provider.getCredentialByID("ef43c6d40a0b5700c77f9bf387afe3");
gs.info("User name: " + credentials.getAttribute("user_name"));
```

### StartDiscovery - Global

Starts discovery jobs.

Use this API during the discovery scanning phase.

### StartDiscovery - addAdditionalIP(String status, String ip, String sensor)

Adds an IP (or comma-separated list of IPs) to an existing discovery (used by networks discovery).

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>String</td>
<td>The discovery status</td>
</tr>
<tr>
<td>ip</td>
<td>String</td>
<td>An IP address, or a comma-separated list of IP addresses.</td>
</tr>
<tr>
<td>sensor</td>
<td>String</td>
<td>The discovery sensor</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
StartDiscovery - discoverFromSchedule(String ip, Boolean samePhase, String sensor)

Handles discovery of devices (whether basic or advanced) from a schedule.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip</td>
<td>String</td>
<td>An IP address, or a comma-separated list of IP addresses.</td>
</tr>
<tr>
<td>samePhase</td>
<td>Boolean</td>
<td></td>
</tr>
<tr>
<td>sensor</td>
<td>String</td>
<td>The discovery sensor</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

StartDiscovery - getCancelScript()

Returns the cancel script.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The cancel script</td>
</tr>
</tbody>
</table>

StartDiscovery - scheduleCancelJob()

If the schedule specifies a cancel time, schedules a job to cancel if necessary.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**StartDiscovery - startFromIP(DiscoverySchedule schedule, String ip)**

Starts a discovery for a single IP using the specified schedule to get information such as MID servers and behaviors.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>schedule</td>
<td>DiscoverySchedule</td>
<td>A DiscoverySchedule instance for the schedule to use.</td>
</tr>
<tr>
<td>ip</td>
<td>String</td>
<td>The IP address to discover in a dot-formatted string.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_id of the status record created.</td>
</tr>
</tbody>
</table>

**StartDiscovery - startFromSchedule(GlideRecord schedule, GlideRecord job)**

Invoked by the discovery script include to kick off a scheduled (or discover now) discovery.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>schedule</td>
<td>GlideRecord</td>
<td>The instance of discovery_schedule that this discovery is based on.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>job</td>
<td>GlideRecord</td>
<td>The instance of sys_trigger that triggered this discovery.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**StateManagementScriptableApi - Global**

Provides a set of methods for manipulating CI operational states and applying CI actions.

The StateManagementScriptableApi methods adhere to restrictions and allowances specified by not allowed CI actions, compatible CI actions, and not allowed operational transitions. If a method attempts to perform a restricted operation, the operation is blocked, an error is logged, and a task is created if appropriate.

The StateManagementScriptableApi methods are static methods. You access the methods using the `SNC.StateManagementScriptableApi` global object.

**StateManagementScriptableApi - addBulkCIAction(String requestorId, String sysIdList, String ciActionName, String ciActionListOld, String leaseTime)**

Add a CI action to a list of CIs.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestorId</td>
<td>String</td>
<td>A sys_id of a workflow context, or the GUID returned from the <code>registerOperator()</code> method.</td>
</tr>
<tr>
<td>sysIdList</td>
<td>String</td>
<td>A list comma separated list of CI sys_ids.</td>
</tr>
<tr>
<td>ciActionName</td>
<td>String</td>
<td>The CI action name.</td>
</tr>
<tr>
<td>ciActionListOld</td>
<td>String</td>
<td>(Optional) A comma separated list of old CI actions that all CIs should be in.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>leaseTime</td>
<td>String</td>
<td>(Optional) Time duration for which the lease is valid for specified CI Action. In the format HH:MM:SS.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A JSON formatted string with name-value pairs for</td>
</tr>
<tr>
<td></td>
<td>• result - true if the action is set for all CIs in the list; otherwise, false.</td>
</tr>
<tr>
<td></td>
<td>• errors - list of errors.</td>
</tr>
</tbody>
</table>

**StateManagementScriptableApi - extendCIActionLease(String requestorId, String ciSysId, String ciActionName, String leaseTime)**

Extend the CI-action-lease time for the registered user. If the previous lease has expired, the new lease time starts now.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestorId</td>
<td>String</td>
<td>A sys_id of a workflow context, or the GUID returned from the registerOperator() method.</td>
</tr>
<tr>
<td>ciSysId</td>
<td>String</td>
<td>The CI's sys_id.</td>
</tr>
<tr>
<td>ciActionName</td>
<td>String</td>
<td>The CI action name.</td>
</tr>
<tr>
<td>leaseTime</td>
<td>String</td>
<td>Time duration for which the lease is valid for specified CI Action. In the format HH:MM:SS.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A JSON formatted string with name-value pairs for</td>
</tr>
<tr>
<td></td>
<td>• result - true if the lease time has been set.</td>
</tr>
<tr>
<td></td>
<td>• errors - list of errors.</td>
</tr>
</tbody>
</table>
**StateManagementScriptableApi - getCIActions(String ciSysId)**

Returns a list of active CI actions for the specified CI.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ciSysId</td>
<td>String</td>
<td>The sys_id of the CI</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A JSON formatted string with name-value pairs for</td>
</tr>
<tr>
<td></td>
<td>• ciActions - a comma separated list of active CI actions for the CI, or</td>
</tr>
<tr>
<td></td>
<td>NO_ACTIVE_ACTION.</td>
</tr>
<tr>
<td></td>
<td>• errors - list of errors.</td>
</tr>
</tbody>
</table>

**StateManagementScriptableApi - getOperationalState(String ciSysId)**

Returns the CI's operational state.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ciSysId</td>
<td>String</td>
<td>Sys_id of the CI</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A JSON formatted string with name-value pairs for</td>
</tr>
<tr>
<td></td>
<td>• operationalStates - one of the string choice values of the operational_status field, or UNKNOWN.</td>
</tr>
<tr>
<td></td>
<td>• errors - list of errors.</td>
</tr>
</tbody>
</table>

**StateManagementScriptableApi - isCompatibleCIAction(String actionName, String otherActionName)**

Determines if the two specified actions are compatible.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>String</td>
<td>A CI action name</td>
</tr>
<tr>
<td>otherActionName</td>
<td>String</td>
<td>A CI action name</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the two specified CI actions are compatible.</td>
</tr>
</tbody>
</table>

StateManagementScriptableApi - isLeaseExpired(String requestorId, String ciSysId, String ciActionName)

Determines if the lease has expired for the requestor of a specified CI Action.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestorId</td>
<td>String</td>
<td>A sys_id of a workflow context, or the GUID returned from the registerOperator() method.</td>
</tr>
<tr>
<td>ciSysId</td>
<td>String</td>
<td>The CI's sys_id.</td>
</tr>
<tr>
<td>ciActionName</td>
<td>String</td>
<td>Name of the CI action.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A JSON formatted string with name-value pairs for</td>
</tr>
<tr>
<td></td>
<td>• result - true if the user’s lease has expired.</td>
</tr>
<tr>
<td></td>
<td>• errors - list of errors.</td>
</tr>
</tbody>
</table>

StateManagementScriptableApi - isNotAllowedAction(String ciType, String opsLabel, String actionName)

For a type of CI, determine if a CI action is not allowed for an operational state.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ciType</td>
<td>String</td>
<td>The CI type</td>
</tr>
<tr>
<td>opsLabel</td>
<td>String</td>
<td>The operational state</td>
</tr>
<tr>
<td>actionName</td>
<td>String</td>
<td>The CI action name</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if the action is not allowed on the specified CI type in the specified operational state.</td>
</tr>
</tbody>
</table>

**StateManagementScriptableApi - isNotAllowedOpsTransition(String ciType, String opsLabel, String transitionOpsLabel)**

For a type of CI, determine if an operational state transition is not allowed.

## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ciType</td>
<td>String</td>
<td>The CI type</td>
</tr>
<tr>
<td>opsLabel</td>
<td>String</td>
<td>The label of the beginning operational state.</td>
</tr>
<tr>
<td>transitionOpsLabel</td>
<td>String</td>
<td>The label of the ending operational state.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Returns true if specified operational state transition is not allowed on the specified CI type.</td>
</tr>
</tbody>
</table>

**StateManagementScriptableApi - isValidRequestor(String requestorId)**

Determine if the specified requestor is a valid active workflow user or a registered user.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestorId</td>
<td>String</td>
<td>A sys_id of a workflow context, or the GUID returned from the <code>registerOperator()</code> method.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String   | A JSON formatted string with name-value pairs for  
• result - true if the requestorId is valid.  
• errors - list of errors. |

### `StateManagementScriptableApi - registerOperator()`

Register an operator for a non-workflow user.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| String   | A JSON formatted string with name-value pairs for  
• requestorId - Registered user GUID that is used to set CI Action/operational states.  
• result - true if successfully registered.  
• errors - list of errors. |

```javascript
var output = SNC.StateManagementScriptableApi.registerOperator();
var jsonUntil = new JSON();
var result = jsonUntil.decode(output);
var requestorId = result.requestorId;
```
StateManagementScriptableApi - removeBulkCiAction(String requestId, String sysIdList, String ciActionName)

Removes a CI action for a list of CIs.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestId</td>
<td>String</td>
<td>A sys_id of a workflow context, or the GUID returned from the registerOperator() method.</td>
</tr>
<tr>
<td>sysIdList</td>
<td>String</td>
<td>A comma separated list of CI sys_ids</td>
</tr>
<tr>
<td>ciActionName</td>
<td>String</td>
<td>The CI action name</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A JSON formatted string with name-value pairs for</td>
</tr>
<tr>
<td></td>
<td>• result - true if the action is removed for all CIs in the list; otherwise,</td>
</tr>
<tr>
<td></td>
<td>false.</td>
</tr>
<tr>
<td></td>
<td>• errors - list of errors.</td>
</tr>
</tbody>
</table>

StateManagementScriptableApi - setBulkCiOperationalState(String requestId, String sysIdList, String opsLabel, String opsStateListOld)

Set the operational state for list of CIs.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestId</td>
<td>String</td>
<td>A sys_id of a workflow context, or the GUID returned from the registerOperator() method.</td>
</tr>
<tr>
<td>sysIdList</td>
<td>String</td>
<td>A comma separated list of CI sys_ids</td>
</tr>
<tr>
<td>opsLabel</td>
<td>String</td>
<td>This is the string label of an operational_status choice.</td>
</tr>
<tr>
<td>opsStateListOld</td>
<td>String</td>
<td>(Optional) A comma separated list of old CI states that all CIs should be in.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
var output = SNC.StateManagementScriptableApi.registerOperator();
var jsonUtil = new JSON();
var result = jsonUtil.decode(output);
var requestorId = result.requestorId;

// list of sys_ids to update
var sys_ids;

// Set list of sys_ids's Operational State to 'Repair in Progress'
output = SNC.StateManagementScriptableApi.setBulkCIOperationalState(requestorId,
    sys_ids,'Repair in Progress');
gs.print(output);

**StateManagementScriptableApi - unregisterOperator( String requestorId)**

Unregister an operator for non-workflow users.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestorId</td>
<td>String</td>
<td>A sys_id of a workflow context, or the GUID returned from the registerOperator() method.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>A JSON formatted string with name-value pairs for</td>
</tr>
<tr>
<td></td>
<td>• result - true if successfully unregistered.</td>
</tr>
<tr>
<td></td>
<td>• errors - list of errors.</td>
</tr>
</tbody>
</table>
The StopWatch API can be used in client-side scripts using ListV2 and ListV3 APIs.

**StopWatch - getTime()**

Returns the number of milliseconds since the timer started.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of milliseconds since the timer started.</td>
</tr>
</tbody>
</table>

**StopWatch - restart()**

Resets the timer to the current time.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**StopWatch - StopWatch()**

Creates an instance of the StopWatch class.

Uses the current time as the start time.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**StopWatch - StopWatch(Date initialDate)**

Creates an instance of the StopWatch class using the specified date as the initial value.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>initialDate</td>
<td>Date</td>
<td>The initial date for the object.</td>
</tr>
</tbody>
</table>

**StopWatch - toString()**

The elapsed time as HH:MM:SS.SSS.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The elapsed time formatted as HH:MM:SS.SSS.</td>
</tr>
</tbody>
</table>

**Style - Scoped, Global**

Creates a style for defining properties such as font size, border, and alignment. You can apply the same style to multiple objects simultaneously.

This API is part of the ServiceNow PDF Generation Utilities plugin (com.snc.apppdfgenerator) and is provided within the sn_pdfgeneratorutils namespace. The plugin is activated by default.

This API is a component used with the Document API to generate a PDF.

You can apply custom styles to the following API elements:
• Cell
• Paragraph
• Table

**Style - Style()**

Instantiates a new `Style` object.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

The following examples shows how to create a `Style` object that you can add to a `Cell`, `Paragraph` or `Table` element.

```javascript
var style = new sn_pdfgeneratorutils.Style();
```

**Style – setBackgroundColor(Color color)**

Specifies a background color of an element.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>color</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

The following example shows how to set element background color. For a document usage example, see [Document API](#).

```javascript
var style = new sn_pdfgeneratorutils.Style();

var color = new sn_pdfgeneratorutils.Color([1, 0.9, 0.9]); // provided as array of RGB float values

style.setBackgroundColor(color);
```
**Style – setBold()**

Sets a style to bold font.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Parameters**

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a style to bold font.

```javascript
var style = new sn_pdfgeneratorutils.Style();
style.setBold();
```

**Style – setBorder(Number width)**

Sets the style border on all four sides of an element.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Style border width in points.</td>
</tr>
</tbody>
</table>

**Parameters**

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a style border. For a document usage example, see [Document API](#).

```javascript
var style = new sn_pdfgeneratorutils.Style();
var width = 2;
style.setBorder(width);
```
Style – setBorderBottom(Number width)
Sets the style border to the bottom of an element.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Style border width in points.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a style border to the bottom of an element. For a document usage example, see Document API.

```javascript
var style = new sn_pdfgeneratorutils.Style();
var width = 2;
style.setBorderBottom(width);
```

Style – setBorderLeft(Number width)
Sets the style border on the left side of an element.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Style border width in points.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a style border on the left side of an element. For a document usage example, see Document API.

```javascript
var style = new sn_pdfgeneratorutils.Style();
var width = 2;
```
Style – setBorderLeft(Number width)
Sets the style border on the right side of an element.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Style border width in points.</td>
</tr>
</tbody>
</table>

The following example shows how to set a style border on the right side of an element. For a document usage example, see Document API.

```javascript
var style = new sn_pdfgeneratorutils.Style();
var width = 2;
style.setBorderLeft(width);
```

Style – setBorderRight(Number width)
Sets the style border to the top of an element.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Style border width in points.</td>
</tr>
</tbody>
</table>

The following example shows how to set a style border to the top of an element. For a document usage example, see Document API.

```javascript
var style = new sn_pdfgeneratorutils.Style();
var width = 2;
style.setBorderRight(width);
```
var style = new sn_pdfgeneratorutils.Style();

var width = 2;

style.setBorderTop(width);

**Style – setColoredBorder(Number width, Color color)**

Sets style border with color on all four sides of an element.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Style border width in points.</td>
</tr>
<tr>
<td>color</td>
<td>Color</td>
<td>Style border color.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a style border with color. For a document usage example, see [Document API](#).

```javascript
var style = new sn_pdfgeneratorutils.Style();

var width = 1.0;
var borderColor = new sn_pdfgeneratorutils.Color([0.8, 0.8, 0.8]);

style.setColoredBorder(width, borderColor);
```

**Style – setColoredBorderBottom(Number width, Color color)**

Sets style border with color to the bottom side of an element.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Style border width in points.</td>
</tr>
<tr>
<td>color</td>
<td>Color</td>
<td>Style border color.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The following example shows how to set a style border to the bottom of an element. For a document usage example, see Document API.

```javascript
var style = new sn_pdfgeneratorutils.Style();
var width = 1.0;
var borderColor = new sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
style.setColoredBorderBottom(width, borderColor);
```

### Style – setColoredBorderLeft(Number width, Color color)
Sets style border with color on the left side of an element.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Style border width in points.</td>
</tr>
<tr>
<td>color</td>
<td>Color</td>
<td>Style border color.</td>
</tr>
</tbody>
</table>

The following example shows how to set a style border. For a document usage example, see Document API.

```javascript
var style = new sn_pdfgeneratorutils.Style();
var width = 1.0;
var borderColor = new sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
style.setColoredBorderLeft(width, borderColor);
```

### Style – setColoredBorderRight(Number width, Color color)
Sets style border with color on the right side of an element.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Style border width in points.</td>
</tr>
<tr>
<td>color</td>
<td>Color</td>
<td>Style border color.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a style border with color. For a document usage example, see Document API.

```javascript
var style = new sn_pdfgeneratorutils.Style();

var width = 1.0;
var borderColor = new sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);

style.setColoredBorderRight(width, borderColor);
```

**Style – setColoredBorderTop(Number width, Color color)**

Sets style border with color to the top side of an element.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Style border width in points.</td>
</tr>
<tr>
<td>color</td>
<td>Color</td>
<td>Style border color.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a style border. For a document usage example, see Document API.

```javascript
var style = new sn_pdfgeneratorutils.Style();
```
var width = 1.0;
var borderColor = new sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
style.setColoredBorderTop(width, borderColor);

**Style – setFontColor(Color color)**

Sets a font color.

**Parameters**

| Name   | Type    | Description 
|--------|---------|-------------
| color  | Color   | Font color. |

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set a font color. For a document usage example, see [Document API](#).

```javascript
var style = new sn_pdfgeneratorutils.Style();
var fontColor = new sn_pdfgeneratorutils.Color([1,0.5,0.5]);
style.setFontColor(fontColor);
```

**Style – setFontSize(Number fontSize)**

Sets a style font size.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fontSize</td>
<td>Number</td>
<td>Font size in points.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
The following example shows how to set a font size.

```javascript
var style = new sn_pdfgeneratorutils.Style();
style.setFontSize(12);
```

**Style – setHorizontalAlignment(String alignment)**

Sets the horizontal alignment for a style.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alignment</td>
<td>String</td>
<td>Horizontal alignment setting. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• center: Align contents to the center.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• left: Align contents to the left.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• right: Align contents to the right.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set horizontal alignment on an element. For a document usage example, see Document API.

```javascript
var style = new sn_pdfgeneratorutils.Style();
var alignment = "Center";
style.setHorizontalAlignment(alignment);
```

**Style – setItalic()**

Sets a style to italic font.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The following example shows how to set an element style to an italic font.

```javascript
var style = new sn_pdfgeneratorutils.Style();
style.setItalic();
```

**Style – setPadding(Number padding)**

Sets the padding of all four sides of an element to the same width.

See also:

- `setPaddingBottom()`
- `setPaddingLeft()`
- `setPaddingRight()`
- `setPaddingTop()`

```
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>padding</td>
<td>Number</td>
<td>Padding width in points as a decimal value.</td>
</tr>
</tbody>
</table>
```

```
var style = new sn_pdfgeneratorutils.Style();
var padding = 2.5;
style.setPadding(padding);
```

**Style – setPaddingBottom(Number padding)**

Sets the value of the bottom padding width of an element.

The following example shows how to set bottom style to 2.5 points. For a document usage example, see Document API.

```
var style = new sn_pdfgeneratorutils.Style();
var padding = 2.5;
style.setPaddingBottom(padding);
```
See also:

- `setPadding()`
- `setPaddingLeft()`
- `setPaddingRight()`
- `setPaddingTop()`

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>padding</td>
<td>Number</td>
<td>Padding width in points as a decimal value.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set bottom element padding to 2.5 points. For a document usage example, see [Document API](#).

```javascript
var style = new sn_pdfgeneratorutils.Style();
var padding = 2.5;
style.setPaddingBottom(padding);
```

### Style – `setPaddingLeft(Number padding)`

Sets the value of the left padding width of an element.

See also:

- `setPadding()`
- `setPaddingBottom()`
- `setPaddingRight()`
- `setPaddingTop()`

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>padding</td>
<td>Number</td>
<td>Padding width in points as a decimal value.</td>
</tr>
</tbody>
</table>
The following example shows how to set left element padding to 2.5 points. For a document usage example, see Document API.

```javascript
var style = new sn_pdfgeneratorutils.Style();
var padding = 2.5;
style.setPaddingLeft(padding);
```

**Style – setPaddingRight(Number padding)**

Sets the value of the right padding width of a style.

See also:
- `setPadding()`
- `setPaddingBottom()`
- `setPaddingLeft()`
- `setPaddingTop()`

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>padding</td>
<td>Number</td>
<td>Padding width in points as a decimal value.</td>
</tr>
</tbody>
</table>

The following example shows how to set right element padding to 2.5 points. For a document usage example, see Document API.

```javascript
var style = new sn_pdfgeneratorutils.Style();
var padding = 2.5;
```
**Style – setPaddingRight(Number padding)**

Sets the value of the right padding width of an element.

See also:
- `setPadding()`
- `setPaddingBottom()`
- `setPaddingLeft()`
- `setPaddingRight()`

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>padding</td>
<td>Number</td>
<td>Padding width in points as a decimal value.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set right element padding to 2.5 points. For a document usage example, see Document API.

```javascript
var style = new sn_pdfgeneratorutils.Style();
var padding = 2.5;
style.setPaddingRight(padding);
```

**Style – setTextAlignment(String alignment)**

Sets the text alignment for a style.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alignment</td>
<td>String</td>
<td>Text alignment position. Valid values:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>• text-center: Aligns text to the center.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• text-justified: Modifies the space between characters to completely fill text between the left and right sides. The final line is left-aligned.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• text-justified-all: Justifies text alignment including the final line.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• text-left: Align text to the left.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• text-right: Align text to the right.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the element text to center alignment.

```javascript
var style = new sn_pdfgeneratorutils.Style();
String alignment = "text-center";
style.setTextAlignment(alignment);
```

### Style – setVerticalAlignment(String alignment)

Sets the vertical alignment for this element.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alignment</td>
<td>String</td>
<td>Vertical alignment setting. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• bottom: Aligns contents to the bottom.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• mid: Aligns contents to the center.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• top: Aligns contents to the top.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The following example shows how to set vertical alignment on an element. For a document usage example, see Document API.

```javascript
var style = new sn_pdfgeneratorutils.Style();
var alignment = "Mid";
style.setVerticalAlignment(alignment);
```

### Stream - Scoped, Global

The Stream API interacts with a stream of items such as records. For example, you can use the `forEach()` method to update the state of each record in a stream returned by the GlideQuery API.

You can get a Stream object in these ways:

- Instantiate a Stream object using the constructor.
- Return a Stream object from the `GlideQuery.select()` method. For more information, see GlideQuery.

This method is static and does not require an instance of the class: `fromArray()`.

Use the Stream API in scoped or global server-side scripts. This API requires the GlideQuery [com.sn_glidequery] plugin.

### Implementation

This API can work with the GlideQuery and Optional APIs in a builder pattern where the method calls chain together, each method building on the returned result of the previous method. Use methods to define the attributes of the query. The methods do not execute until you call a terminal method, a method that returns a query result, allowing you to define the requirements of the query before executing it.

If the query returns a single record, the system wraps the result in an Optional object. If the query returns a stream of records, the system wraps the result in a Stream object. These objects let you manage the result using a set of methods in each API.
For example, here's a script that performs a query on the Task table and groups the records by priority and returns groups with total reassignments greater than four.

```javascript
var query = new global.GlideQuery('task')
    .where('active', true) //Returns new GlideQuery object with a "where" clause.
    .groupBy('priority') //Returns new GlideQuery object with a "group by" clause.
    .aggregate('sum', 'reassignment_count') //Returns new GlideQuery object with a "sum(reassignment_count)" clause.
    .having('sum', 'reassignment_count', '>', 4) //Returns new GlideQuery object with a "having reassignment_count > 4" clause.
    .select() //Returns a stream of records wrapped in a Stream object.
    .toArray(10); //Terminal method in the Stream class that executes the query and returns the result.
```

**Terminal methods**

For performance reasons, a query only fetches data when you call a terminal method. These are the terminal methods from the `Stream` class:

- `every()`
- `find()`
- `forEach()`
- `reduce()`
- `some()`
- `toArray()`

**Stream - Stream(Function nextFn)**

Instantiates a Stream object.

Instead of using this constructor, you can return a Stream object based on a query using the `GlideQuery` API.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nextFn</td>
<td>Function</td>
<td>A function that retrieves the next item in the stream.</td>
</tr>
</tbody>
</table>

This example shows you how to create a Stream object based on a random number generator. Make sure to include the `limit()` method to avoid creating an infinite loop.
new Stream(Math.random)
  .map(Math.round)
  .map(function (num) {
    return num === 1 ? 'heads' : 'tails';
  })
  .limit(10)
  .forEach(gs.info)

Output:

*** Script: tails
*** Script: heads
*** Script: tails
*** Script: heads
*** Script: tails
*** Script: heads
*** Script: tails
*** Script: tails
*** Script: tails

Stream - chunk(Number count)

Returns results in batches of arrays, each containing the number of records passed to the method.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>Number</td>
<td>Number of records in each array returned from the stream.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stream</td>
<td>Object used to interact with a stream of items such as records.</td>
</tr>
</tbody>
</table>

This example shows how to query a table and chunk the result into batches of arrays.

```javascript
var chunkResult = new GlideQuery('cmdb_ci_hardware')
  .select('asset', 'purchase_date')
  .limit(10)
```
```javascript
.chunk(5) // returns arrays of 5 sys_ids at a time
.toArray(100);

gs.info(JSON.stringify(chunkResult));

Output:

```

```json
[
  {
    "asset": "03clba8837f3100044e0bfc8bcbe5da8",
    "purchase_date": null,
    "sys_id": "10884798c61122750108b095e21e4080"
  },
  {
    "asset": "8fclba8837f3100044e0bfc8bcbe5da9",
    "purchase_date": null,
    "sys_id": "108a9205c61122750786e160f9d343e"
  },
  {
    "asset": "dfc1fa8837f3100044e0bfc8bcbe5d20",
    "purchase_date": null,
    "sys_id": "55b35562c0a8010e01cflf22378e0ae9"
  },
  {
    "asset": "9fclfa8837f3100044e0bfc8bcbe5d22",
    "purchase_date": null,
    "sys_id": "55b37e80c0a8010e00028a1d14e2d7"
  },
  {
    "asset": "d3clfa8837f3100044e0bfc8bcbe5d2e",
    "purchase_date": null,
    "sys_id": "60cc6f8cc0a8010e0016f4f6e55beea"
  }
],
```

```json
[
  {
    "asset": null,
    "purchase_date": null,
    "sys_id": "304481257f701200bee45f19befa915b"
  },
  {
    "asset": null,
    "purchase_date": null,
```

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
This example shows how to create a child query using batches of IDs. When you call the `flatMap()` method after using the `chunk()` method, the system iterates over the batch of records instead of each individual record.

```javascript
var chunkResult = new global.GlideQuery('cmdb_ci_hardware')
    .select('sys_id')
    .map(function (device) { return device.sys_id; })
    .chunk(5) // returns arrays of 5 sys_ids at a time
    .flatMap(function (deviceIds) {
        return new GlideQuery('cmdb_sam_sw_install')
            .where('installed_on', 'IN', deviceIds)
            .select('software_model', 'installed_on');
    });

gs.info(JSON.stringify(chunkResult));
```

**Stream - every(Function predicate)**

Applies a predicate function to every item in the Stream object. If the predicate returns true for every item in the stream, the method returns true. If the predicate returns false for any item in the stream, the method returns false.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>predicate</td>
<td>Function</td>
<td>Predicate function to apply to every record or item inside the Stream object. The function must take each item in the stream as input and return a boolean.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the predicate function returns true for every item in the stream.</td>
</tr>
</tbody>
</table>

Valid values:
- **true**: The predicate function returns true for every item in the stream.
- **false**: The predicate function does not return true for every item in the stream.

This example shows how to apply a predicate function to every item in a stream.

```javascript
var hasOnlyShortDescriptions = new global.GlideQuery('task')
  .whereNotNull('description')
  .select('description')
  .every(function (t) {
    return t.description.length < 10;
  });

gs.info(hasOnlyShortDescriptions);
```

**Output:**

```
false
```

### Stream - filter(Function predicate)

Applies a predicate function to each item in the Stream object. If the predicate returns true, the method returns the stream. If the predicate returns false, it returns an empty Stream object.

For better performance, use the `where()`, `whereNotNull()`, and `whereNull()` methods in the `GlideQuery` class instead of this method where possible. See `GlideQuery`. 

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th><strong>Parameters</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>predicate</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Returns</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>Stream</td>
</tr>
</tbody>
</table>

This example shows how to check all records in the Task table against a defined filter. If the records match the filter, the system returns the stream of records. Otherwise, it returns an empty Stream object.

```javascript
var shoutingTasks = new global.GlideQuery('task')
  .whereNotNull('description')
  .select('description')
  .filter(function (task) {
    return task.description.toUpperCase() === task.description;
  });
```

**Stream - find(Function predicate)**

Returns the first record or item in the Stream object that matches the predicate function. If no predicate function is provided, then the method returns the first record or item in the Stream.

<table>
<thead>
<tr>
<th><strong>Parameters</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>predicate</td>
</tr>
</tbody>
</table>
This example shows how to return the first record from the stream.

```javascript
var UserOptional = new global.GlideQuery('sys_user')
  .where('active', true)
  .where('company.name', 'ServiceNow')
  .select()
  .find()
  .map(function (user) {
    return JSON.stringify(user);
  })

gs.info(UserOptional);

Output:

Optional<"sys_id":"babb4639b76233004fbc2089ee11a97f">"}
This example shows you how to query the User table and then create a child query from the result. This example executes an N+1 query, which can cause performance issues. Avoid this use case in a production environment.

```javascript
var records = new global.GlideQuery('sys_user')
  .where('last_login', '>', '2015-12-31')
  .select('first_name', 'last_name')
  .flatMap(function (u) {
    return new global.GlideQuery('task')
      .where('closed_by', u.sys_id)
      .select('closed_at', 'description')
      .map(function (t) {
        return {
          first_name: u.first_name,
          last_name: u.last_name,
          description: t.description,
          closed_at: t.closed_at
        };
      });
  });
  .limit(5)
  .toArray(100);

gs.info(JSON.stringify(records));
```

Output:

```
[
  {
    "first_name":"System",
    "last_name":"Administrator",
    "description":null,
    "closed_at":"2020-08-23 13:14:56"
  },
  {
    "first_name":"System",
    "last_name":"Administrator",
    "description":null,
    "closed_at":"2020-08-23 13:07:43"
  },
  {
    "first_name":"System",
    "last_name":"Administrator",
    "description":null,
    "closed_at":"2020-06-15 06:59:05"
  }
]```
Stream - forEach(Function fn)

Applies the specified function to each record or item in the stream.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fn</td>
<td>Function</td>
<td>Function to apply to each item in the stream.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows you how to print the result of each item in the stream.

```javascript
var firstNames = new global.GlideQuery('sys_user')
  .select('first_name')
  .forEach(function (u) {
    gs.debug(u.first_name);
  });
```

Output:

### Script: [DEBUG] survey
### Script: [DEBUG] Lucius
### Script: [DEBUG] Jimmie
### Script: [DEBUG] Melinda
### Script: [DEBUG] Jewel
Stream - fromArray(Object arr)

Returns a Stream object that contains the values from the provided array.

ℹ️ Note: This method is static. You do not need an instance of the class to use this method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>arr</td>
<td>Array</td>
<td>Array of values to create the stream from.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stream</td>
<td>Object used to interact with a stream of items such as records.</td>
</tr>
</tbody>
</table>

This example shows how to create a Stream object containing an array of values.

```javascript
var nameStream = Stream.fromArray(['Bob', 'Sue', 'Sam']
  .map(function (name) {
    return name.toUpperCase();
  })
  .toArray(3);

gs.info(JSON.stringify(nameStream));
```

Output:

```
["BOB","SUE","SAM"]
```

Stream - limit(Number count)

Limits the number of results returned by the stream.
For better performance, use the `limit()` method in the `GlideQuery` class where possible. See `GlideQuery`. You may need to use this method to limit results with the `Stream.flatMap()` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>Number</td>
<td>Number of records to return.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stream</td>
<td>Object used to interact with a stream of items such as records.</td>
</tr>
</tbody>
</table>

This example shows you how to limit the results returned from the `Stream.flatMap()` method.

```javascript
var records = new global.GlideQuery('sys_user')
    .where('last_login', '>', '2015-12-31')
    .select('first_name', 'last_name')
    .flatMap(function (u) {
        return new GlideQuery('task')
            .where('closed_by', u.sys_id)
            .select('closed_at', 'description')
            .map(function (t) {
                return {
                    first_name: u.first_name,
                    last_name: u.last_name,
                    description: t.description,
                    closed_at: t.closed_at
                };
            });
    })
    .limit(5)
    .toArray(100);

gs.info(JSON.stringify(records));
```

Output:

```json
[
]
```
Stream - map(Function fn)

Applies a function to each item in a stream and returns the updated Stream object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fn</td>
<td>Function</td>
<td>Function to apply to the result of the query that takes the each item in the stream as input.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stream</td>
<td>Object containing the stream of records updated after applying the function.</td>
</tr>
</tbody>
</table>

This example shows you how to apply a function to every item in the stream.

```javascript
var users = new global.GlideQuery('sys_user')
    .whereNotNull('first_name')
    .select('first_name')
    .map(function (u) {
        return u.first_name.toUpperCase();
    })
    .toArray(100);

gs.info(JSON.stringify(users));
```

Output:

```
[
    "SURVEY",
    "LUCIUS",
    "JIMMIE",
    "MELINDA",
    "JEWEL",
    "SEAN",
    "JACINTO",
    "KRYSLE",
    "BILLIE",
    "CHRISTIAN",
    ...
]
```

**Stream - reduce(function reducerFn, Any initialValue)**

Executes a reducer function on each item in the stream, resulting in single output value.

This method is similar to the native JavaScript `reduce()` method. For more information, see [w3schools documentation](https://www.w3schools.com/jsref/jsref_reduce.asp).
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| reducerFn | Function | Function to apply to each item in the stream that reduces the stream to a single value. This function must take two arguments:  
  - acc: Accumulator that accumulates all values returned by the function.  
  - cur: Current item being accumulated in the array. |
| initialValue | Any   | Value passed to the function as the initial value. |

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any</td>
<td>Accumulated total of all items returned by the reducer function.</td>
</tr>
</tbody>
</table>

This example shows you how to return the record with the longest name from the User table.

```javascript
var longestName = new global.GlideQuery('sys_user')
  .whereNotNull('first_name')
  .select('first_name')
  .reduce(function (acc, cur) {
    return cur.first_name.length > acc.length
    ? cur.first_name
    : acc;
  }, '');

gs.info(JSON.stringify(longestName));
```

**Output:**

"ATF_TestItilUser1"

### Stream - some(Function predicate)

Applies a predicate function, a function that takes a single value and returns true or false, to each item in the stream. If the predicate returns true for any item in the stream, the method returns true.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>predicate</td>
<td>Function</td>
<td>Predicate function to apply to the items inside the Stream object. Must return a Boolean value.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the predicate function returned true for an item in the stream. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The predicate function returned true for an item in the stream.</td>
</tr>
<tr>
<td></td>
<td>• false: The predicate function did not return true for an item in the stream.</td>
</tr>
</tbody>
</table>

This example shows how to check whether any descriptions in the Task table are over 1,000 characters long.

```javascript
var hasLongDescriptions = new global.GlideQuery('task')
    .whereNotNull('description')
    .select('description')
    .some(function (t) {
        return t.description.length > 1000;
    });

gs.info(hasLongDescriptions);
```

Output:

false

### Stream - toArray(Number count)

Returns an array containing the given number of items from the stream.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>Number</td>
<td>The maximum number of items from the stream to return in the array.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array containing the given number of items from the stream.</td>
</tr>
</tbody>
</table>

This example shows you how to transform a stream of records into a JavaScript array.

```javascript
var users = new global.GlideQuery('sys_user')
  .limit(10)
  .select('first_name', 'last_name')
  .toArray(50);

gs.info(JSON.stringify(users));
```

Output:

```json
[
  {
    "first_name": "Jewel",
    "last_name": "Agresta",
    "sys_id": "02826bf03710200044e0bfc8bcbe5d64"
  },
  {
    "first_name": "Sean",
    "last_name": "Bonnet",
    "sys_id": "02826bf03710200044e0bfc8bcbe5d6d"
  },
  {
    "first_name": "Jacinto",
    "last_name": "Gawron",
    "sys_id": "02826bf03710200044e0bfc8bcbe5d76"
  },
  {
    "first_name": "Krystle",
    "last_name": "Stika",
```
Subflow - Scoped

Runs published Flow Designer subflows.

This API is deprecated and replaced by the FlowAPI - Scoped, GlobalFlowAPI - Scoped, Global.

The Subflow API can only be used in server scripts.

Use the sn_fd namespace to access the Subflow API.

Before interacting with a subflow using the Subflow API, you must first create and publish the subflow in the Flow Designer interface. Because the Subflow API only interacts with pre-built subflows, there is no constructor for the class.

Note: To optimize instance performance, avoid calling these methods from a script for an asynchronous business rule. Instead, create a scheduled job record within the Flow Designer UI.

Scoped Subflow - startAsync(String scopeName.subflowName, Map inputs)

Runs a published subflow asynchronously.

Asynchronous calls are non-blocking, allowing the client to execute other code in the script without having to wait for the subflow to complete.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopeName.subflowName</td>
<td>String</td>
<td>The application scope for the subflow and the internal name of the subflow to run. If</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scopeName</td>
<td>String</td>
<td>If scopeName is not included, the scope of the user currently logged in is used. Retrieve the internal name of the subflow using the Internal name column on the Flow Designer landing page.</td>
</tr>
<tr>
<td>inputs</td>
<td>Map</td>
<td>Name-value pairs that define subflow inputs. If a subflow includes mandatory inputs, they must be included. For inputs of Reference or Document ID field types, use a GlideRecord object as the value.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>PlanResponse object containing the following properties:</td>
</tr>
<tr>
<td></td>
<td>• contextId: Sys_id of the execution details record for the executed subflow. Access the execution details by navigating to the Flow Executions tab in Flow Designer and filtering by sys_id.</td>
</tr>
<tr>
<td></td>
<td>An exception occurs when the subflow:</td>
</tr>
<tr>
<td></td>
<td>• Does not exist within the specified application scope, or the subflow or scope name has been misspelled.</td>
</tr>
<tr>
<td></td>
<td>• Is not published.</td>
</tr>
<tr>
<td></td>
<td>• Is passed an input object that does not match the subflow inputs.</td>
</tr>
<tr>
<td></td>
<td>• Exceeds the recursion limit set by the com.glide.hub.flow_engine.indirect_recursion_limit system property. The default value is three.</td>
</tr>
</tbody>
</table>

```javascript
//Run a subflow that takes two inputs: user, a sys_user record, and laptop_welcome_message, a string.

(function startSubflowAsync() {
    try {
        var userToProvisionFor = new GlideRecord('sys_user');
        userToProvisionFor.get('62826bf03710200044e0bfc8bcbe5df1');
    }
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
var inputs = {};
inputs['user'] = userToProvisionFor;
inputs['laptop_welcome_message'] = 'Welcome Onboard!!';

var result = sn_fd.Subflow.startAsync('sn_devstudio.provisionlaptop', inputs);

//The Sys ID of a flow execution (contextId)
var contextId = result.contextId;

}) catch (ex) {
    var message = ex.getMessage();
    gs.error(message);
}

})();

SummaryTableWriter - Global
The SummaryTableWriter script include enables the creation of summary sets for reports that can be generated with standard queries and trending.

This script provides methods that enable you to customize the output of the summary sets for reports, such as adding a title, defining a query on which to base the report data, and specifying the stack column and X axis.

SummaryTableWriter - generate()
Creates the summary set of records for reports.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

var opened = new SummaryTableWriter('incident', '');
opened.generate();
SummaryTableWriter - setQuery(String query)
Defines the query to use to load the data for the summary record set.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>String</td>
<td>Query to use to load the summary data.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var summarySet = new SummaryTableWriter('incident', 'category');
summarySet.setQuery("active=true");
summarySet.generate();
```

SummaryTableWriter - setStackOn(String column)
Defines the column on which to stack the incident records.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>column</td>
<td>String</td>
<td>Column that is used to stack the incident records.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var summarySet = new SummaryTableWriter('incident', 'category');
summarySet.setQuery("active=true");
summarySet.setStackOn('priority');
summarySet.generate();
```

SummaryTableWriter - setTitle(String title)
Sets the title of the summary set, which is also the chart title.

```javascript
var summarySet = new SummaryTableWriter('incident', 'category');
summarySet.setQuery("active=true");
summarySet.setTitle('Summary Title');
summarySet.generate();
```
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>String</td>
<td>Title given to the summary set that is used as the chart title. Also used as the key field for chart generators to determine whether the data was generated.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var opened = new SummaryTableWriter('incident', '');
opened.setTitle("Trend of Open Incidents");
opened.setTrend('opened_at', 'month');
opened.generate();
```

### SummaryTableWriter - setTrend(String field, String interval)

Sets the time field used for the X axis or stacking field; can also set the time interval used to calculate the trend.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Time field that is used for X axis values if grouping is not used. If grouping is used, then this is the stacking field.</td>
</tr>
<tr>
<td>interval</td>
<td>String</td>
<td>Time interval that is used to calculate the trend.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var opened = new SummaryTableWriter('incident', '');
opened.setTitle("Trend of Open Incidents");
opened.setTrend('opened_at', 'month');
opened.generate();
```
SummaryTableWriter - SummaryTableWriter(String tableName, String columnName)

Produces a summary set for the COUNT of all incidents grouped by category.

The custom chart definition table contains a field named Table that is hidden by default. If the field is set, which it is in some of the base custom charts, then, no matter what is defined within the custom chart scripts, clicking on the chart points you to this table.

For example, if you create a custom chart against the [task] table through a script, but you copied the custom chart from a custom chart that contains Table = incident, then clicking on any bar in the custom script will go to incident.do rather than to task.do.

To check the value of the Table field, click the additional actions icon and select Configure > Form Layout and move the Table field from Available to Selected. Open the custom chart definition and set the Table field to none.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>tableName</td>
</tr>
<tr>
<td>columnName</td>
</tr>
</tbody>
</table>

```javascript
var summarySet = new SummaryTableWriter('incident', 'category');
summarySet.generate();
```

SVGToPDFConversionAPI API - Scoped, Global

Enables adding an SVG to a PDF and converting an SVG to PDF.

This API is part of the ServiceNow PDF Generation Utilities plugin (com.snc.apppdfgenerator) and is provided within the sn_pdfgeneratorutils namespace. The plugin is activated by default.

See also PDFGenerationAPI.

SVGToPDFConversionAPI - SVGToPDFConversionAPI()

Instantiates a new SVGToPDFConversionAPI object.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>
The following examples shows how to create a SVGToPDFConversionAPI object.

```javascript
var v = new sn_pdfgeneratorutils.SVGToPDFConversionAPI();
```

**SVGToPDFConversionAPI - addSVGToPDF(String svg, String inputPdfSysId, String targetTable, String targetSysId, String nameForPdf, Number pageNo, Number x, Number y, Number svgImgWidth, Number svgImgHeight)**

Adds an SVG image to a PDF on a specified page with image size and location coordinates.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>svg</td>
<td>String</td>
<td>SVG to convert to a PDF document.</td>
</tr>
<tr>
<td>inputPdfSysId</td>
<td>String</td>
<td>Sys_id of the target PDF in which to create a copy of with an SVG. The target PDF is in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>targetTable</td>
<td>String</td>
<td>Name of the table containing the record to which the PDF is attached. You can find this value in the same row as the attachment listed in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>targetTableSysId</td>
<td>String</td>
<td>Sys_id of the record to which the PDF is attached. You can find this value in the same row as the attachment listed in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>targetTableSysId</td>
<td>String</td>
<td>Name of the table on which to attach the converted PDF.</td>
</tr>
<tr>
<td>pdfName</td>
<td>String</td>
<td>Name to give the PDF. Default: Sys_id of the PDF in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>pageNo</td>
<td>Number</td>
<td>Page number in the attached PDF on which to place the SVG.</td>
</tr>
<tr>
<td>x</td>
<td>Number</td>
<td>Horizontal position of the SVG in the page. Value is in points.</td>
</tr>
<tr>
<td>y</td>
<td>Number</td>
<td>Vertical position of the SVG in the page. Value is in points.</td>
</tr>
</tbody>
</table>
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>svgImgWidth</td>
<td>Number</td>
<td>Width of the SVG image in pixels.</td>
</tr>
<tr>
<td>svgImgHeight</td>
<td>Number</td>
<td>Height of the SVG image in pixels.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing sys_id of a new PDF attachment with inserted SVG if successful, error message otherwise.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;attachment_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

<Object>.attachment_id | If SVG conversion is successful, sys_id of the attached PDF. The file is listed in the Attachments [sys_attachment] table. |
| Data type: String      |                                                                                                                                               |

<Object>.message | Message confirming success or error. Possible values:                                                                                     |
|                  | - Adding SVG content to PDF failed.                                                                                                          |
|                  | - Adding SVG to PDF and creating new PDF is successful.                                                                                     |
|                  | - Given target record [<tableName> - <targetTableSysId>] does not exist. – Target table sys_id is not in the table provided. Make sure you include the correct table name for the record. |
|                  | - Input PDF attachment sys id to add the SVG content is null                                                                               |
|                  | - Input SVG content is empty or null                                                                                                         |
|                  | - PdfReader is not opened with owner password                                                                                               |
|                  | - The target table name to attach the modified PDF : <targetTable> is empty or null                                                          |
|                  | - The target table name to attach the modified PDF : <targetTable> is not valid                                                             |
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>&lt;Object&gt;.status</td>
<td>Status indicating whether the operation is successful.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• success - Operation was successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – Operation was not successful. The <strong>message</strong> provides details.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how get SVG from the description field of an incident record and add it to page 5 of a PDF attachment. The new PDF with SVG is located in the Attachments [sys_attachment] table.

```javascript
var gr = new GlideRecord("incident");

var svg;

if (gr.get("<tableSysId>")) {
    svg = gr.description.toString();
}

var result = new sn_pdfgeneratorutils.SVGToPDFConversionAPI().addSVGToPDF(svg,
    "<inputPdfSysId>", "incident", "<tableSysId>", "AddSVGToPDF", 5, 30, 30, 300, 100);

gs.info(JSON.stringify(result));
```

**Output:**

```json
{"attachment_id": "<sys_id>", "message": "Adding SVG to PDF and creating new PDF is successful.", "status": "success"}
```

**SVGToPDFConversionAPI - convertSVGToPDF(String svg, String pdfName, String targetTable, String targetTableSysId)**

Converts an SVG provided as string to PDF.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>svg</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pdfName</td>
<td>String</td>
<td>Name to give the PDF. Default: Sys_id of the PDF in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>targetTable</td>
<td>String</td>
<td>Name of the table on which to attach the converted PDF.</td>
</tr>
<tr>
<td>targetTableSys</td>
<td>String</td>
<td>Sys_id of the record on which to attach the converted PDF.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing sys_id of the PDF attachment converted from SVG if successful, error message otherwise.</td>
</tr>
<tr>
<td>&lt;Object&gt;.attachment_id</td>
<td>If SVG conversion is successful, sys_id of the attached PDF. The file is listed in the Attachments [sys_attachment] table.</td>
</tr>
</tbody>
</table>
| <Object>.message | Message confirming success or error. Possible values:  
  * Converting SVG image to PDF document failed.  
  * Exception while reading Source document contents.  
  * Given target record [<tableName> - <targetTableSysId>] does not exist. – Target table sys_id is not in the table provided. Make sure you include the correct table name for the record.  
  * SVG to PDF conversion is successful. |
## Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• The Target table name - <code>&lt;tableName&gt;</code> to attach the converted SVG is not valid</td>
</tr>
<tr>
<td></td>
<td>• The Target table name - <code>&lt;tableName&gt;</code> to attach the converted SVG is empty or null</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

| <Object>.status            | Status indicating whether the operation is successful. Possible values:     |
|                           | • success - Operation was successful.                                       |
|                           | • failure – Operation was not successful. The `message` provides details.   |
|                           | Data type: String                                                           |

The following example shows how get SVG from the description field of an incident record, to convert the SVG to PDF and attach it to a record.

```javascript
var gr = new GlideRecord("incident");

var svg;

if (gr.get("<tableSysId>")) {
    svg = gr.description.toString();
}

var v = new sn_pdfgeneratorutils.SVGToPDFConversionAPI();

var result = v.convertSVGToPDF(svg, "ConvertSVGToPDF", "incident", "<targetTableSysId>");
gs.info(JSON.stringify(result));
```

**Output:**

```json
{"attachment_id":"<pdf_attachment_sys_id>","message":"SVG to PDF conversion is successful.","status":"success"}
```
SVGToPDFConversionAPI - convertSVGToPDFWithSize(String svg, String nameForPDF, String targetTable, String targetTableSysId, Number svgImgWidth, Number svgImgHeight)

Converts an SVG provided as string to PDF with image width and height values.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>svg</td>
<td>String</td>
<td>SVG to convert to a PDF document.</td>
</tr>
<tr>
<td>pdfName</td>
<td>String</td>
<td>Name to give the PDF. Default: Sys_id of the PDF in the Attachments [sys_attachment] table.</td>
</tr>
<tr>
<td>targetTable</td>
<td>String</td>
<td>Name of the table on which to attach the converted PDF.</td>
</tr>
<tr>
<td>targetTableSysId</td>
<td>String</td>
<td>Sys_id of the record on which to attach the converted PDF.</td>
</tr>
<tr>
<td>svgImgWidth</td>
<td>Number</td>
<td>Width of the SVG image in pixels.</td>
</tr>
<tr>
<td>svgImgHeight</td>
<td>Number</td>
<td>Height of the SVG image in pixels.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Object containing sys_id of the PDF attachment converted from SVG if successful, error message otherwise.</td>
</tr>
<tr>
<td>&lt;Object&gt;.attachment_id</td>
<td>If SVG conversion is successful, sys_id of the attached PDF. The file is listed in the Attachments [sys_attachment] table. Data type: String</td>
</tr>
<tr>
<td>&lt;Object&gt;.message</td>
<td>Message confirming success or error. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Converting SVG image to PDF document failed.</td>
</tr>
<tr>
<td></td>
<td>• Exception while reading Source document contents.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Given target record [&lt;tableName&gt; - &lt;targetTableSysId&gt;] does not exist. – Target table sys_id is not in the table provided. Make sure you include the correct table name for the record.</td>
</tr>
<tr>
<td></td>
<td>• The Target table name - &lt;tableName&gt; to attach the converted SVG is not valid</td>
</tr>
<tr>
<td></td>
<td>• The Target table name - &lt;tableName&gt; to attach the converted SVG is empty or null</td>
</tr>
<tr>
<td></td>
<td>• SVG to PDF conversion is successful.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

<Object>.status  Status indicating whether the operation is successful. Possible values:  
• success - Operation was successful.  
• failure – Operation was not successful. The message provides details.     
Data type: String

The following example shows how get SVG from the description field of an incident record, to convert the SVG to PDF, and attach it to a record.

```javascript
var gr = new GlideRecord("incident");

var svg;

if (gr.get("<tableSysId>")) {
    svg = gr.description.toString();
}

var v = new sn_pdfgeneratorutils.SVGToPDFConversionAPI();

var result = v.convertSVGToPDFWithSize(svg, "ConvertSVGToPDF", "incident", "<targetTableSysId>", 600, 150);
gs.info(JSON.stringify(result));
```

Output:
SystemDocument - Scoped, Global


This API requires the Document Management plugin (com.snc.platform_document_management) and is provided within the sn_doc_services namespace. For information, see Document Services.

To create a document, use the DocumentService API.

SystemDocument - SystemDocument(String docName)

Instantiates a SystemDocument object.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>docName</td>
<td>String</td>
<td>Name of the document.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to instantiate a SystemDocument object and set several fields for a document record. See also DocumentService API.

```javascript
var d = new sn_doc_services.SystemDocument('My document');

// Define the document fields
var reviewers = '62826bf03710200044e0bfc8bcbe5df1,a8f98bb0eb32010045e1a511526fe3a';
d.description('description');
d.classification('restricted');
d.state('review');
d.department('93b25282c0a8000b0b55c8ab34e2f1e6');
d.template(false);
d.type('policy');
d.reviewers(reviewers);
d.audience('external');
```
var s = new sn_doc_services.DocumentService();
gs.info(JSON.stringify(s.createDocument(d), null, 2));

Output:
{
    "message": "Create document sysId : 1040420224503410f877a6fed1c2b031 is successful.",
    "request_id": "1040420224503410f877a6fed1c2b031",
    "status": "success"
}

SystemDocument - audience(String audience)
Sets the Audience field of a document record to specify external or internal availability. Represents the Audience field in a Document Management form.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>audience</td>
<td>String</td>
<td>String representing the intended audience of the document. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• external</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• internal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: internal</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the Audience field in a new document record. See also DocumentService - Scoped, Global.

var d = new sn_doc_services.SystemDocument('My document');
var s = new sn_doc_services.DocumentService();
d.audience('external');
gs.info(JSON.stringify(s.createDocument(d), null, 2));

Output:
SystemDocument - classification(String classification)

Sets the Classification field of a document record to identify the level of information sensitivity. Represents the Classification field in a Document Management form.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>classification</td>
<td>String</td>
<td>Classification of the document. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• confidential</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• public</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• restricted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: public</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The value of the classification property of a SystemDocument object.</td>
</tr>
</tbody>
</table>

The following example shows how to set the classification field in a new document record. See also DocumentService API.

```javascript
var d = new sn_doc_services.SystemDocument('My restricted document');
var s = new sn_doc_services.DocumentService();
d.classification('restricted');
gs.info(JSON.stringify(s.createDocument(d), null, 2));
```

Output:

```
{
  "message": "Create document sysId : d80435c624103410f877a6fed1c2b0d0 is successful."
}
SystemDocument - `createdFromTemplate(String docTemplate)`
Sets the template from which to generate a document. When the document is generated, the specified template is listed in the Created From Template field in the Documents [ds_document_list] table.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>docTemplateName</td>
<td>String</td>
<td>Sys_id of a document listed in the Documents [ds_document_list] table. The document must have the Template checkbox selected (true). See also DocumentVersionService API.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to create a new document record from an existing template. See also DocumentService API.

```javascript
var d = new sn_doc_services.SystemDocument('New doc from template');
var s = new sn_doc_services.DocumentService();
d.createdFromTemplate('<doc_template_sys_id>');
gs.info(JSON.stringify(s.createDocument(d), null, 2));
```

Output:

```
{
  "message": "Create document sysId: deaef1ce24103410f877a6fed1c2b085 is successful.",
  "request_id": "deaef1ce24103410f877a6fed1c2b085",
  "status": "success"
}
```

SystemDocument - `department(String docDept)`
Sets the Department field of a document record.
The following example shows how to set the department field in a new document record. See also DocumentService API.

```javascript
var d = new sn_doc_services.SystemDocument('My document');

// Assign the doc to the human resources department
d.department('<dept_sys_id>');

var s = new sn_doc_services.DocumentService();
gs.info(JSON.stringify(s.createDocument(d), null, 2));
```

Output:

```
{
  "message": "Create document sysId : 491efbee24187410f877a6fed1c2b0a9 is successful.",
  "request_id": "491efbee24187410f877a6fed1c2b0a9",
  "status": "success"
}
```

**SystemDocument - description(String description)**

Sets the Description field of a document record.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>String</td>
<td>Description of the document.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the description field in a document record. See also: [DocumentService API](#).

```javascript
var d = new sn_doc_services.SystemDocument('My document');
d.description('Information describing the doc.);
var s = new sn_doc_services.DocumentService();
gs.info(JSON.stringify(s.createDocument(d), null, 2));
```

Output:

```json
{
  "message": "Create document sysId : 9acd7fea24587410f877a6fed1c2b060 is successful.",
  "request_id": "9acd7fea24587410f877a6fed1c2b060",
  "status": "success"
}
```

**SystemDocument - name(String docName)**

Sets the Name field of a document record.

**Note:** You can also set the name as a parameter in the `SystemDocument()` constructor.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>docName</td>
<td>String</td>
<td>Name of the document.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the Name field in a new document record. See also [DocumentService API](#).
var d = new sn_doc_services.SystemDocument();

d.name('Document named using method instead of constructor');

var s = new sn_doc_services.DocumentService();
gs.info(JSON.stringify(s.createDocument(d), null, 2));

Output:
```
{
  "message": "Create document sysId : c19d7f2624587410f877a6fed1c2b0eb is successful.",
  "request_id": "c19d7f2624587410f877a6fed1c2b0eb",
  "status": "success"
}
```

**SystemDocument** - owner(String owner)

Sets the Owner field of a document record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>owner</td>
<td>String</td>
<td>Sys_id of a user listed in the Users [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Role required: admin</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Current user</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the owner field in a new document record. See also DocumentService API.

```javascript
var d = new sn_doc_services.SystemDocument('My document');
var s = new sn_doc_services.DocumentService();
d.owner('aa826bf03710200044e0bfc8bcbe5dd6');
gs.info(JSON.stringify(s.createDocument(d), null, 2));

Output:
```
Create document sysId: 8d19f9ca24103410f877a6fed1c2b0aa is successful.

**SystemDocument - reviewers(String reviewers)**

Sets the Reviewers field of a document record.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reviewers</td>
<td>String</td>
<td>List of one or more document reviewers by sys_id. The reviewer must be a user listed in the Users [sys_user] table.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the reviewers field in a new document record. See also [DocumentService API](#).

```javascript
var d = new sn_doc_services.SystemDocument('My document');

var reviewers = '62826bf03710200044e0bfc8bcbe5df1,a8f98bb0eb32010045e1a5115206fe3a';
d.reviewers(reviewers);

var s = new sn_doc_services.DocumentService();
gs.info(JSON.stringify(s.createDocument(d), null, 2));
```

**Output:**

```json
{
  "message": "Create document sysId: 8d19f9ca24103410f877a6fed1c2b0aa is successful.",
  "request_id": "8d19f9ca24103410f877a6fed1c2b0aa",
  "status": "success"
}
```

**SystemDocument - state(String state)**

Sets the State field of a document record.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| state | String | State of the document representing where the document is in the workflow. Valid values:  
• draft  
• submit  
• review  
• complete  
Default: draft |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the state field in a new document record. See also DocumentService API.

```javascript
var d = new sn_doc_services.SystemDocument('My document');
var s = new sn_doc_services.DocumentService();
d.state('submit');
gs.info(JSON.stringify(s.createDocument(d), null, 2));
```

Output:

```
{
  "message": "Create document sysId : 19aab54e24103410f877a6fed1c2b03d is successful.\",
  "request_id": "19aab54e24103410f877a6fed1c2b03d",
  "status": "success"
}
```

**SystemDocument - template(Boolean template)**

Specifies whether a document record is a template.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>template</td>
<td>Boolean</td>
<td>Flag that specifies whether a document record is a template. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: The default version of this document is a template.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: This document is not a template.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to indicate that a document record is a template. See also DocumentService API.

```javascript
var d = new sn_doc_services.SystemDocument('My document');
d.template(true);
var s = new sn_doc_services.DocumentService();
gs.info(JSON.stringify(s.createDocument(d), null, 2));
```

Output:

```
{
    "message": "Create document sysId : f7f006ce24103410f877a6fed1c2b053 is successful.",
    "request_id": "f7f006ce24103410f877a6fed1c2b053",
    "status": "success"
}
```

**SystemDocument - type(String docType)**

Sets the Type field of a document record.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>docType</td>
<td>String</td>
<td>Type of document. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• contract</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• guideline</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• policy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• procedure</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: None</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the type field in a new document record. See also DocumentService API.

```javascript
var d = new sn_doc_services.SystemDocument('How to use the document mgmt API');
var s = new sn_doc_services.DocumentService();

d.type('guideline');

gs.info(JSON.stringify(s.createDocument(d), null, 2));
```

Output:

```json
{
    "message": "Create document sysId : 8adfb10224503410f877a6fed1c2b0e4 is successful.",
    "request_id": "8adfb10224503410f877a6fed1c2b0e4",
    "status": "success"
}
```

### SystemDocumentList - Scoped, Global


This API requires the Document Management plugin (com.snc.platform_document_management) and is provided within the `sn_doc_services` namespace. For information, see Document Services.
To create or manage a document list, use the DocumentListService API.

**SystemDocumentList - SystemDocumentList(String docListName)**

Instantiates a SystemDocumentList object.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>docListName</td>
<td>String</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td>Name of the document list.</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to instantiate a SystemDocumentList object.

```javascript
var docList = new sn_doc_services.SystemDocumentList();
```

**SystemDocumentList - description(String description)**

Sets the Description field of a new document list record.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>String</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td>Description of the document list.</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add a description to the document list. See also DocumentListService API.

```javascript
var dL = new sn_doc_services.SystemDocumentList('My document list');
// Define the document list field
dL.description('description');
```
var docList = new sn_doc_services.DocumentListService();
gs.info(JSON.stringify(docList.createDocumentList(dL), null, 2));

Output:
{
"message": "Create List for the given name : My document list, List sysId : b2c021a924683c10f877a6fed1c2b0b1 is successful.",
"request_id": "b2c021a924683c10f877a6fed1c2b0b1",
"status": "success"
}

SystemDocumentList - name(String docListName)
Sets the Name field of a document list record.

Note: You can also set the name as a parameter in the SystemDocumentList() constructor.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>docListName</td>
<td>String</td>
<td>Name of the document list.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to rename an existing document list. See also DocumentListService.

```javascript
var docListID = 'b2c021a924683c10f877a6fed1c2b0b1';
var docListUpdate = new sn_doc_services.SystemDocumentList();
docListUpdate.name('name change');

var docListSvc = new sn_doc_services.DocumentListService();
gs.info(JSON.stringify(docListSvc.updateDocumentList(docListID, docListUpdate), null, 2));
```

Output:
{
"message": "Update List for the given sysId : b2c021a924683c10f877a6fed1c2b0b1 is successful.",
"request_id": "b2c021a924683c10f877a6fed1c2b0b1",
"status": "success"
}
SystemDocumentListEntry - Scoped, Global


This API requires the Document Management plugin (com.snc.platform_document_management) and is provided within the sn_doc_services namespace. For information, see Document Services. Before using the methods in this API, add a document with its version and create a document list. Use the DocumentListEntryService API to add or remove documents from a document list.

Before using the methods in this API, you must add a document with its versions, and create a document list.

- DocumentService – Add, update, or delete a document.
- DocumentVersionService – Add, update, or delete a document version.
- DocumentListService – Add, update, or delete a document list.

SystemDocumentListEntry - SystemDocumentListEntry(String listID)

Instantiates a SystemDocumentListEntry object.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The following example shows how to instantiate a `SystemDocumentListEntry` object.

```javascript
var docList = new sn_doc_services.SystemDocumentListEntry("<docListSysId>");
```

### SystemDocumentListEntry - description(String description)

Sets the Description field of a document list entry record.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>String</td>
<td>Description of the document list entry.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

The following example shows how to add a description to the document list entry provided using the constructor. See also `DocumentListEntryService` API.

```javascript
var docListEntry = new
    sn_doc_services.SystemDocumentListEntry('21afddea2460fc10f877a6fed1c2b0dd');

docListEntry.description('birth');

docListEntry = new
    sn_doc_services.SystemDocumentListEntry /*#__PURE__*/
    Object);

gs.info(JSON.stringify(docListEntry, null, 2));
```

Output:

```json
{
    "message": "Create List Entry , List Entry sysId : d3f0b13624e8fc10f877a6fed1c2b0d9is successful.",
    "request_id": "d3f0b13624e8fc10f877a6fed1c2b0d9",
    "status": "success"
}
```

### SystemDocumentListEntry - document(String docSysId)

Specifies the document template to add to a document list.

To define a document as a template:
• Define the document object setting the `SystemDocument – template()` method to true. Create or update the document record with the methods in the `DocumentService` API.

• Select the Template check box of a document in the Documents `[ds_document]` table.

The document record specified must have a version. Use the following APIs to define and add a document version:

• `DocumentVersionService` – Add, update, or delete a document version.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>docSysId</code></td>
<td>String</td>
<td>Sys_id of a document record in the Documents <code>[ds_document]</code> table. The document specified must be a template and have a version.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to add a document template as a document list entry. See also `DocumentListEntryService` API.

```javascript
var docListEntry = new sn_doc_services.SystemDocumentListEntry('21afddea2460fc10f877a6fed1c2b0dd');
docListEntry.document("9acd7fea24587410f877a6fed1c2b060");
docListEntry.description('birth certificate');

var dlEntrySvc = new sn_doc_services.DocumentListEntryService();
gs.info(JSON.stringify(dlEntrySvc.createDocumentListEntry(docListEntry), null, 2));
```

**Output:**

```json
{
  "message": "Create List Entry , List Entry sysId : d3f0b13624e8fc10f877a6fed1c2b0d9 is successful.",
  "request_id": "d3f0b13624e8fc10f877a6fed1c2b0d9"
}
```
SystemDocumentListEntry - documentList(String listSysId)

Specifies the document list in which to add a document.

Note: You can alternatively set the document list as a parameter in the SystemDocumentListEntry() constructor.

Use the following APIs to define and create a document list:

- DocumentListService – Add, update, or delete a document list.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the document list in which to add a document. See also DocumentListEntryService API.

```javascript
var docListEntry = new sn_doc_services.SystemDocumentListEntry();
docListEntry.documentList('21afddea2460fc10f877a6fed1c2b0dd');
docListEntry.document('deaef1ce24103410f877a6fed1c2b085');
docListEntry.description('passport');

var dlEntrySvc = new sn_doc_services.DocumentListEntryService();
gs.info(JSON.stringify(dlEntrySvc.createDocumentListEntry(docListEntry), null, 2));
```

Output:

```json
{
  "message": "Create List Entry , List Entry sysId : c4ccaa8b24283050f877a6fed1c2b003is successful.",
  "request_id": "c4ccaa8b24283050f877a6fed1c2b003"
}
```
SystemDocumentVersion - Scoped, Global


This API requires the Document Management plugin (com.snc.platform_document_management) and is provided within the sn_doc_services namespace. For information, see Document Services.

To create a document version, use the DocumentVersionService API.

SystemDocumentVersion - SystemDocumentVersion(String docSysId)

Instantiates a SystemDocumentVersion object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to instantiate a SystemDocumentVersion object and create a document version. See also DocumentVersionService.

```javascript
var versionDefinition = new sn_doc_services.SystemDocumentVersion('491efbee24187410f877a6fed1c2b0a9');
versionDefinition.type('url');
versionDefinition.url('http://one/1.0');
var versionSvc = new sn_doc_services.DocumentVersionService();
gs.info(JSON.stringify(versionSvc.createDocumentVersion(versionDefinition), null, 2));
```

Output:
SystemDocumentVersion - document(String docSysId)

Specifies the document record for this version.

Note: You can alternatively set the name as a parameter in the SystemDocumentVersion() constructor.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the name field of a document version. See also DocumentVersionService – createDocumentVersion().

```javascript
var versionDefinition = new sn_doc_services.SystemDocumentVersion();
versionDefinition.document('491efbee24187410f877a6fed1c2b0a9');
var versionSvc = new sn_doc_services.DocumentVersionService();
gs.info(JSON.stringify(versionSvc.createDocumentVersion(versionDefinition), null, 2));
```

Output:

```javascript
{
    "message": "Adding an entry to DocumentVersion for the given docId : 491efbee24187410f877a6fed1c2b0a9, sysId : 1c504c91242cf810f877a6fed1c2b08b is successful.",
    "request_id": "1c504c91242cf810f877a6fed1c2b08b",
    "status": "success"
}
```
SystemDocumentVersion - type(String docType)

Sets the Type field of a document version.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>docType</td>
<td>String</td>
<td>Document source type. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• url</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• attachment</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: attachment</td>
</tr>
</tbody>
</table>

**Note:** An attachment can currently only be added in the Document Versions [ds_document_version] table UI and not with the API.

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the type field and create a document version. See also [DocumentVersionService](#).

```javascript
var versionDefinition = new sn_doc_services.SystemDocumentVersion('491efbee24187410f877a6fed1c2b0a9');
versionDefinition.type('url');
versionDefinition.url('http://one/1.0');

var versionSvc = new sn_doc_services.DocumentVersionService();
gs.info(JSON.stringify(versionSvc.createDocumentVersion(versionDefinition), null, 2));
```

```json
{
  "message": "Adding an entry to DocumentVersion for the given docId : 491efbee24187410f877a6fed1c2b0a9, sysId : d5a340d9242c81f877a6fed1c2b069 is successful."
}
```
SystemDocumentVersion - url(String url)
Sets the URL of a document version.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>URL of the source document for this version.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the URL field as the source of a document version. See also DocumentVersionService.

```javascript
var versionDefinition = new sn_doc_services.SystemDocumentVersion('491efbee24187410f877a6fed1c2b0a9');
versionDefinition.type('url');
versionDefinition.url('http://one/1.0');

var versionSvc = new sn_doc_services.DocumentVersionService();
gs.info(JSON.stringify(versionSvc.createDocumentVersion(versionDefinition), null, 2));
```

SystemDocumentVersion - versionNumber(Number versionNumber)
Sets the number of a document version.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>versionNumber</td>
<td>Number</td>
<td>Version number for this document.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows how to set the version number and create a document version. See also DocumentVersionService.

```javascript
var versionDefinition = new sn_doc_services.SystemDocumentVersion('491efbee24187410f877a6fed1c2b0a9');

versionDefinition.type('url');
versionDefinition.url('http://one/1.0');
versionDefinition.versionNumber(5);

var versionSvc = new sn_doc_services.DocumentVersionService();
gs.info(JSON.stringify(versionSvc.createDocumentVersion(versionDefinition), null, 2));
```

Output:

```json
{
  "message": "Adding an entry to DocumentVersion for the given docId: 491efbee24187410f877a6fed1c2b0a9, sysId: ff47449d242cf810f877a6fed1c2b06c is successful.",
  "request_id": "ff47449d242cf810f877a6fed1c2b06c",
  "status": "success"
}
```

## Table - Scoped, Global

Creates a Table object to add to a PDF document. Defines the data to use in each cell and sets styles, margins, and alignment.

This API is part of the ServiceNow PDF Generation Utilities plugin (com.snc.apppdfgenerator) and is provided within the sn_pdfgeneratorutils namespace. The plugin is activated by default.

This API is a component used with the Document API to generate a PDF.
Build a Table object using the methods in this class. You can use these additional classes to add cells, paragraphs, and styles to your table:

Cell API

Create a cell using the `Cell` API. Then, add the cell to your table using one of these methods:

- `Table.addCell()`
- `Table.addCellWithStyle()`
- `Table.addHeaderCell()`
- `Table.addImageCell()`
- `Table.addParagraphCell()`
- `Table.addTextCell()`

Paragraph API

Create a paragraph using the `Paragraph` API. Add the paragraph to a cell in the table using `Cell.addParagraph()`.

Style API

Set styles, colors, and borders using the `Style` API. Add the styles to the table using one of these methods:

- `Table.setDefaultStyle()`
- `Table.setHeaderStyle()`

You can then apply the Table object to a PDF document using the following class and method.

Document API

Use the `Document.addTable()` method to add your table to a PDF document. You can use the `Document.saveAsAttachment()` method to attach the document to a record.

Table - `Table(Boolean ltr, Array columnWidths, Boolean largeTable)`

Instantiates a new Table object with text direction, column width, and layout settings.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| ltr      | Boolean| Flag that indicates text direction of the language in the table. For information, see [W3C: Script direction and languages](https://www.w3.org/International/questions/qa-script-direction). Valid values:  
  - true: Text direction is left-to-right.  
  - false: Text direction is right-to-left. |
| columnWidths | Array | Numbers specifying the width of each column in a table. Sizes are based proportionally. For example, `var columnWidths = [2, 1, 1];` is a three-column table with a first column twice as large as the other two. Default: Each column is equal width. |
| largeTable | Boolean | Flag that indicates whether 100% width and fixed layout are set implicitly. Valid values:  
  - true: Table uses 100% width with a fixed layout. If true, use the `complete()` method when all content has been added to indicate that the table is complete.  
  - false: Automatic table size. Default: false |

The following example shows how to create a 2-column Table object.

```javascript
var table = new sn_pdfgeneratorutils.Table(true, [70,200], false);
```

### Table – `addCell(Cell cell)`

Adds a cell element to the table.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cell</td>
<td>Cell</td>
<td>Cell element to add to the table.</td>
</tr>
</tbody>
</table>
This example shows how to create a Table object from the incident table and add a Number and a Short Description cell to each row.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var whiteColor =  sn_pdfgeneratorutils.Color([1,1,1]);
var greyColor =  sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
var headerBgColor = new sn_pdfgeneratorutils.Color([0.4,0.6,0.8]);

// Query Incident
var gr = new GlideRecord("incident");
gr.query();

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [70,200], false);

var headerStyle = new sn_pdfgeneratorutils.Style;
headerStyle.setBackgroundColor(headerBgColor);
headerStyle.setTextAlignment("text-center");
headerStyle.setBold();
headerStyle.setFontColor(whiteColor);
table.setHeaderStyle(headerStyle);

var nParagraph = new sn_pdfgeneratorutils.Paragraph("Number");
var sParagraph = new sn_pdfgeneratorutils.Paragraph("Short Description");

var hdrCell1 = new sn_pdfgeneratorutils.Cell;
var hdrCell2 = new sn_pdfgeneratorutils.Cell;
hdrCell1.addParagraph(nParagraph);
hdrCell2.addParagraph(sParagraph);
table.addHeaderCell(hdrCell1);
table.addHeaderCell(hdrCell2);

var row = 0;
```
while(gr.next()) {
    var numCell = new sn_pdfgeneratorutils.Cell;
    var sdCell = new sn_pdfgeneratorutils.Cell;

    var numberParagraph = new sn_pdfgeneratorutils.Paragraph(gr.number);
    var sdParagraph = new sn_pdfgeneratorutils.Paragraph(gr.short_description);

    numCell.addParagraph(numberParagraph);
    sdCell.addParagraph(sdParagraph);

    if (row % 2 == 1) {
        table.setDefaultbackGroundColor(greyColor);
    } else {
        table.setDefaultbackGroundColor(whiteColor);
    }

    table.addCell(numCell);
    table.addCell(sdCell);

    row = row + 1;
}

document.addTable(table);
document.saveAsAttachment("incident", "<record_sys_id>", "filename.pdf");

**Table – addCellWithStyle(Cell cell, Style style)**

Adds a cell element with an applied style to the table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cell</td>
<td>Cell</td>
<td>Cell object to add to the table.</td>
</tr>
<tr>
<td>style</td>
<td>Style</td>
<td>Style to apply to the cell.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
This example shows how to apply a style to a specified Cell object and save the document as an attachment to a record.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var whiteColor = sn_pdfgeneratorutils.Color([1,1,1]);
var greyColor = sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
var cellBgColor = new sn_pdfgeneratorutils.Color([0.4,0.6,0.8]);

// Query Incident
var gr = new GlideRecord("incident");
gr.query();

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [70,200], false);

var cellStyle = new sn_pdfgeneratorutils.Style;
cellStyle.setBackgroundColor(cellBgColor);
cellStyle.setTextAlignment("text-center");
cellStyle.setBold();
cellStyle.setFontColor(whiteColor);

var nParagraph = new sn_pdfgeneratorutils.Paragraph("Number");
var sParagraph = new sn_pdfgeneratorutils.Paragraph("Short Description");

var hdrCell1 = new sn_pdfgeneratorutils.Cell;
var hdrCell2 = new sn_pdfgeneratorutils.Cell;

hdrCell1.addParagraph(nParagraph);
hdrCell2.addParagraph(sParagraph);

table.addHeaderCell(hdrCell1);
table.addHeaderCell(hdrCell2);

var row = 0;

while (gr.next()) {
    var numCell = new sn_pdfgeneratorutils.Cell;
    var sdCell = new sn_pdfgeneratorutils.Cell;

    var numberParagraph = new sn_pdfgeneratorutils.Paragraph(gr.number);
    var sdParagraph = new sn_pdfgeneratorutils.Paragraph(gr.short_description);
}
numCell.addParagraph(numberParagraph);
sdCell.addParagraph(sdParagraph);

if (row % 2 == 1) {
    table.setDefaultbackGroundColor(greyColor);
} else {
    table.setDefaultbackGroundColor(whiteColor);
}

table.addCellWithStyle(numCell, cellStyle);
table.addCell(sdCell);

row = row + 1;

}
document.addTable(table);
document.saveAsAttachment("incident", "sys_id", "filename.pdf");

Table – addHeaderCell(Cell cell)
Adds a header Cell object to the table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cell</td>
<td>Cell</td>
<td>Header cell to add to the table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to add a header cell to a table that contains a Paragraph object. For more information on Paragraph objects, see the Paragraph API.

var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var whiteColor = sn_pdfgeneratorutils.Color([1,1,1]);
var greyColor = sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
var headerBgColor = new sn_pdfgeneratorutils.Color([0.4,0.6,0.8]);
// Query Incident
var gr = new GlideRecord("incident");
gr.query();

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [70,200], false);

var headerStyle = new sn_pdfgeneratorutils.Style;
headerStyle.setBackgroundColor(headerBgColor);
headerStyle.setTextAlignment("text-center");
headerStyle.setBold();
headerStyle.setFontColor(whiteColor);

table.setHeaderStyle(headerStyle);

var nParagraph = new sn_pdfgeneratorutils.Paragraph("Number");
var sParagraph = new sn_pdfgeneratorutils.Paragraph("Short Description");

var hdrCell1 = new sn_pdfgeneratorutils.Cell;
var hdrCell2 = new sn_pdfgeneratorutils.Cell;

hdrCell1.addParagraph(nParagraph);
hdrCell2.addParagraph(sParagraph);

table.addHeaderCell(hdrCell1);
table.addHeaderCell(hdrCell2);

var row = 0;

while(gr.next()) {
    var numCell = new sn_pdfgeneratorutils.Cell;
    var sdCell = new sn_pdfgeneratorutils.Cell;

    var numberParagraph = new sn_pdfgeneratorutils.Paragraph(gr.number);
    var sdParagraph = new sn_pdfgeneratorutils.Paragraph(gr.short_description);

    numCell.addParagraph(numberParagraph);
    sdCell.addParagraph(sdParagraph);

    if (row % 2 == 1) {
        table.setDefaultbackGroundColor(greyColor);
    } else {
        table.setDefaultbackGroundColor(whiteColor);
    }

    row++;
}
Table – addImageCell(Image image)

Adds a cell that contains an image to the table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>image</td>
<td>Image</td>
<td>Image object to add to the cell.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to create a table with two image cells and attach the document to a record.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [70,200], false);

var icon = new sn_pdfgeneratorutils.Image("<sys_id>");
var picture = new sn_pdfgeneratorutils.Image("<sys_id>");

table.addImageCell(icon);
table.addImageCell(picture);

document.addTable(table);
document.saveAsAttachment("incident", "<record_sys_id>", "filename.pdf");
```
Table – addParagraphCell(Paragraph p)
Adds a cell that contains a paragraph to the table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>p</td>
<td>Paragraph</td>
<td>Paragraph object to add to the cell.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to create a table with a single cell containing paragraph content.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [1], false);

var paragraph = new sn_pdfgeneratorutils.Paragraph("Content to add to the cell");

table.addParagraphCell(paragraph);

document.addTable(table);

document.saveAsAttachment("incident", "<record_sys_id>", "filename.pdf");
```

Table – addTextCell(String text)
Adds a cell that contains a string to the table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>Text to add to the cell.</td>
</tr>
</tbody>
</table>
This example shows how to create a table with a single cell containing a string.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [1], false);

table.addTextCell("Text to add to the cell");

document.addTable(table);
document.saveAsAttachment("incident", "record_sys_id", "filename.pdf");
```

Table – complete()
Indicates that all the intended content has been added to a large table.

Use with the `Table.flush()` method to add additional content to a table already added to a document. This method only applies when the `largeTable` parameter in the constructor is set to true.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to indicate adding content to a table already added to a document is complete.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var whiteColor = sn_pdfgeneratorutils.Color([1,1,1]);
```
var greyColor = sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
var headerBgColor = new sn_pdfgeneratorutils.Color([0.4,0.6,0.8]);

// Query Incident
var gr = new GlideRecord("incident");
gr.query();

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [1,2,1,1], true);
var headerStyle = new sn_pdfgeneratorutils.Style;
headerStyle.setBackgroundColor(headerBgColor);
headerStyle.setTextAlignment("text-center");
headerStyle.setBold();
headerStyle.setFontColor(whiteColor);
table.setHeaderStyle(headerStyle);

var numberParagraph = new sn_pdfgeneratorutils.Paragraph("Number");
var descParagraph = new sn_pdfgeneratorutils.Paragraph("Short Description");
var stateParagraph = new sn_pdfgeneratorutils.Paragraph("State");
var assignedParagraph = new sn_pdfgeneratorutils.Paragraph("Assigned to");

var hdrCell1 = new sn_pdfgeneratorutils.Cell;
var hdrCell2 = new sn_pdfgeneratorutils.Cell;
var hdrCell3 = new sn_pdfgeneratorutils.Cell;
var hdrCell4 = new sn_pdfgeneratorutils.Cell;

hdrCell1.addParagraph(numberParagraph);
hdrCell2.addParagraph(descParagraph);
hdrCell3.addParagraph(stateParagraph);
hdrCell4.addParagraph(assignedParagraph);
table.addHeaderCell(hdrCell1);
table.addHeaderCell(hdrCell2);
table.addHeaderCell(hdrCell3);
table.addHeaderCell(hdrCell4);

var row = 0;

while(gr.next()) {
    var numCell = new sn_pdfgeneratorutils.Cell;
    var sdCell = new sn_pdfgeneratorutils.Cell;

    row++;
Table – donotSplitRowOnPageBreak(Boolean value)
Prevents splitting a row across two pages, when possible.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| value | Boolean| Flag that indicates whether to split a row across two pages, or move the entire row onto the second page. However, this property does not apply when the row spans more than a single page. Valid values:  
  • true: If the remaining part of a page is not large enough for the row, adds the entire row to a new page.  
  • false: If the remaining part of a page is not large enough for the row, splits the row across two pages.  
  Default: false |

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

This example shows how to prevent splitting a row across two pages.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [25,25,25,75,400], false);

  table.addTextCell("Text to add to the cell");
  table.addTextCell("More text to add to the cell");
  table.addTextCell("Even more text to add to the cell");
  table.addTextCell("Even more text to add to the cell");
  table.addTextCell("Even more text to add to the cell");

  table.donotSplitRowOnPageBreak(true);

  document.addTable(table);

  document.saveAsAttachment("incident", "record_sys_id", "filename.pdf");
```
Table – flush()
Adds additional content to a table that is already added to a document.

Use with the Table.complete() method to indicate that you have added all additional content to the table. This method only applies when the largeTable parameter in the constructor is set to true.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to add a text cell to a table that is already added to a document.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

var whiteColor =  sn_pdfgeneratorutils.Color([1,1,1]);
var greyColor =  sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
var headerBgColor = new sn_pdfgeneratorutils.Color([0.4,0.6,0.8]);

// Query Incident
var gr = new GlideRecord("incident");
gr.query();

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [1,2,1,1], true);

var headerStyle = new sn_pdfgeneratorutils.Style;
headerStyle.setBackgroundColor(headerBgColor);
headerStyle.setTextAlignment("text-center");
headerStyle.setBold();
headerStyle.setFontColor(whiteColor);

table.setHeaderStyle(headerStyle);
```
```javascript
var numberParagraph = new sn_pdfgeneratorutils.Paragraph("Number");
var descParagraph = new sn_pdfgeneratorutils.Paragraph("Short Description");
var stateParagraph = new sn_pdfgeneratorutils.Paragraph("State");
var assignedParagraph = new sn_pdfgeneratorutils.Paragraph("Assigned to");

var hdrCell1 = new sn_pdfgeneratorutils.Cell;
var hdrCell2 = new sn_pdfgeneratorutils.Cell;
var hdrCell3 = new sn_pdfgeneratorutils.Cell;
var hdrCell4 = new sn_pdfgeneratorutils.Cell;

hdrCell1.addParagraph(numberParagraph);
hdrCell2.addParagraph(descParagraph);
hdrCell3.addParagraph(stateParagraph);
hdrCell4.addParagraph(assignedParagraph);

table.addHeaderCell(hdrCell1);
table.addHeaderCell(hdrCell2);
table.addHeaderCell(hdrCell3);
table.addHeaderCell(hdrCell4);

var row = 0;

while(gr.next()) {
  var numCell = new sn_pdfgeneratorutils.Cell;
  var sdCell = new sn_pdfgeneratorutils.Cell;
  var stateCell = new sn_pdfgeneratorutils.Cell;
  var assignedCell = new sn_pdfgeneratorutils.Cell;

  var numberParagraph = new sn_pdfgeneratorutils.Paragraph(gr.number);
  var sdParagraph = new sn_pdfgeneratorutils.Paragraph(gr.short_description);
  var stateParagraph = new sn_pdfgeneratorutils.Paragraph(gr.state);
  var assignedParagraph = new sn_pdfgeneratorutils.Paragraph(gr.assigned_to);

  numCell.addParagraph(numberParagraph);
  sdCell.addParagraph(sdParagraph);
  stateCell.addParagraph(stateParagraph);
  assignedCell.addParagraph(assignedParagraph);

  if (row % 2 == 1) {
    table.setDefaultbackGroundColor(greyColor);
  } else {
    // Add other code here...
  }
  row++;
}
```

```javascript
  table.setDefaultBackgroundColor(whiteColor);
  
  table.addCell(numCell);
  table.addCell(sdCell);
  table.addCell(stateCell);
  table.addCell(assignedCell);
  
  row = row + 1;
  
}

document.addTable(table);
  
table.addTextCell("A cell added later");
  table.flush();
  table.complete();
  document.saveAsAttachment("incident", "<record_sys_id>", "filename.pdf");
```

### Table – `getDefaultStyle()`
Retrieves the default style.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Style</td>
<td>Default style.</td>
</tr>
</tbody>
</table>

This example shows how to set and then return the default style.

```javascript
var table = new sn_pdfgeneratorutils.Table(true, [1], false);

var style = new sn_pdfgeneratorutils.Style();

style.setItalic();

table.setDefaultStyle(style);

```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
var defaultStyle = table.getDefaultStyle();

gs.info(defaultStyle);

Table – getHeaderStyle()

Returns the style applied to the table’s header.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Style</td>
<td>Style applied to the table’s header.</td>
</tr>
</tbody>
</table>

This example shows how to set and return the table's header style.

```javascript
var table = new sn_pdfgeneratorutils.Table(true, [1,1], false);

var whiteColor = sn_pdfgeneratorutils.Color([1,1,1]);
var headerBgColor = new sn_pdfgeneratorutils.Color([0.4,0.6,0.8]);

var headerStyle = new sn_pdfgeneratorutils.Style;
headerStyle.setBackgroundColor(headerBgColor);
headerStyle.setTextAlignment("text-center");
headerStyle.setBold();
headerStyle.setFontColor(whiteColor);

table.setHeaderStyle(headerStyle);

var hdrCell1 = new sn_pdfgeneratorutils.Cell;
var hdrCell2 = new sn_pdfgeneratorutils.Cell;

table.addHeaderCell(hdrCell1);
table.addHeaderCell(hdrCell2);

var styleObject = table.getHeaderStyle();

gs.info(styleObject);
```
Table – `setBorder(Number width)`

Sets a border of designated width around the outer edges of the table.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Width of the border.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unit: Points</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to create a table with two-pixel border and attach the table to a record.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [1], false);

  table.addTextCell("Text to add to the cell");
  table.setBorder(2);

document.addTable(table);
"document.saveAsAttachment("incident", "<record_sys_id>", "filename.pdf");
```

Table – `setDefaultBackGroundColor(Color color)`

Sets the table’s default background color.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>Color</td>
<td>Color object used as the default background color for the table.</td>
</tr>
</tbody>
</table>
This example shows how to set the default background color for the table.

```javascript
// declare table by providing width array with automatic table size
var table = new sn_pdfgeneratorutils.Table(true, [1], false);

var color = new sn_pdfgeneratorutils.Color([0.8, 0.8, 0.8]);

table.setDefaultbackGroundColor(color);

table.addTextCell("Text to add to the cell");
```

**Table – setDefaultStyle(Style defaultStyle)**

Sets the default style to apply to the table.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultStyle</td>
<td>Style</td>
<td>Default style to apply to the table.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to apply a default style to a table.

```javascript
var table = new sn_pdfgeneratorutils.Table(true, [1], false);

var style = new sn_pdfgeneratorutils.Style();

style.setItalic();

table.setDefaultStyle(style);

table.addTextCell("Text to add to the cell");

var defaultStyle = table.getDefaultStyle();
```
Table – setFixedPosition(Number pageNumber, Number left, Number bottom, Number width)

Sets the table to a fixed position on the page.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pageNumber</td>
<td>Number</td>
<td>Number of the page to add the table to.</td>
</tr>
<tr>
<td>left</td>
<td>Number</td>
<td>Number of pixels from the left margin to add the table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unit: Points</td>
</tr>
<tr>
<td>bottom</td>
<td>Number</td>
<td>Number of pixels from the bottom margin to add the table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unit: Points</td>
</tr>
<tr>
<td>width</td>
<td>Number</td>
<td>Width of the table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unit: Points</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to reposition the table into the bottom left corner of the page.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [1], false);

table.addTextCell("Text to add to the cell");
table.setFixedPosition(1, 36, 36, 500);
document.addTable(table);
document.saveAsAttachment("incident", "<record_sys_id>", "filename.pdf");
```
Table – setHeaderStyle(Style headerStyle)
Sets the Style object to apply to the table’s header.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>headerStyle</td>
<td>Style</td>
<td>Style object to apply to the table’s header.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to set and return the table’s header style.

```javascript
var table = new sn_pdfgeneratorutils.Table(true, [1,1], false);

var whiteColor = sn_pdfgeneratorutils.Color([1,1,1]);
var headerBgColor = new sn_pdfgeneratorutils.Color([0.4,0.6,0.8]);

var headerStyle = new sn_pdfgeneratorutils.Style;
headerStyle.setBackgroundColor(headerBgColor);
headerStyle.setTextAlignment("text-center");
headerStyle.setBold();
headerStyle.setFontColor(whiteColor);

table.setHeaderStyle(headerStyle);

var hdrCell1 = new sn_pdfgeneratorutils.Cell;
var hdrCell2 = new sn_pdfgeneratorutils.Cell;
table.addHeaderCell(hdrCell1);
table.addHeaderCell(hdrCell2);

var styleObject = table.getHeaderStyle();
gs.info(styleObject);
```

Table – setHorizontalAlignment(String alignment)
Sets the horizontal alignment of the table.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alignment</td>
<td>String</td>
<td>Alignment for the table. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• center: Align contents to the center.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• left: Align contents to the left.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• right: Align contents to the right.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to create a small table aligned in the center of the page.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [1], false);

table.addTextCell("Text to add to the cell");
table.setWidth(90);
table.setHorizontalAlignment("center");

document.addTable(table);
document.saveAsAttachment("incident", "<record_sys_id>", "filename.pdf");
```

**Table – setMargin(Number margin)**

Sets all margins around the table to the same width.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>margin</td>
<td>Number</td>
<td>Value of the top, right, bottom, and left margins in points.</td>
</tr>
</tbody>
</table>
This example shows how to set a margin for the entire table.

    var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
    var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

    // declare table by providing width array and Boolean for large table
    var table = new sn_pdfgeneratorutils.Table(true, [1], false);

    table.addTextCell("Text to add to the cell");
    table.setMargin(3);
    document.addTable(table);
    document.saveAsAttachment("incident", "<record_sys_id>", "filename.pdf");

Table – setMarginBottom(Number margin)

Sets the margin at the bottom of the page.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>margin</td>
<td>Number</td>
<td>Height of the bottom margin in points.</td>
</tr>
</tbody>
</table>

This example shows how to set a margin for the bottom of the page to three points.

    var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
    var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

    var whiteColor =  sn_pdfgeneratorutils.Color([1,1,1]);
    var greyColor =  sn_pdfgeneratorutils.Color([0.8,0.8,0.8]);
    var headerBgColor = new sn_pdfgeneratorutils.Color([0.4,0.6,0.8]);

| Returns
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
```javascript
// Query Incident
var gr = new GlideRecord("incident");
gr.query();

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [70, 200], false);

var headerStyle = new sn_pdfgeneratorutils.Style;
headerStyle.setBackgroundColor(headerBgColor);
headerStyle.setTextAlignment("text-center");
headerStyle.setBold();
headerStyle.setFontColor(whiteColor);
table.setHeaderStyle(headerStyle);

var nParagraph = new sn_pdfgeneratorutils.Paragraph("Number");
var sParagraph = new sn_pdfgeneratorutils.Paragraph("Short Description");

var hdrCell1 = new sn_pdfgeneratorutils.Cell;
var hdrCell2 = new sn_pdfgeneratorutils.Cell;

hdrCell1.addParagraph(nParagraph);
hdrCell2.addParagraph(sParagraph);
table.addHeaderCell(hdrCell1);
table.addHeaderCell(hdrCell2);

// set the bottom margin to three points
table.setBottomMargin(3);

var row = 0;

while(gr.next()) {
    var numCell = new sn_pdfgeneratorutils.Cell;
    var sdCell = new sn_pdfgeneratorutils.Cell;

    var numberParagraph = new sn_pdfgeneratorutils.Paragraph(gr.number);
    var sdParagraph = new sn_pdfgeneratorutils.Paragraph(gr.short_description);

    numCell.addParagraph(numberParagraph);
sdCell.addParagraph(sdParagraph);

    if (row % 2 == 1) {
        table.setDefaultBackGroundColor(greyColor);
    }
    row++;
}
```
} else {
    table.setDefaultBackgroundColor(whiteColor);
}

table.addCell(numCell);
table.addCell(sdCell);

row = row + 1;
}

document.addTable(table);
document.saveAsAttachment("incident", ",<record_sys_id>", "filename.pdf");

Table – setMarginLeft(Number margin)
Sets the margin at the left side of the page.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>margin</td>
<td>Number</td>
<td>Width of the left margin in points.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to set a margin for the left side of the page.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [1], false);

table.addTextCell("Text to add to the cell");
table.setMarginLeft(100);

document.addTable(table);
document.saveAsAttachment("incident", ",<record_sys_id>", "filename.pdf");
```
Table – setMarginRight(Number margin)

Sets the margin at the right side of the page.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>margin</td>
<td>Number</td>
<td>Width of the right margin in points.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

This example shows how to set a margin for the left side of the page.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [1], false);

table.addTextCell("Text to add to the cell");
table.setMarginRight(100);
document.addTable(table);
"document.saveAsAttachment("incident", ", record_sys_id", "filename.pdf");
```

Table – setMarginTop(Number margin)

Sets a margin at the top of the page.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>margin</td>
<td>Number</td>
<td>Height of the top margin in points.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
This example shows how to set a margin at the top of the page.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [1], false);

    table.addTextCell("Text to add to the cell");
    table.setMarginTop(100);

document.addTable(table);
document.saveAsAttachment("incident", "<record_sys_id>", "filename.pdf");
```

### Table – setWidth(Number width)
Sets the table's total width.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Number</td>
<td>Width of the table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unit: Points</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to create a table 90 points wide and attach the document to a record.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
var document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [1], false);

    table.addTextCell("Text to add to the cell");
    table.setWidth(90);
    table.setHorizontalAlignment("center");
    table.setMarginTop(100);

document.addTable(table);
document.saveAsAttachment("incident", "<record_sys_id>", "filename.pdf");
```
Table – useAllAvailableWidth()
Expands the table to use the entire width available on the page.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to create a table that expands to the available width on the page.

```javascript
var pageSize = new sn_pdfgeneratorutils.PdfPage("A4");
document = new sn_pdfgeneratorutils.Document.createDocument(pageSize);

// declare table by providing width array and Boolean for large table
var table = new sn_pdfgeneratorutils.Table(true, [1], false);

table.addTextCell("Text to add to the cell");
table.useAllAvailableWidth();
document.addTable(table);
document.saveAsAttachment("incident", ":<record_sys_id>:", "filename.pdf");
```

TableUtils - Global
TableUtils is a class of shortcuts for accessing table related information. The TableUtils class is available to server-side scripts.

TableUtils - tableExists()
Checks to see if a table exists.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the table exists.</td>
</tr>
</tbody>
</table>

```javascript
var table = new TableUtils("my_table");
gs.print("Does 'my_table' exist? " + table.tableExists());
```

Output: Does 'my_table' exist? false

**TableUtils - drop(String tableName)**

Drops a database table.

**Note:** Use with extreme caution. Dropping a table permanently deletes the table and all its data. If the table is extended, use `dropTableAndExtensions`.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table to drop</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var tu = new TableUtils();
tu.drop("table_that_will_be_lost_forever");
```

Output:

dropping table table_that_will_be_lost_forever
Starting cache flush
Cache flush complete
TABLE DROP: admin dropped table table_that_will_be_lost_forever
**TableUtils - dropAndClean(String tableName)**

Drops a database table and cleans up references to the table.

⚠️ **Note:** Use with extreme caution. Dropping a database table permanently deletes the table and all its data. If the table is extended, use `dropTableAndExtensions`.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>Name of the table to drop</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var tu = new TableUtils();
tu.dropAndClean("table_that_will_be_lost_forever");
```

**Output:**

```
dropping table table_that_will_be_lost_forever
Starting cache flush
Cache flush complete
TABLE DROP: admin dropped table table_that_will_be_lost_forever
*** Script: removing gauges for table_that_will_be_lost_forever
*** Script: removing forms for table_that_will_be_lost_forever
*** Script: removing styles for table_that_will_be_lost_forever
*** Script: removing forms sections for table_that_will_be_lost_forever
*** Script: removing lists for table_that_will_be_lost_forever
*** Script: removing related lists for table_that_will_be_lost_forever
*** Script: removing references to table_that_will_be_lost_forever
*** Script: removing dictionary entries for table_that_will_be_lost_forever
Background message, type:info, message: Table deleted
```

**TableUtils - dropTableAndExtensions(String tableName)**

Drops a database table, all of it’s extended tables, and cleans up references to the tables.
Note: Use with extreme caution. Dropping a database table permanently deletes the table and all of its data.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>The table to drop</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var tu = new TableUtils();
tu.dropTableAndExtensions("table_that_will_be_lost_forever");
```

Output:

dropping table parent_table_that_will_be_lost_forever
Starting cache flush
Cache flush complete
TABLE DROP: admin dropped table ext_table_that_will_be_lost_forever
removing gauges for ext_table_that_will_be_lost_forever
removing forms for ext_table_that_will_be_lost_forever
removing styles for ext_table_that_will_be_lost_forever
removing forms sections for ext_table_that_will_be_lost_forever
removing lists for ext_table_that_will_be_lost_forever
removing related lists for ext_table_that_will_be_lost_forever
removing references to ext_table_that_will_be_lost_forever
removing dictionary entries for ext_table_that_will_be_lost_forever
Background message, type:info, message: Table deleted
removing gauges for parent_table_that_will_be_lost_forever
removing forms for parent_table_that_will_be_lost_forever
removing styles for parent_table_that_will_be_lost_forever
removing forms sections for parent_table_that_will_be_lost_forever
removing lists for parent_table_that_will_be_lost_forever
removing related lists for parent_table_that_will_be_lost_forever
removing references to parent_table_that_will_be_lost_forever
removing dictionary entries for parent_table_that_will_be_lost_forever
Background message, type:info, message: Table deleted

TableUtils - getAbsoluteBase()
Returns the base table name from which the table was extended.

ℹ️ **Note:** For any table under the cmdb_ci hierarchy, this method returns cmdb_ci and not cmdb, which is the actual base table.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

```javascript
var table = new TableUtils("cmdb_ci_server");
gs.print(table.getAbsoluteBase());
```

Output: cmdb_ci

TableUtils - getAllExtensions()
Returns the list of tables that extend a table, includes the base table.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>ArrayList</td>
</tr>
</tbody>
</table>
```javascript
var table = new TableUtils("task");
gs.print(table.getAllExtensions());
```

Output:
```javascript
[task, incident, issue, kb_submission, sysapproval_group, change_request,
  change_request_imac, sc_task,
  problem, sc_req_item, ticket, ast_transfer_order, planned_task, change_task, change_phase,
  sc_request]
```

**TableUtils - getHierarchy()**

Returns a list of all classes participating in the hierarchy of the specified table.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArrayList</td>
<td>A list of all classes in the specified table's hierarchy.</td>
</tr>
</tbody>
</table>

```javascript
var table = new TableUtils("cmdb_ci_server");
gs.print(table.getHierarchy());
```

Output: `[cmdb_ci_server, cmdb_ci_computer, cmdb_ci_hardware, cmdb_ci,
    cmdb_ci_mainframe, cmdb_ci_linux_server, cmdb_ci_mainframe_lpar,
    cmdb_ci_esx_server, cmdb_ci_unix_server, cmdb_ci_solaris_server,
    cmdb_ci_hpux_server, cmdb_ci_aix_server, cmdb_ci_osx_server,
    cmdb_ci_netware_server, cmdb_ci_win_server]

**TableUtils - getTables()**

Returns the table hierarchy.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArrayList</td>
<td>A list of table names in the parent hierarchy.</td>
</tr>
</tbody>
</table>

// Get the hierarchy of tables
var table = new TableUtils("cmdb_ci_computer");
var tableArrayList = table.getTables();

// Use the j2js method to convert the Java ArrayList to JavaScript
gs.include("j2js");
var tableArray = j2js(tableArrayList);

// Write the value of each element in the JavaScript array
var i = 0;
while (i < tableArray.length) {
    gs.print("Table with index "+ i + ": " + tableArray[i]);
    i++;
}

Output:
Table with index 0: cmdb_ci_computer
Table with index 1: cmdb_ci_hardware
Table with index 2: cmdb_ci
Table with index 3: cmdb

TableUtils - getTableExtensions()
Returns a list of tables that extend a table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArrayList</td>
<td>A list of table names that extend the table.</td>
</tr>
</tbody>
</table>
This example shows that 58 tables in this instance extend the Computer [cmdb_ci_computer] table. (Output condensed below.)

```javascript
// Get the tables that extend the table
var table = new TableUtils("cmdb_ci_computer");
var tableArrayList = table.getTableExtensions();

// Use the j2js method to convert the Java ArrayList to JavaScript
gs.include("j2js");
var tableArray = j2js(tableArrayList);

// Write the value of each element in the JavaScript array
var i = 0;
while ( i < tableArray.length ) {
    gs.print("Table with index "+i+: "+tableArray[i]);
    i++;
}
```

Output:

- Table with index 0: cmdb_ci_mainframe_hardware
- Table with index 1: cmdb_ci_handheld_computing
- Table with index 2: cmdb_ci_ucs_blade
- Table with index 3: cmdb_ci_storage_switch
- Table with index 4: cmdb_ci_server
- Table with index 5: cmdb_ci_hmc_server
- ...
- Table with index 56: cmdb_ci_pc_hardware
- Table with index 57: cmdb_ci_ucs_rack_unit

**TableUtils - hasExtensions()**

Determines if a table has been extended.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Boolean</td>
<td>True if the table has extensions.</td>
</tr>
</tbody>
</table>
var table = new TableUtils("cmdb_ci_server");
go.print(table.hasExtensions());

Output:
true

**TableUtils - isBaseClass()**

Determines if a table is a base class, meaning it has no parents and has extensions.

For example, Task is a base class since it is not extended from another table and has tables extended from it. Sys_user is not a base class because it has no parents, but does not have extensions.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean     | Flag that indicates whether a table is a base class table meaning it has no parents but has extensions. Valid values:  
  • true: Table is a base class.  
  • false: Table is not a base class. |

### Example

```javascript
var table = new TableUtils("task");
go.print("Task is base class: " + table.isBaseClass());

var table = new TableUtils("sys_user");
go.print("User is base class: " + table.isBaseClass());
```

Output:

```
Task is base class: true
User is base class: false
```
**TableUtils - isSoloClass()**

Determines if the table has no parents and no extensions.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the table has no parent and no table is extended from it.</td>
</tr>
</tbody>
</table>

```javascript
var table = new TableUtils("task");
gs.print("task is solo class: " + table.isSoloClass());

var table = new TableUtils("cmdb_ci_win_server");
gs.print("cmdb_ci_win_server is solo class: " + table.isSoloClass());

var table = new TableUtils("sys_user");
gs.print("sys_user is solo class: " + table.isSoloClass());
```

**Output:**

```
task is solo class: false
cmdb_ci_win_server is solo class: false
sys_user is solo class: true
```

**TableUtils - TableUtils(String tableName)**

Creates an instance of a TableUtils class.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>String</td>
<td>The table name</td>
</tr>
</tbody>
</table>

```javascript
var tu = new TableUtils("incident");
```
Scoped TemplatePrinter handles printing from a mail script to the email message.

There is no constructor for the scoped TemplatePrinter API. The methods are called in mail scripts using the `template` global variable.

**Scoped TemplatePrinter - print(String string)**

Prints the string to the email body.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>String</td>
<td>The string to print</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
template.print("Incident number - " + current.number + "\n");
```

**Scoped TemplatePrinter - space(Number spaces)**

Adds non-breaking spaces to the email body.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spaces</td>
<td>Number</td>
<td>The number of non-breaking spaces to output to the email body.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
template.space(4);
```
**TimelineSpan - Global**

This class defines a set of properties that describe the characteristics and interactive behavior of an element rendered within a `TimelineItem`.

Since it is extremely important for all of a `TimelineItem`'s collection of spans to be unique, the creation of a new instance should be performed via the `createTimelineItem` method of an existing `TimelineItem` instance.

**TimelineSpan - addPredecessor(Object Array objArray)**

Adds multiple relationships between the current instance and other `TimelineSpan` objects by enumerating through the array of JavaScript objects.

Each object should have an internal property `relationship_sys_id` and `predecessor_sys_id` specified.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>objArray</td>
<td>Object Array</td>
<td>JavaScript object array that contains two internal properties: <code>relationship_sys_id</code> and <code>predecessor_sys_id</code>.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**TimelineSpan - addPredecessor(String strPredecessorSysId, String strRelationshipSysId, String strTableName)**

Adds the specified relationship between the current instance and another `TimelineSpan` and allows the relationship to open a `GlideWindow` to display information about the relationship.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strPredecessorSysId</td>
<td>String</td>
<td>The sys ID of the planned task that is the predecessor of the relationship.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strRelationshipSysId</td>
<td>String</td>
<td>The sys ID of the relationship of the relationship.</td>
</tr>
<tr>
<td>strTableName</td>
<td>String</td>
<td>The name of the table for the relationship.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

TimelineSpan - addPredecessor(String strPredecessorSysId, String strRelationshipSysId)

Adds the specified relationship between the current instance and another TimelineSpan with sys ID `strPredecessorSysId`.

The drawn line will not have any double click handlers associated with it.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strPredecessorSysId</td>
<td>String</td>
<td>The sys ID of the planned task that is the predecessor of the relationship.</td>
</tr>
<tr>
<td>strRelationshipSysId</td>
<td>String</td>
<td>The sys ID of the relationship of the relationship.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

TimelineSpan - getAllowXDragLeft()

Returns the boolean value of the `AllowXDragLeft` property.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the object’s start time can be adjusted; False otherwise.</td>
</tr>
</tbody>
</table>

**TimelineSpan - getAllowXDragRight()**

Returns the boolean value of the `AllowXDragRight` property.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the object’s end time can be adjusted; false otherwise.</td>
</tr>
</tbody>
</table>

**TimelineSpan - getAllowXMove()**

Returns the boolean value of the `AllowXMove` property.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the object can be moved; false otherwise.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**TimelineSpan - getAllowYMove()**

Returns the boolean value of the `AllowYMove` property.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the object can be moved vertically; false otherwise.</td>
</tr>
</tbody>
</table>

**TimelineSpan - getAllowYMovePredecessor()**

Returns the boolean value of the `AllowYMovePredecessor` property.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the a dashed relationship line can be drawn from the current object to a new successor; false otherwise.</td>
</tr>
</tbody>
</table>

**TimelineSpan - getInnerSegmentClass()**

Returns the name of the current inner segment class for the TimelineSpan.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the class for the current inner segment style.</td>
</tr>
</tbody>
</table>

**TimelineSpan - getInnerSegmentEndTimeMs()**

Returns the time in milliseconds of the end time of the inner segment portion of the *TimelineSpan*.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The end time of the TimelineSpan inner segment portion in milliseconds.</td>
</tr>
</tbody>
</table>

**TimelineSpan - getInnerSegmentStartTimeMs()**

Returns the time in milliseconds of the start time of the inner segment portion of the *TimelineSpan*.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The start time of the TimelineSpan inner segment portion in milliseconds.</td>
</tr>
</tbody>
</table>
**TimelineSpan - getIsChanged()**

Returns a boolean that specifies whether or not the current timeline item has been modified after initialization.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the current span has been marked as changed; otherwise false.</td>
</tr>
</tbody>
</table>

**TimelineSpan - getPointIconClass()**

Returns a string that specifies the name of the icon class to use for displaying the element on the timeline if the current instance has zero duration.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The name of the icon class to use for displaying the current TimelineSpan if the duration is zero.</td>
</tr>
</tbody>
</table>

**TimelineSpan - getPredecessors()**

Returns an array of all the predecessor objects associated with the current instance. Each array object is a HashMap that contains a predecessor_sys_id and relationship_sys_id property.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>List of HashMaps that contain two internal properties: predecessor_sys_id and relationship_sys_id.</td>
</tr>
<tr>
<td>Array</td>
<td></td>
</tr>
</tbody>
</table>

**TimelineSpan - getSpanColor()**

Returns the string name of the color specified for displaying this span.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The HTML color name to use as the background color for the element.</td>
</tr>
</tbody>
</table>

**TimelineSpan - getSpanText()**

Returns the string that specifies the text to display adjacent to the time element.

**Note:** This text will only display if the GlideTimeline object has enabled timeline text via glideTimeline.showTimelineText(true).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>---------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>Text displayed adjacent to the element.</td>
<td></td>
</tr>
</tbody>
</table>

**TimelineSpan - getStartTimeMs()**

Returns the start time in milliseconds of the current `TimelineSpan` object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The start time of the element in milliseconds.</td>
</tr>
</tbody>
</table>

**TimelineSpan - getSysId()**

Returns the sys ID of the current object.

This method is useful for returning the sys Id when the current object instance was created without a specific sys Id to obtain the dynamically generated GUID.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The unique system ID of the current element.</td>
</tr>
</tbody>
</table>
**TimelineSpan - getTable()**

Returns the name of the table where the sys ID is referenced.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>none</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

**TimelineSpan - getTooltip()**

Returns the text/html to display in the tooltip when the **TimelineSpan** element is being hovered over.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>none</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

**TimelineSpan - setAllowXDragLeft(Boolean bFlag)**

Sets a flag that determines whether the element’s start date can be dragged left or right therefore adjusting the duration of the task.

The effect of this behavior is controlled by the script include that handles the appropriate event. The default value for this property is **false**.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bFlag</td>
<td>Boolean</td>
<td><strong>True</strong> to enable the element's start date to be adjusted; <strong>false</strong> otherwise.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### TimelineSpan - `setAllowXDragRight(Boolean bFlag)`

Sets a flag that determines whether the element's end date can be dragged left or right therefore adjusting the duration of the task.

The effect of this behavior is controlled by the script include that handles the appropriate event. The default value for this property is **false**.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bFlag</td>
<td>Boolean</td>
<td><strong>True</strong> to enable the element's end date to be adjusted; <strong>false</strong> otherwise.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### TimelineSpan - `setAllowXMove(Boolean bFlag)`

Sets a flag that determines whether the element can be moved to start at a different time.

The effect of this behavior is controlled by the script include that handles the appropriate event. The default value for this property is **false**.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bFlag</td>
<td>Boolean</td>
<td>True to enable the element to be moved horizontally; false otherwise.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**TimelineSpan - setAllowYMove(Boolean bFlag )**

Sets a flag that determines whether the element can be dragged vertically on the timeline.

The effect of this behavior is controlled by the script include that handles the appropriate event. The default value for this property is **false**.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bFlag</td>
<td>Boolean</td>
<td>True to enable the element to be moved vertically; false otherwise.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**TimelineSpan - setAllowYMovePredecessor(Boolean bFlag)**

Sets a flag that determines whether a dashed relationship line can be drawn from this element interactively on the timeline.

The effect of this behavior is controlled by the script include that handles the appropriate event. The default value for this property is false.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bFlag</td>
<td>Boolean</td>
<td><strong>True</strong> to enable a relationship line to be drawn from this element; <strong>false</strong> otherwise.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

TimelineSpan - setInnerSegmentClass(String styleClass)

Specifies the name of the class to use for stylizing the inner segment if it exists.

The default value is **green**.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>styleClass</td>
<td>String</td>
<td>One of the following values: <strong>green</strong>, <strong>blue</strong>, or <strong>silver</strong>.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

TimelineSpan - setInnerSegmentTimeSpan(Number startTimeMs, Number endTimeMs)

Creates an inner segment to show within the current timespan defined by the range specified.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startTimeMs</td>
<td>Number</td>
<td>The start time in milliseconds.</td>
</tr>
<tr>
<td>endTimeMs</td>
<td>Number</td>
<td>The end time in milliseconds.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**TimelineSpan - setPointIconClass(String iconClassName)**

Sets the icon class to use for displaying the current element on the timeline if the current instance has zero duration.

⚠️ Note: This only affects the current TimelineSpan object and will take precedence over the defaultPointIconClass specified by the GlideTimeline.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iconClassName</td>
<td>String</td>
<td>String that specifies one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• milestone</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• blue_square</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• sepia_square</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• green_square</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• red_square</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• black_square</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• blue_circle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• sepia_circle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• green_circle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• red_circle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• black_circle</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**TimelineSpan - setSpanColor(String strColor)**

Sets the color for displaying this span.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strColor</td>
<td>String</td>
<td>The HTML color name for the color of this span.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### TimelineSpan - setSpanText(String strSpanText)

Sets the text to display adjacent to the time element.

**Note:** This text will only be displayed if the GlideTimeline object has enabled timeline text via `glideTimeline.showTimelineText(true)`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strSpanText</td>
<td>String</td>
<td>The text to display next to the time element.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### TimelineSpan - setTimeSpan(Number nStartTime, Number nEndTimeMs)

Sets the start and end dates for the current span.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nStartTime</td>
<td>Number</td>
<td>The start time in milliseconds.</td>
</tr>
<tr>
<td>nEndTimeMs</td>
<td>Number</td>
<td>The end time in milliseconds.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**TimelineSpan - setTimeSpan(String strStartTime, String strEndTimeMs)**

Sets the start and end times for the current span.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strStartTime</td>
<td>String</td>
<td>The start time in milliseconds.</td>
</tr>
<tr>
<td>strEndTimeMs</td>
<td>String</td>
<td>The end time in milliseconds.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**TimelineSpan - setTooltip(String strTooltipText)**

Sets the text to display in the tooltip when the `TimelineSpan` element is being hovered over.

**Note:** You can specify valid HTML in the string that sets the tooltip.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strTooltipText</td>
<td>String</td>
<td>The text to display in the tooltip.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
TaskStateUtil - Global

The TaskStateUtil API is in the task state management utility script include and is used for working with task-type table state attributes.

The TaskStateUtil API is primarily used by the Task Active State Management business rule to set the active field based on state changes. Configurations are defined in the task.state dictionary element, usually using dictionary overrides since state values vary by table.

The TaskStateUtil API can be called by any server script to determine inactive states, default work, or default close states for a table.

The required attributes are defined on the planned_task table so all planned task types are supported. We will eventually add the attributes to other task types and eventually the base task table. You are free to do this if you want to leverage this feature now.

These attributes can be defined on the task.state dictionary element or a dictionary override for extended task tables.

### Related Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>close_states</td>
<td>Semicolon delimited list of state values that are inactive, used to identify whether the task should be set to active or inactive. This is a required attribute to use the TaskStateUtil functionality.</td>
</tr>
<tr>
<td>default_close_state</td>
<td>Optional attribute to define the state value of the default close state if you want to define business rules that automatically close a task. Defaults to 3, typically Closed Complete if attribute is not defined.</td>
</tr>
<tr>
<td>default_work_state</td>
<td>Optional attribute to define the state value of the default working state if you want to define business rules that automatically set a task for working. Defaults to 2, typically Work in Progress if the attribute is not defined.</td>
</tr>
</tbody>
</table>
TaskStateUtil - ATTR_DEFAULT_WORK
The name of the attribute that identifies default work state.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATTR_DEFAULT_WORK</td>
<td>String</td>
<td>Identifies default work state. Value: default_work_state</td>
</tr>
</tbody>
</table>

TaskStateUtil - ATTR_DEFAULT_CLOSE
The name of the attribute that identifies the default close state.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATTR_DEFAULT_CLOSE</td>
<td>String</td>
<td>Identifies the default close state. Value: default_close_state</td>
</tr>
</tbody>
</table>

TaskStateUtil - ATTR_INACTIVE_STATES
The name of the attribute that identifies inactive states.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATTR_INACTIVE_STATES</td>
<td>String</td>
<td>Identifies inactive states. Value: close_states</td>
</tr>
</tbody>
</table>

TaskStateUtil - getDefaultCloseState
Returns the value for the default closed state.

The default closed state value is 3 if the default_close_state attribute has not been specified.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>State value representing the closed state.</td>
</tr>
</tbody>
</table>

```javascript
var stateUtil = new TaskStateUtil(current);
// get the close state
var defaultCloseState = stateUtil.getDefaultCloseState();
current.state = defaultCloseState;
```

**TaskStateUtil - getDefaultWorkState()**

Returns the value for the default work state.

The default work state value is 2 if the default_work_state attribute has not been specified.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The state value representing the working state.</td>
</tr>
</tbody>
</table>

```javascript
var stateUtil = new TaskStateUtil(current);
// get the work state
var defaultWorkState = stateUtil.getDefaultWorkState();
current.state = defaultWorkState;
```

**TaskStateUtil - getInactiveStates**

Returns a list of the inactive state values.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Array of state values that are inactive.</td>
</tr>
</tbody>
</table>

```javascript
var stateUtil = new TaskStateUtil(current);
//get the inactive state values
var inactiveStates = stateUtil.getInActiveStates();
```

**TaskStateUtil - isStateInactive(String state)**

Returns the active status of the specified state.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>String</td>
<td>The state value to check.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the state is inactive.</td>
</tr>
</tbody>
</table>

```javascript
var stateUtil = new TaskStateUtil(current);
var previousStateInactive = stateUtil.isStateInactive(previous.state);
var currentStateInactive = stateUtil.isStateInactive(current.state);
```

**TaskStateUtil - runMarkClosed**

Decides whether the **mark closed** business rule should be run or not.
### TaskStateUtil - runTaskCloser

Decides whether the task closer business rule should be run or not.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Whether the business rule should be allowed to run or not.</td>
</tr>
</tbody>
</table>

### TaskStateUtil - runTaskReopener

Decides whether the task reopener business rule should be run or not.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Whether the business rule should be allowed to run or not.</td>
</tr>
</tbody>
</table>
TaskStateUtil - setDefaultWorkState(String defaultWorkState)

Enables the user to specify their own default work state.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultWorkState</td>
<td>String</td>
<td>The value to use for the default work state.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TaskStateUtil</td>
<td>A self-reference to allow for method chaining.</td>
</tr>
</tbody>
</table>

TaskStateUtil - SYSTEM_DEFAULT_CLOSE

The value of the default close state is Closed Complete on the Task table.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM_DEFAULT_CLOSE</td>
<td>Integer</td>
<td>Value of the default close state is Closed Complete on the Task table. Value: 3</td>
</tr>
</tbody>
</table>

TaskStateUtil - SYSTEM_DEFAULT_WORK

The value of the default work state is Work in progress on the Task table.

Field

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM_DEFAULT_WORK</td>
<td>Integer</td>
<td>Value of the default work state is Work in progress on the Task table. Value: 2</td>
</tr>
</tbody>
</table>

TaskStateUtil - SYSTEM_INACTIVE_STATES

The values of the default inactive states: Closed Complete, Closed Incomplete, Closed Skipped on the Task table.
### TaskStateUtil - TaskStateUtil(GlideRecord task)

Creates a TaskStateUtil object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>task</td>
<td>GlideRecord</td>
<td>This must be a GlideRecord from a task table.</td>
</tr>
</tbody>
</table>

```javascript
var stateUtil = new TaskStateUtil(current);
```

### Transformer - Scoped, Global

Manipulate time-series data to prepare the data for evaluation and analysis.

The Transformer class can be used in scoped and global server scripts. When using the Transformer class, use the `sn_clotho` namespace identifier.

The general use case is to determine the period to be evaluated, select the records from the table with the metric field, define the type of transform to run, and then execute the transform.

```javascript
// create the start and end time
var start = new GlideDateTime();
start.addSeconds(-1 * 60 * 60);
var end = new GlideDateTime();

//mb_demo_drone is a table with metric fields.
var drones = new GlideRecord("mb_demo_drone");
drones.addQuery("model", "Kingfisher Phantom");
drones.query();

//build a transform that returns a simple average
var builder = new sn_clotho.Transformer(drones);
builder.metric("mb_demo_mt_rem_battery").avg().label("Original");
```
This class is part of the MetricBase application.

**Scoped Transformer - `execute(GlideDateTime start, GlideDateTime end)`**

Run the transform.

Use the `metric()` and `groupBy()` methods before calling `execute()`. The `execute()` method can only be called once for each Transformer object.

Actions performed as part of the transform do not change the data in the MetricBase database.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>GlideDateTime</td>
<td>The beginning of the period to be evaluated.</td>
</tr>
<tr>
<td>end</td>
<td>GlideDateTime</td>
<td>The end of the period to be evaluated.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformResult</td>
<td>The transformed data.</td>
</tr>
</tbody>
</table>

```javascript
var minutesAgoStart = 60;
var end = new GlideDateTime();
var start = new GlideDateTime(end);
start.addSeconds(-1 * 60 * minutesAgoStart);

// query subject records
var grDrone = new GlideRecord('mb_demo_drone');
grDrone.query();

// building transform; get the average transforms of a metric, grouping by model
var transformer = new sn_clotho.Transformer(grDrone);
transformer.groupBy("fleet").metric("mb_demo_mt_altitude").avg().label('avg - %g:fleet:');

// execute and return result for visualization
var tfrmResult = transformer.execute(start, end);
```
Scoped Transformer - `groupBy(String field)`

Specify a field to be used to group the data.

If you are going to use the `groupBy()` method, it must be called before the `execute()` method.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>field</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify the transform characteristics.</td>
</tr>
</tbody>
</table>

```javascript
var transformer = new sn_clotho.Transformer(grDrone);
var trnsfrm = transformer.groupBy("fleet");
```

Scoped Transformer - `metric(String metricName)`

Specify the metric field to be used in the transform.

You can specify multiple metrics to be used in the transform. The `metric()` method cannot be called after the `execute()` method is called.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>metricName</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify the transform characteristics.</td>
</tr>
</tbody>
</table>

```javascript
var transformer = new sn_clotho.Transformer(grDrone);
var trnsfrm = transformer.metric("mb_demo_mt_altitude");
```
Scoped Transformer - Transformer(GlideRecord sourceRecords)

Create a Transformer object.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceRecords</td>
<td>GlideRecord</td>
<td>Contains the records for which metrics are to be evaluated. Can be one record or many.</td>
</tr>
</tbody>
</table>

```java
//where drones is a GlideRecord created from a table with a metric field
var builder = new sn_clotho.Transformer(drones);
```

TransformerDefinition - Scoped, Global

Provides methods to instantiate and manage the TransformerDefinition object, which associates a transformation rules list and a record path.

The TransformerDefinition API works along with the Transformer and TransformerRuleList APIs. Together, these APIs transform XML nodes or any entity in a structured JSON document into an output of name-value pairs. Supported JSON entities including objects and elements within an array, such as strings, numbers, and other arrays.

- The TransformerRuleList API enables you to create transformation rule lists that define what data in the source document to include in the output and how to transform the source data.

- The TransformerDefinition API associates a transformation rule list with a JSON/XML record path to define reusable transform definition objects. You can use a transform definition object to transform one or more source documents.

In addition, this object provides methods that enable you to obtain the record path of the TransformerDefinition object. It can also validate whether the definition has a valid TransformerRuleList associated with it.

The Transformer API performs the actual data transformation, one data entity at a time, using the specified transformation rule list to create the desired output data.

You can use the TransformerDefinition Class in both scoped and global server scripts. When using this class, use the `sn_tfrm` namespace identifier. Also, before this API is available in an instance, you must activate the Transformation Service plugin (com.glide.transform).
TransformerDefinition - recordPath()

If set when the TransformerDefinition object was instantiated, returns the record path for the source data file associated with the TransformerDefinition object.

A record path (JSONPath for JSON, XPath for XML) identifies the container (parent) of nodes to parse as records in the source document.

For JSON documents, these nodes can be any object, array, or JSON primitive (string, number, boolean). For XML documents, these nodes are the child elements of the container.

If recordPath is unspecified, the entire document is inferred as a single record. For JSON documents, if the top node is an array and recordPath is unspecified, then the top array is the record's container, and each of its entries is a record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>If set, the record path associated with the transformer definition. If not set, returns null.</td>
</tr>
</tbody>
</table>

```javascript
var path = '$.records';
var transformerDefinition = new sn_tfrm.TransformerDefinition(transformerRuleList, recordPath);
var transformer = new sn_tfrm.Transformer(transformerDefinition, responseBody);
```

```javascript
var recordPath = transformerDefinition.recordPath();
```

```javascript
var results = [];
while (transformer.transform()) {
    results.push(transformer.getRow());
}
```
TransformerDefinition - TransformerDefinition(Object transformerRuleList, String recordPath)

Instantiates a TransformerDefinition object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>transformerRuleList</td>
<td>Object</td>
<td>TransformerRuleList to associate with this transformer definition.</td>
</tr>
<tr>
<td>recordPath</td>
<td>String</td>
<td>Optional. The path (JSONPath for JSON, XPath for XML) that identifies the container (parent) of nodes to parse as records in the source document. For JSON documents, these nodes can be any object, array, or JSON primitive (string, number, boolean). For XML documents, these nodes are the child elements of the container. If recordPath is unspecified, the entire document is inferred as a single record. For JSON documents, if the top node is an array and recordPath is unspecified, then the top array is the record's container, and each of its entries is a record.</td>
</tr>
</tbody>
</table>

```javascript
var stockAPI = new sn_ws.RESTMessageV2('Stock Details', 'Default GET');
var response = stockAPI.execute();
var responseBody = response.getBody();

var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJSON()
  .addSymbol('ticker', 'quote.symbol')
  .addSymbol('change_percentage', 'quote.change_change')
  .thenMultiply('100')
  .thenRoundDown('0')
  .addSymbol('close_price', 'quote.close')
  .thenAdaptCurrency('USD', false)
  .addSymbol('summary')
  .thenConcat('Shares of ')  
  .thenConcatSymbol('ticker')
  .thenConcat(' closed at ')  
  .thenConcatSymbol('close');
```
var path = '$.*'; // Transform all objects in the JSON document
var transformerDefinition = new sn_tfrm.TransformerDefinition(transformerRuleList,
    recordPath);
var transformer = new sn_tfrm.Transformer(transformerDefinition, responseBody);

var results = [];
while (transformer.transform()) {
    results.push(transformer.getRow());
}

TransformerDefinition - validate()
Validates whether a valid TransformerRuleList is associated with the TransformerDefintion object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether a valid TransformerRuleList is associated with</td>
</tr>
<tr>
<td></td>
<td>the specified TransformerDefintion object.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Valid TransformerRuleList associated</td>
</tr>
<tr>
<td></td>
<td>• false: Invalid or no TransformerRuleList associated</td>
</tr>
</tbody>
</table>

if (transformerDefinition.validate()) {
    var results = [];

while (transformer.transform()) {
    results.push(transformer.getRow());
}

TransformerRuleList - Scoped, Global

Enables you to create a list of rules for transforming various types of JSON and XML data into name-value pair output.

The TransformerRuleList API works along with the Transformer and TransformerDefinition APIs. Together, these APIs transform XML nodes or any entity in a structured JSON document into an output of name-value pairs. Supported JSON entities including objects and elements within an array, such as strings, numbers, and other arrays.

- The TransformerRuleList API enables you to create transformation rule lists that define what data in the source document to include in the output and how to transform the source data.

- The TransformerDefinition API associates a transformation rule list with a JSON/XML record path to define reusable transform definition objects. You can use a transform definition object to transform one or more source documents.

- The Transformer API performs the actual data transformation, one data entity at a time, using the specified transformation rule list to create the desired output data.

When you instantiate the TransformerRuleList object, you must define whether it describes a JSON or XML source document using the .fromJSON() or .fromXML() methods. For example:

```javascript
var trl = sn_tfrm.TransformerRuleList().fromJSON(); or var trl = sn_tfrm.TransformerRuleList().fromXML();
```

Then use the addRule() method to define a transformation rule for each element in the source document that you want to include as a name-value pair in the output.

Use "adapter methods", such as setName() or thenAdd(), to define how to manipulate data elements when the transformation is performed. Adapter methods work along with the addRule() method. They define adaptations to apply to the output field defined by the addRule() method such as formatting, rounding, and applying patterns. You can apply one or more adapter methods to each data transformation. Adapter methods enable you to do things such as:
• Concatenate text.
• Perform mathematical functions, such as add, subtract, divide, and multiply.
• Round values up or down (round up/down, ceiling/floor).
• Define whether to use the minimum or maximum value when comparing a source data element to a specified value.
• Apply a currency code.
• Convert between different units of measure.
• Format data elements.
• Apply patterns.
• Replace specified data elements with a specified value.
• Split and rearrange strings.

You can use the TransformerRuleList class in both scoped and global server scripts. When using this class, use the sn_tfrm namespace identifier. Also, before this API is available in an instance, you must activate the Transformation Service plugin (com.glide.transform).

**TransformerRuleList - addRule(String rule, String path)**

Creates an entry in the associated transformation rules list which defines a field to create in the output.

You can create rules for any element in a source document. For example, `.addRule('ticker', '$.quote.symbol')` creates the field "ticker" in the output and copies over the value in quote.symbol of the source.

Once you define a rule using `addRule()`, you then use adaptor methods, such as `thenAdd()`, `thenReplace()`, and `thenFloor()` to manipulate the rule's output data. You can define as many adaptor methods as needed for a single rule. All adapter methods directly after an `addRule()` call, until the next `addRule()` call, apply to that rule. Adapter methods are cumulative with the result of all adapter methods being the final value saved in the output field.

For example, in the following code snippet, `thenMultiply()` and `thenRoundDown()` apply to `addRule('change_percentage', '$.quote.changePercent')`; `addRule('close_price', '$.quote.close')` starts a new rule. If the value in `$quote.changePercent` is .011, then the final output value is "1" (.011 * 100 rounded down to the ones position).
The `addRule()` method also supports the parameter `summary`: `addRule('summary')`. This implementation creates the name field "summary:" in the output, but does not correlate it to any field in the source. You can then use the adaptor methods `thenConcat()` and `thenConcatSymbol()` to modify the contents of the summary field.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rule</td>
<td>String</td>
<td>Name of the element in the destination output.</td>
</tr>
<tr>
<td>path</td>
<td>String</td>
<td>Optional. JSONPath or XPath to the data element in the source document.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The adapter methods <code>thenConcat()</code> and <code>thenConcatSymbol()</code> do not require you to define this parameter if no other adapter methods are defined for the rule. All other adapter methods require this parameter.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
    .fromJSON()
    .setName('Stock Report 05022019 JSON')
    .addRule('ticker', '$.quote.symbol')
    .addRule('change_percentage', '$.quote.changePercent')
    .thenMultiply('100')
    .thenRoundDown('0')
    .addRule('close_price', '$.quote.close')
    .thenAdaptCurrency('USD', false)
    .addRule('summary')
    .thenConcat('Shares of ')  
    .thenConcatSymbol('ticker')
    .thenConcat(' closed at ')  
    .thenConcatSymbol('close_price');
```
TransformerRuleList - setName(String name)

Defines a name for the associated TransformerRuleList object for logging purposes.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name to use to identify the associated TransformerRuleList object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Although the API does not force this name to be unique, it is helpful if</td>
</tr>
<tr>
<td></td>
<td></td>
<td>they are unique within an instance.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJSON()
  .setName('Stock Report 05022019 JSON')
  .addRule('ticker', '$.quote.symbol')
  .addRule('change_percentage', '$.quote.changePercent')
  .thenMultiply('100')
  .thenRoundDown('0')
  .addRule('close_price', '$.quote.close')
  .thenAdaptCurrency('USD', false)
  .addRule('summary')
  .thenConcat('Shares of ')  
  .thenConcatSymbol('ticker')
  .thenConcat(' closed at ')  
  .thenConcatSymbol('close_price');
```

TransformerRuleList - thenAdaptCurrency(String currencyCode, Boolean outputNumericCurrencyValue)

Adds an adapter to the current rule that ties the specified currency code to the output field defined in the associated addRule() call.
Note: This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>currencyCode</td>
<td>String</td>
<td>Currency code to tie to the destination data element, such as USD, EUR, and GBP.</td>
</tr>
</tbody>
</table>
| outputNumericCurrencyValue        | Boolean   | Optional. Flag that indicates whether to display the currency code. Valid values:
|                                  |           | • true: do not display the country code; numeric value only
|                                  |           | • false: display the country code
|                                  |           | Default: false                                                             |

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
    .fromJSON()
    .setName('Stock Report 05022019 JSON')
    .addRule('ticker', '$.quote.symbol')
    .addRule('change_percentage', '$.quote.changePercent')
    .thenMultiply('100')
    .thenRoundDown('0')
    .addRule('close_price', '$.quote.close')
    .thenAdaptCurrency('USD', false)
    .addRule('summary')
    .thenConcat('Shares of ')  
    .thenConcatSymbol('ticker')
    .thenConcat(' closed at ')  
    .thenConcatSymbol('close_price');
```
TransformerRuleList - thenAdaptDuration(String inputDuration, String outputDuration)

Adds an adapter to the current rule that converts the source field from one unit of measure to another, such as from minutes to seconds or weeks to days.

Note: This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the addRule() method. They define adaptations to apply to the output field defined by the addRule() method such as formatting, rounding, and applying patterns.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputDuration</td>
<td>String</td>
<td>Current unit of measure of the source field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NANosecond</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MICROsecond</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MILLisecond</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SECOND</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MINUTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• HOUR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DAY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• WEEK</td>
</tr>
<tr>
<td>outputDuration</td>
<td>String</td>
<td>Unit of measure to convert the source field to in the output.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NANosecond</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MICROsecond</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MILLisecond</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SECOND</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MINUTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• HOUR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DAY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• WEEK</td>
</tr>
</tbody>
</table>
### TransformerRuleList - thenAdd(Number operand)

Adds an adapter to the current rule that adds the passed in value to the source field.

⚠️ **Note:** This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operand</td>
<td>Number</td>
<td>Value to add to the source field.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
    .fromJSON()
    .setName('Stock Report 05022019 JSON')
    .addRule('ticker', '$.quote.symbol')
    .addRule('change_percentage', '$.quote.changePercent')
    .thenMultiply('100')
    .thenRoundDown('0')
    .addRule('close_price', '$.quote.close')
    .thenAdaptCurrency('USD', false)
    .addRule('daily', '$.quote.weekly')
    .thenAdaptDuration('WEEK', 'DAY');
```

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList().fromJSON()
    .setName('Stock Report 05182019 JSON')
    .addSymbol('ticker', '$.symbol')
    .addSymbol('change_percentage', '$.changePercent')
```
TransformerRuleList - thenApplyMap(Object map)

Adds an adapter to the current rule that searches the associated source field for a list of strings and replaces them in the output with the specified replacement values.

Note: This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the addRule() method. They define adaptations to apply to the output field defined by the addRule() method such as formatting, rounding, and applying patterns.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>map</td>
<td>Object</td>
<td>Name/value pairs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• name: text to locate in the source data element</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• value: text to replace it with in the output</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

In this example, the thenApplyMap() call searches the source text in the company_info field for Inc. and St. and replaces them with Incorporated and Street in the company_info field in the output.

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
    .fromJSON()
    .setName('Stock Report 05022019 JSON')
    .addRule('ticker', '$.quote.symbol')
    .addRule('change_percentage', '$.quote.changePercent')
    .thenMultiply('100')
    .thenRoundDown('0')
    .addRule('close_price', '$.quote.close')
```
TransformerRuleList - thenApplyPattern(String matchPattern, String outputPattern)

Adds an adapter to the current rule that matches a specified regex pattern to content in the source field and then replaces/reformats that content with a second regex pattern and stores that value in the output.

⚠️ Note: This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the addRule() method. They define adaptations to apply to the output field defined by the addRule() method such as formatting, rounding, and applying patterns.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>matchPattern</td>
<td>String</td>
<td>Regex pattern to use to locate the text to replace/reformat.</td>
</tr>
<tr>
<td>outputPattern</td>
<td>String</td>
<td>Regex pattern to use to update the located text.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

In this example, any company name that ends with "Inc." is transformed to "Incorporated", such as Company Inc. to Company Incorporated.

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
    .fromJSON()
    .setName('Stock Report 05022019 JSON')
    .addRule('ticker', '$.quote.symbol')
    .addRule('change_percentage', '$.quote.changePercent')
    .thenMultiply('100')
    .thenRoundDown('0')
    .addRule('close_price', '$.quote.close')
    .thenAdaptCurrency('USD', false)
    .addRule('company', '$.quote.company')
    .thenApplyPattern('(.+)(\w{3}\.)', '$1Incorporated');
```
TransformerRuleList - thenCeiling(Number operand)

Adds an action to the current rule to round the source field up at the decimal position specified by the passed in value.

Unlike straight rounding where the number is rounded based on the value of the digit in the specified decimal position (0-4 round down, 5-9 round up), ceiling always rounds up. For example, the ceiling value for 2.156 and 2.152 is always 2.16 for the passed in decimal position of 2; whereas for straight rounding the values would be 2.16 and 2.15 respectively. For negative numbers, the ceiling operation makes the number more positive, as in the ceiling value of -2.156 is -2.15; whereas the same number rounded up makes the value more negative, -2.16.

Note: This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operand</td>
<td>Number</td>
<td>Decimal position to the right of the decimal point at which to round the number up (apply ceiling). For example, if this value is 2 and the source data element is 6.421, the resulting value is 6.43.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
    .fromJSON()
    .setName('Mortgage Rates 05022019 JSON')
    .addRule('interest_rate', '$.quote.interest_rate')
    .thenCeiling('2') // Always round the interest rate up to the hundredths place
    .addRule('monthly_payment', '$.quote.total_loan_amount')
    .thenDivideBy('180');```
TransformerRuleList - thenConcat(String value)

Adds an adapter to the current rule that concatenates the passed in string to the output field.

Unlike other adapter methods, this method does not have to be applied to a source field. You can use this method to create completely new data strings in the output.

**Note:** This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>Text to concatenate to the end of the current data element.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJSON()
  .setName('Stock Report 05022019 JSON')
  .addRule('ticker', '$.quote.symbol')
  .addRule('change_percentage', '$.quote.changePercent')
  .thenMultiply('100')
  .thenRoundDown('0')
  .addRule('close_price', '$.quote.close')
  .thenAdaptCurrency('USD', false)
  .addRule('summary')
  .thenConcat('Shares of ')  
  .thenConcatSymbol('ticker')
  .thenConcat(' closed at ')  
  .thenConcatSymbol('close_price');
```

TransformerRuleList - thenConcatSymbol(String symbol)

Adds an adapter to the current rule that concatenates the value of a field previously defined in the rules list to the current output field.
Note: This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>symbol</td>
<td>String</td>
<td>Name of the rules list element to append to the output field.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJSON()
  .setName('Stock Report 05022019 JSON')
  .addRule('ticker', '$.quote.symbol')
  .addRule('change_percentage', '$.quote.changePercent')
  .thenMultiply('100')
  .thenRoundDown('0')
  .addRule('close_price', '$.quote.close')
  .thenAdaptCurrency('USD', false)
  .addRule('summary')
  .thenConcat('Shares of ')  
  .thenConcatSymbol('ticker')
  .thenConcat(' closed at ')  
  .thenConcatSymbol('close_price');
```

**TransformerRuleList - thenDivideBy(Number operand)**

Adds an adapter to the current rule that divides the source field by the passed in value.

Note: This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.
TransformerRuleList - thenDivideInto(Number operand)

Adds an adapter to the current rule that divides the passed in value by the source field.

Note: This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operand</td>
<td>Number</td>
<td>Value into which to divide the source field.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>
var transformerRuleList = new sn_tfrm.TransformerRuleList()
    .fromJSON()
    .setName('Stock Report 05022019 JSON')
    .addRule('ticker', '$.quote.symbol')
    .addRule('change_percentage', '$.quote.changePercent')
    .thenMultiply('100')
    .thenRoundDown('0')
    .addRule('dividend_per_share', '$.quote.total_shares')
    .thenDivideInto('$.quote.total_dividends');

TransformerRuleList - thenFloor(Number operand)

Adds an adapter to the current rule that rounds the source field down at the
decimal position specified by the passed in value and stores it in the output.

Unlike straight rounding where the number is rounded based on the value of
the digit in the specified decimal position (0-4 round down, 5-9 round up), floor
always rounds down. For example, the floor value for 2.156 and 2.152 is always
2.15 for the passed in decimal position of 2; whereas for straight rounding the
values would be 2.16 and 2.15 respectively. For negative numbers, the floor
operation makes the number more negative, as in the floor value of -2.156 is
-2.16; whereas the same number rounded down makes the value more positive,
-2.15.

ℹ️ Note: This is an adapter method and cannot be used on its own. Adapter
methods work in conjunction with the addRule() method. They define
adaptations to apply to the output field defined by the addRule() method
such as formatting, rounding, and applying patterns.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operand</td>
<td>Number</td>
<td>Decimal position to the right of the decimal point at which to round the number down (apply floor).</td>
</tr>
</tbody>
</table>

For example, if this value is 2 and the source data element is 6.427, the resulting value is 6.42.

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>
var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJSON()
  .setName('Mortgage Rates 05022019 JSON')
  .addRule('interest_rate', '$.quote.interest_rate')
  .thenFloor('2') // Always round the interest rate down to the hundreths place
  .addRule('monthly_payment', '$.quote.total_loan_amount')
  .thenDivideBy('180');

TransformerRuleList - thenFormat(String matchPattern, String outputPattern)

Adds an adapter to the current rule that reformats the content in the source field that matches the specified match pattern, with the specified output pattern.

**Note:** This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>matchPattern</td>
<td>String</td>
<td>Pattern to match against the source field. Valid values: • @: any character • #: decimal digit • : literal escape (@ for the @ symbol)</td>
</tr>
<tr>
<td>outputPattern</td>
<td>String</td>
<td>Pattern to replace the content with in the output. Valid values: • @: any character • #: decimal digit • : literal escape (@ for the @ symbol)</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>
This example reformats the quote date from YYYY/MM/DD to YYYY MM-DD.

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJSON()
  .setName('Stock Report 05022019 JSON')
  .addRule('ticker', '$.quote.symbol')
  .addRule('change_percentage', '$.quote.changePercent')
  .thenMultiply('100')
  .thenRoundDown('0')
  .addRule('close_price', '$.quote.close')
  .thenAdaptCurrency('USD', false)
  .addRule('date', '$.quote.date')
  .thenFormat('####/##/##', '#### ##-##');
```

### TransformerRuleList - thenMax(Number operand)

Adds an adapter to the current rule that compares the passed in value against the source field and copies the greater of the two values to the output field.

⚠️ **Note:** This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operand</td>
<td>Number</td>
<td>Value to compare to the source field.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJSON()
  .setName('Mortgage Rates 05022019 JSON')
  .addRule('lowest_interest_rate', '$.quote.interest_rate')
  .thenMax('3.5') // Interest rate cannot be less than 3.5%
  .addRule('dividend_per_share', '$.quote.total_shares')
  .thenDivideInto('$quote.total_dividends');
```
TransformerRuleList - thenMin(Number operand)

Adds an adapter to the current rule that compares the passed in value against the source field and copies the lower of the two values to the output field.

ℹ️ **Note:** This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operand</td>
<td>Number</td>
<td>Value to compare to the source field.</td>
</tr>
</tbody>
</table>

TransformRuleList - thenMultiply(Number operand)

Adds an adapter to the current rule that multiplies the source field by the passed in value.

ℹ️ **Note:** This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operand</td>
<td>Number</td>
<td>Value by which to multiply the source field.</td>
</tr>
</tbody>
</table>

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJSON()
  .setName('Stock Report 05022019 JSON')
  .addSymbol('broker_fee_percentage', '$.quote.broker_fee')
  .thenMin('10') // Maximum of 10% broker fee
  .addSymbol('dividend_per_share', '$.quote.total_shares')
  .thenDivideInto('$quote.total_dividends');
```
**TransformerRuleList - thenReplace(String matchString, String replaceString)**

Adds an adapter to the current rule that finds all text within the source field that matches a specified string and replaces it with an updated string.

ℹ️ **Note:** This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>matchString</td>
<td>String</td>
<td>String to match against the source field to identify the text to replace.</td>
</tr>
<tr>
<td>replaceString</td>
<td>String</td>
<td>String with which to replace the matching text.</td>
</tr>
</tbody>
</table>

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJSON()
  .setName('Stock Report 05022019 JSON')
  .addRule('ticker', '$.quote.symbol')
  .addRule('change_percentage', '$.quote.changePercent')
  .thenMultiply('100')
  .thenRoundDown('0')
  .addRule('close_price', '$.quote.close')
  .thenAdaptCurrency('USD', false)
  .addRule('summary')
  .thenConcat('Shares of ')
  .thenConcatSymbol('ticker')
  .thenConcat(' closed at ')
  .thenConcatSymbol('close_price');
```
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

In this example, any company name that ends with "Inc." is transformed to "Incorporated", such as Company Inc. to Company Incorporated.

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJSON()
  .setName('Stock Report 05022019 JSON')
  .addRule('ticker', '$.quote.symbol')
  .addRule('change_percentage', '$.quote.changePercent')
  .thenMultiply('100')
  .thenRoundDown('0')
  .addRule('close_price', '$.quote.close')
  .thenAdaptCurrency('USD', false)
  .addRule('company', '$.quote.company')
  .thenReplace('Inc.', 'Incorporated');
```

TransformerRuleList - thenRoundDown(Number operand)

Adds an adapter to the current rule that rounds the source field down at the decimal position specified by the passed in value.

For negative numbers, the thenRoundDown() method makes the number more positive; the round down value of -2.156 is -2.15. The thenFloor() method makes the same number more negative, -2.16.

**Note:** This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the addRule() method. They define adaptations to apply to the output field defined by the addRule() method such as formatting, rounding, and applying patterns.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operand</td>
<td>Number</td>
<td>Decimal position to the right of the decimal point at which to round the number down. For example, if this value is 2 and the source data element is 6.427, the resulting value is 6.42.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList90
  .fromJSON()
  .setName('Mortgage Rates 05022019 JSON')
  .addRule('interest_rate', '$.quote.interest_rate')
  .thenRoundDown('2') // Always round the interest rate down to the hundredths place
  .addRule('monthly_payment', '$.quote.total_loan_amount')
  .thenDivideBy('180');
```

### TransformerRuleList - thenRound Up(Number operand)

Adds an adapter to the current rule that rounds the source data element up at the decimal position specified by the passed in value.

For negative numbers, the `thenRoundUp()` method makes the number more negative; the round up value of -2.156 is -2.16. The `thenCeiling()` method makes the same number more positive, -2.15.

**Note:** This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operand</td>
<td>Number</td>
<td>Decimal position to the right of the decimal point at which to round the number up.</td>
</tr>
</tbody>
</table>

  For example, if this value is 2 and the source data element is 6.422, the resulting value is 6.43.

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>
```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJSON()
  .setName('Mortgage Rates 05022019 JSON')
  .addRule('interest_rate', '$.quote.interest_rate')
  .thenRoundUp('2') // Always round the interest rate up to the hundredths place
  .addRule('monthly_payment', '$.quote.total_loan_amount')
  .thenDivideBy('180');
```

**TransformerRuleList - thenSplit(String splitPattern, String replaceString)**

Adds an adapter to the current rule that splits, reorganizes, and replaces strings within the source field and saves them in the output field.

The `splitPattern` parameter denotes how to divide the source text string into segments. This parameter can be a typical word separator such as a space (" "), comma (','), or semicolon (';'), and can also be any string, such as 'name'. For example, if the source text is "Smith John Michael" and the `splitPattern` is a space, then the available segments are "Smith", "John", and "Michael". To reference a segment, use $#, where # is the number of the order of the segment in the source text string. For example, if the method call is `thenSplit(' ', '$2 $3 $1')`, then the output is "John Michael Smith". In addition, you can add constants within the output pattern, such as `thenSplit(' ', 'First name: $2 Middle name: $3 Last name: $1')`.

**Note:** This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>splitPattern</td>
<td>String</td>
<td>String that denotes how to split the text into segments.</td>
</tr>
<tr>
<td>replaceString</td>
<td>String</td>
<td>String that defines the output string, including segment references and constants. Not all segments need to be referenced. Reference the segments defined by the <code>splitPattern</code> using $#, where # is the number of the order of the segment in the source data element. Reference the entire source data element using $0.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

In this example, the CEO name is last name, first name in the source data and in the output it will be transformed to first name last name.

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJson()
  .setName('Stock Report 05022019 JSON')
  .addRule('ticker', '$.quote.symbol')
  .addRule('change_percentage', '$.quote.changePercent')
  .thenMultiply('100')
  .thenRoundDown('0')
  .addRule('close_price', '$.quote.close')
  .thenAdaptCurrency('USD', false)
  .addRule('CEO', '$.quote.CEO')
  .thenSplit(',', '$2 $1');
```

TransformerRuleList - `thenSubtract(Number operand)`

Adds an adapter to the current rule that subtracts the passed in value from the source field and stores it in the output field.

**Note:** This is an adapter method and cannot be used on its own. Adapter methods work in conjunction with the `addRule()` method. They define adaptations to apply to the output field defined by the `addRule()` method such as formatting, rounding, and applying patterns.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operand</td>
<td>Number</td>
<td>Value to subtract from the source data element.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Current TransformerRuleList object.</td>
</tr>
</tbody>
</table>

```javascript
var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJson();
```
.setName('Stock 05182019 JSON')
.addRule('ticker', '$.quote.symbol')
.addRule('change_percentage', '$.quote.changePercent')
.thenMultiply('100')
.thenRoundDown('0')
.addRule('close_price', '$.quote.close')
.thenAdaptCurrency('USD', false)
.addRule('trade_price', '$.quote.current_price')
.addMultiply('$quote.trade_quantity')
.thenSubtract('$quote.discount_dollars');

Transformer - Scoped, Global

Parses and transforms structured JSON or XML source content into structured key/value pair output, using a defined set of rules.

The scripted Transformer API works along with the TransformerDefinition and TransformerRuleList APIs. Together, these APIs transform XML nodes or any entity in a structured JSON document into an output of name-value pairs. Supported JSON entities including objects and elements within an array, such as strings, numbers, and other arrays.

- The TransformerRuleList API enables you to create transformation rule lists that define what data in the source document to include in the output and how to transform the source data.

- The TransformerDefinition API associates a transformation rule list with a JSON/XML record path to define reusable transform definition objects. You can use a transform definition object to transform one or more source documents.

- The Transformer API performs the actual data transformation, one data entity at a time, using the specified transformation rule list to create the desired output data.

You can use the Transformer class in both scoped and global server scripts. When using this class, use the sn_tfrm namespace identifier. Before this API is available in an instance, you must activate the Transformation Service plugin (com.glide.transform).

The following example illustrates how to use the TransformerRuleList API to define the transformation rules, the TransformerDefinition API to define the transformation criteria, and the Transformer API to actually perform the transformation.
This code example retrieves an external JSON-based stock details document, creates rules for transforming that data into a tabular table, and then transforms the source document, one row at a time. The following is a snippet of the JSON source document being transformed:

```
{
  "NOW": {
    "quote": {
      "symbol": "NOW",
      "companyName": "ServiceNow Inc.",
      "primaryExchange": "New York Stock Exchange",
      "sector": "Technology",
      "open": 166.78,
      "openTime": 1522935000556,
      "close": 165.77,
      "changePercent": 0.00656,
      ...
    },
    ...
  }
}
```

```javascript
var stockAPI = new sn_ws.RESTMessageV2('Stock Details', 'Default GET');
var response = stockAPI.execute();
var responseBody = response.getBody(); // obtain the source JSON document

/* Define the list of rules to use to transform the acquired JSON stock detail information into a tabular table */

var transformerRuleList = new sn_tfrm.TransformerRuleList() // instantiate the rule list object .fromJSON() // indicate that the source document is JSON .addRule('ticker', '$.quote.symbol') // add a rule to copy the value in the "symbol" field of the source document to the ticker field in the output document (no changes) .addRule('change_percentage', '$.quote.changePercent') // copy the "changePercent" field from source into the change_percentage field of output document .thenMultiply('100') // multiply the change_percentage value by 100 .thenRoundDown('0') // add then round it down to a whole number .addRule('close_price', '$.quote.close') // copy the "close" field to the close_price field in the 21--[';output .thenAdaptCurrency('USD', false) // attach the US dollar code to the close_price field, but do not display the symbol .addRule('summary') // add a blank "summary" field to the output (no corresponding source field) .thenConcat('Shares of ') // in the summary field concatenate the string "Shares of "
```
.thenConcatSymbol('ticker') // then concatenate the "ticker" field from the source document
.thenConcat(' closed at ') // then concatenate the string " closed at "
.thenConcatSymbol('close_price'); // then concatenate the "close_price" field from the source document

// Create a transformer definition that associates the rule list to use and the record path of the set of records in the source document to transform.
var path = '$.*';
var transformerDefinition = new sn_tfrm.TransformerDefinition(transformerRuleList, path);

// Instantiate the transformer object.
var transformer = new sn_tfrm.Transformer(transformerDefinition, responseBody);

// Transform the source data, one row at a time, until all rows are processed.
var results = [];
while (transformer.transform()) {
    results.push(transformer.getRow());
}

Output:

```json
{ticker: "Now", change_percentage: "0", close_price: "165.77 USD", summary: "Shares of Now closed at 165.77" }
```

**Transformer - getRow()**

Returns the row resulting from the last transformation (or null if no row exists).

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Object</td>
</tr>
</tbody>
</table>

```javascript
var results = [];
while (transformer.transform()) {
    results.push(transformer.getRow());
}:
```
Output:

```javascript
var results = [];
while (transformer.transform()) {
    results.push(transformer.getRow());
}
```

Transformer - `transform()`

Transforms the next available row/node in the source document.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether there is a valid next row to transform.</td>
</tr>
<tr>
<td></td>
<td>• true: Valid next row</td>
</tr>
<tr>
<td></td>
<td>• false: No additional rows</td>
</tr>
</tbody>
</table>

Transformer - `Transformer(Object transformerDefinition, String document)`

Instantiates a Transformer object (constructor).

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>transformerDefinition</td>
<td>Object</td>
<td>Object that describes the content transformation; includes rule list and JSONPath/XPath record path.</td>
</tr>
</tbody>
</table>
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>document</td>
<td>String</td>
<td>Source document to translate.</td>
</tr>
</tbody>
</table>

### Example

```javascript
var stockAPI = new sn_ws.RESTMessageV2('Stock Details', 'Default GET');
var response = stockAPI.execute();
var responseBody = response.getBody();

var transformerRuleList = new sn_tfrm.TransformerRuleList()
  .fromJSON()
  .addRule('ticker', 'quote.symbol')
  .addRule('change_percentage', 'quote.change')
  .thenMultiply('100')
  .thenRoundDown('0')
  .addRule('close_price', 'quote.close')
  .thenAdaptCurrency('USD', false)
  .addRule('summary')
  .thenConcat('Shares of ')  
  .thenConcatSymbol('ticker')
  .thenConcat(' closed at ')  
  .thenConcatSymbol('close')

var path = '$.*';
var transformerDefinition = new sn_tfrm.TransformerDefinition(transformerRuleList, path);
var transformer = new sn_tfrm.Transformer(transformerDefinition, responseBody);

var results = [];
while (transformer.transform()) {
  results.push(transformer.getRow());
}
```

### TransformPart - Scoped, Global

Use the `TransformPart` class to specify details of the transform to be done.

The `TransformPart` class can be used in scoped and global server scripts. When using the `TransformPart` class, use the `sn_clotho` namespace identifier.
There is no constructor for this class. TransformPart objects are returned by many Transformer and TransformPart methods.

The methods of this class define the transforms to be done. The actual transformation is done when the execute() method is called on the Transformer object.

The order the TransformPart methods are called is important.

• You must call the metric() method before calling a transform method.
• You cannot use the metric() or groupBy() methods after calling a transform method.
• Intermediate transforms are not returned in a result unless the collect() method is called for the intermediate result you want.

```java
// where tp is a TransformPart object
// example 1
tp.avg().add(2);
var tr = tp.execute();
// tr contains avg+2, but not avg

// example 2
tp.avg().add(2);
tp.avg();
var tr = tp.execute();
// tr contains both avg and avg + 2

// example 3
tp.avg().collect().add(2);
var tr = tp.execute();
// tr contains both avg and avg + 2
```

This class is part of the MetricBase application.

**TransformPart - add(Number constant)**

Add the specified number to the value in each time stamp.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>constant</td>
<td>Number</td>
<td>The number to add to the value in each time stamp.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformPart - avg()**

Aggregate the selected metric series into one series containing the average value for each time stamp.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformPart - bottom(Number count)**

Create a result set that for each time stamp returns specified number of bottom values. This method results in 'count' number of series. Each value retains the label of its source series.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>Number</td>
<td>The number of series to return. The series are labeled 0 to count - 1.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>
**TransformPart - ceil(Number ceiling)**

Replace the value in any time stamp that is greater than the specified value with the specified value.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ceiling</td>
<td>Number</td>
<td>The maximum allowed value for any time stamp.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformPart - collect()**

Mark this transform for collection.

Transforms that are part of a chain, but not the last transform, are by default not collected. A collected transform is returned as part of the transform result.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformPart - count()**

Aggregate the selected metric series into one series containing the number of values for each time stamp.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformPart - div(Number constant)**

Divide the value in each time stamp by the specified number.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>constant</td>
<td>Number</td>
<td>The number by which to divide the value of each time stamp.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformPart - filter(Object aggregator, Object duration)**

Create a series using the specified aggregator for the specified time.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aggregator</td>
<td>Object</td>
<td>Can be:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AVG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CHISQUARE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• LAST</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• MAX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• MEDIAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• MIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• STDDEV</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For definitions of these options, see MetricBase transforms.

<table>
<thead>
<tr>
<th>duration</th>
<th>Object</th>
<th>Time period for the series.</th>
</tr>
</thead>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>TransformPart object that you can use to specify transform characteristics.</td>
</tr>
</tbody>
</table>

#### TransformPart - floor(Number floor)

Replace the value in any time stamp that is less than the specified value with the specified value.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>floor</td>
<td>Number</td>
<td>The minimum value for any time stamp.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

#### TransformPart - fractiles(Array fractions)

Create series made up of the value that the specified percentage of values is below. Returns a series for each fraction in the specified array.
The value in a time stamp in a returned series is the value at which the specified fraction of the samples for that time stamp is below. For example, if the fraction is 0.5, then the value in the time stamp is the value where half the values in the input series are below (median).

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fractions</td>
<td>Array of numbers</td>
<td>The fractions to use on the input series.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics. Contains one series for each fraction specified.</td>
</tr>
</tbody>
</table>

```java
// returns a single series containing the median for each time stamp, which means that half a time stamp's values are below the returned value
fractiles([.5])
// returns four series, one series for each of the 25%, 50%, 75%, and 100% quartiles
fractiles([.25, .5, .75, 1])
// returns the median, 95% percentile, the max value
fractiles([.50, .95, 1])
```

### TransformPart - getResult()

Return the part of the result relevant to this transform.

The `collect()` method must be called before the `execute()` method, and the `execute()` method must be called before calling the `getResult()` method.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformResult</td>
<td>Contains the transform results associated with this part of the transform.</td>
</tr>
</tbody>
</table>

```javascript
var t = new sn_clotho.Transformer(drones);
t.metric("mb_demo_mt_altitude");
var avgTform = t.avg();
t.execute();
var avgTformResult = avgTform.getResult();
```

**TransformPart - `groupBy(String field)`**

Specify a field to be used to group the data.

The `groupBy()` method cannot be called after a transform has been run.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>A field in the table to be used to group the transform results.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformPart - `interpolate(Object count)`**

Create a data value for a NaN data item by interpolating from adjacent data values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>Object</td>
<td>Specifies the number of data samples in each direction to check for a non NaN value. If a non NaN value is not found, NaN is used.</td>
</tr>
</tbody>
</table>
### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

### TransformPart - iqr()

Perform an Interquartile range transform.

Creates a result set of four series.

- IQR, the median of all entries
- IQR range, below Q1-1.5IQR, or above Q3+1.5IQR
- Q1, the median of the smallest half of entries
- Q3, the median of the largest half of entries

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

### TransformPart - label(String label)

Add a label for the resulting series.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>String</td>
<td>The label for the transform results.</td>
</tr>
<tr>
<td>Returns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------</td>
<td></td>
</tr>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
<td></td>
</tr>
</tbody>
</table>

**TransformPart - limit(Object count)**

Returns at most the specified number of values, starting at the most recent non-NaN value.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>Object</td>
<td>A number of time stamps.</td>
</tr>
</tbody>
</table>

<p>| Returns |
|---------|----------------------------------|</p>
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformPart - log(Number base)**

Run a logarithm on the value in each time stamp where the result is the log of the specified base for the time stamp value.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>base</td>
<td>Number</td>
<td>The base for the logarithm calculation.</td>
</tr>
</tbody>
</table>

<p>| Returns |
|---------|----------------------------------|</p>
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>
TransformPart - max()

Returns a series with the maximum value for each time stamp.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

TransformPart - median()

Create a series containing the median of values for each time stamp across a set of series.

If there are n series in the TransformPart object, then if n is odd, the \((n / 2 + 1)\) value for a time stamp is the median. If n is even, the average of the \((n / 2)\) and \((n / 2 + 1)\) values for a time stamp is the median.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

TransformPart - metric(String metric)

Specify the metric field to be used in the transform.

You can specify multiple metrics to be used in the transform. The \(\text{metric()}\) method cannot be called after the transform has been run.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metric</td>
<td>String</td>
<td>Name of the metric field.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

### TransformPart - `min()`

Returns a series with the minimum value for each time stamp.

### TransformPart - `mul(Number constant)`

Multiply the value in each time stamp by the specified number.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>constant</td>
<td>Number</td>
<td>The number by which to multiply the value of each time stamp.</td>
</tr>
</tbody>
</table>
TransformPart - partition(String aggregator, GlideDateTime duration, GlideDateTime base)
Partition the series into intervals of the same duration.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aggregator</td>
<td>String</td>
<td>The aggregator to use. Can be min, max, avg, or last.</td>
</tr>
<tr>
<td>duration</td>
<td>GlideDateTime or an ISO 8601 formatted string</td>
<td>The interval length.</td>
</tr>
<tr>
<td>base</td>
<td>GlideDateTime or an ISO 8601 formatted string</td>
<td>The zero offset for partitioning. For example, if you partition by day (24h), then set the base to Monday at midnight in your time zone. If you partition by 30 days, then set the base to 1st day of the most recent month.</td>
</tr>
</tbody>
</table>

TransformPart - resample(Number count)
Specify the number of data points to include in the result.

Aligns a series with a fixed number of data points in the given range. If the original series has more data points than specified, multiple values are averaged. If the original series has fewer data points than specified, data points are added by interpolating data points between existing data points.
You can use the `resample()` method to reduce the number of samples in the result to more closely match the number of samples you are going to display.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>Number</td>
<td>The number of samples to include in the result.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformPart - resample(Number min, Number max)**

Specify the minimum and maximum number of samples to include in the result.

This method is useful when you are showing series with different time periods (granularities).

This method determines the average number of points per series and if that fits between the specified minimum and maximum, each series is resampled to that average number of points. If the calculated average is greater than the maximum specified or smaller than the minimum specified, the specified maximum or minimum is used.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>min</td>
<td>Number</td>
<td>The minimum number of samples to include in the result. If not enough samples are available, interpolation is used to create samples.</td>
</tr>
<tr>
<td>max</td>
<td>Number</td>
<td>The maximum number of samples to include in the result.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>
**TransformPart - resample(String aggregator, Number numValues)**

Specify an aggregator to use to create a result set of the specified size. The aggregator can be LAST, AVG, MIN, or MAX.

You can use the `resample()` method to reduce the number of samples in the result to more closely match the number of samples you are going to display.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aggregator</td>
<td>String</td>
<td>Can be LAST, AVG, MIN, or MAX.</td>
</tr>
<tr>
<td>numValues</td>
<td>Number</td>
<td>The number of samples to include in the result set. When the number of values requested is greater than the number of values in the data for the requested time period, <code>interpolate()</code> is used to add values between existing points to reach the requested number of values.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformPart - resample(String aggregator, GlideDuration duration)**

Specify an aggregator to use to create a result set over the specified duration. The aggregator can be LAST, AVG, MIN, or MAX.

Resamples the underlying data to the requested period.

- When the time series is stored at a 1 minute interval and `resample(AVG, new GlideDuration("1:00"))` is called, the result will have one data point at every hour reflecting the average of the previous 60 values.
- When the period requested is smaller than the period of the data, interpolation is used to calculate the intervening data points.

You can use the `resample()` method to reduce the number of samples in the result to more closely match the number of samples you are going to display.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aggregator</td>
<td>String</td>
<td>Can be LAST, AVG, MIN, or MAX.</td>
</tr>
<tr>
<td>duration</td>
<td>GlideDuration</td>
<td>The time period for the result set.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformPart - round(Number precision)**

Round the value in each time stamp to the specified precision.

Performs this calculation on each value.

\[(v / \text{precision}) \times \text{precision}\]

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>precision</td>
<td>Number</td>
<td>The value to be used in the rounding calculation.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformPart - stddev()**

Create a series containing the standard deviation of values for each time stamp across a set of series.
TransformPart - sub(Object constant)
Subtract the specified number from the value in each time stamp.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>constant</td>
<td>Object</td>
<td>The number to subtract from the value in each time stamp.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

TransformPart - sum()
Aggregate the selected metric series into one series containing the sum of all values for each time stamp.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformPart - top(Number count)**

Create a result set that for each time stamp returns the specified number of top values. This method results in 'count' number of series. Each value retains the label of its source series.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>Number</td>
<td>The number of series to return. The series are labeled 0 to count - 1.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransformPart</td>
<td>A TransformPart object that can be used to specify transform characteristics.</td>
</tr>
</tbody>
</table>

**TransformResult - Scoped, Global**

Provides the result of a transformation run on time-series data.

The TransformResult class can be used in scoped and global server scripts. When using the Transformer class, use the `sn_clotho` namespace identifier.

There is no constructor for this class. TransformResult objects are returned by many TransformPart methods.

This class is part of the MetricBase application.

**TransformResult - byGroup()**

Returns an array of Data objects. Returns an error if no group was specified for the transform.
**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>An array of Data objects, with each object corresponding to a group.</td>
</tr>
</tbody>
</table>

**TransformResult - getByLabel(String label)**

Returns the transformed data with the specified label.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>String</td>
<td>The label that identifies the data to be retrieved.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>The Data object with the transform results.</td>
</tr>
</tbody>
</table>

**TransformResult - getData()**

Returns a single Data object, or null if the result is empty.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>The Data object with the transform results.</td>
</tr>
</tbody>
</table>
TransformResult - toArray()

Returns the transformed data as an array. This method turns a Data object into an array.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>The Data object formatted as an array.</td>
</tr>
</tbody>
</table>

UniversalRequestUtilsSNC API - Scoped

Enables handling universal request lifecycles. Provided as a script include record.

The UniversalRequestUtilsSNC API requires the Universal Request (com.snc.universal_request) plugin and is provided within the sn_uni_req namespace.

For information, refer to Universal Request.

UniversalRequestUtilsSNC - createUniversalRequest(Object copyFields)

Creates a universal request and returns the sys_id of the newly created universal request record.

This method is called using the UniversalRequestUtils prototype in the sn_uni_req namespace. For example, `sn_uni_req.UniversalRequestUtils().createUniversalRequest(<copyFields>)`.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>copyFields</td>
<td>Object</td>
<td>JSON object containing field names and values to set on a new record in the Universal Requests [universal_request] table. Provide details by field name and field value in the format <code>{ '&lt;field_name&gt;': '&lt;field_value&gt;' }</code>.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>The following fields are considered the most useful details to assign to a primary ticket.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'assignment_group': 'String',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'assigned_to': 'String',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'contact_type': 'String',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'description': 'String',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'opened_by': 'String',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'opened_for': 'String',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'priority': 'String',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'restricted': Boolean,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'short_description': 'String'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Other fields not listed might be eligible for your use case. For a list of table fields and related fields in the system, view the data dictionary.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>🔄 Note: Do not include the primary_ticket or sys_id fields in the copyFields JSON object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: Default assignment group is set per assignment rules.</td>
</tr>
<tr>
<td>copyFields.contact_type</td>
<td>String</td>
<td>Optional. Method by which the resource was initially reported. Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• chat</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• email</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• phone</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• social</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• web</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Maximum length: 40</td>
</tr>
</tbody>
</table>
## Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>copyFields.description</td>
<td>String</td>
<td>Optional. Detailed description of the problem associated with the resource. Maximum length: 4,000</td>
</tr>
<tr>
<td>copyFields.opened_by</td>
<td>String</td>
<td>Optional. Sys_id of the person that initially opened the resource. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td>copyFields.opened_for</td>
<td>String</td>
<td>Optional. Required for HR Service Delivery cases. The sys_id of the user for whom the resource was opened.</td>
</tr>
</tbody>
</table>
| copyFields.priority | String | Optional. Priority of the resource. Specified as a number. Possible values:  
  - 1: Critical  
  - 2: High  
  - 3: Moderate  
  - 4: Low  
  Default: 3 |
| copyFields.restricted | Boolean | Optional. Flag that indicates if the universal request has restricted access. Refer to Universal Request roles and groups. Valid values:  
  - true: Universal request is only accessible to users with the sn_uni_req.sensitiveinfo_agent role. Users with this role have permissions to view sensitive information, such as payroll details.  
  - false: Universal request access is unrestricted.  
  Default: false |
| copyFields.short_description | String | Concise description of the resource. Maximum length: 160 |
The following example shows how to set fields on a new universal request. In this use case, this code is added to the beginning of the Inbound Email Actions [sysevent_in_email_action] scoped Create HR Case script. If the case doesn’t apply to HR, agents can transfer the case to the appropriate department.

```javascript
var copiedFields = {
  'opened_for': '<user_sys_id>',
  'short_description': email.subject,
  'restricted': false,
  'priority': '1'
};

// createUniversalRequest() must be called to create the universal request prior to creating the child case
var ur = new sn_uni_req.UniversalRequestUtils().createUniversalRequest(copiedFields);
current.universal_request = ur;
```

**UniversalTaskUtils API - Scoped, Global**

The `UniversalTaskUtils` class provides methods for managing universal tasks. With the ServiceNow Universal Task application, agents can create tasks for employees. For example, agents can ask for additional information or request an action to resolve a parent ticket or request. Universal tasks are available for any ticket type that extends the Task [task] table.

Using this API you can change the state of active universal tasks to "Complete" or "Cancelled", obtain all active universal tasks for a specified parent task, check whether a parent universal task has any children, and apply templates to a universal task record. You can use these methods in scripts and in the Visable condition builders on the Tab configuration form to manage the data that appears on the Universal Task tab. For additional information, see Add a Task tab on the Standard Ticket page.
You can use this API in both scoped and global applications. The Universal Task application (sn_uni_task) must be installed on the associated instance to have access to this API. You must always specify the sn_uni_task namespace when calling methods in this API.

For additional information on the Universal Task application, see Universal Task.

*UniversalTaskUtils - applyTemplate(String templateSysId, GlideRecord uniTaskGr)*

Applies the specified universal task template to the specified universal task record.

Before you can use this method, there must be universal task templates configured in your instance. For details, see Universal Task templates.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>templateSysId</td>
</tr>
<tr>
<td>uniTaskGr</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

This code example shows a function that acquires, and then applies, the Feedback Template.
UniversalTaskUtils - getActiveChildTasks(String parentSysId)

Returns the active child task records, in the form of a GlideRecord, for the specified parent ticket.

You can then use the GlideRecord API, scoped or global depending on the calling application's scope, to access the returned data, such as using the getRowCount() method to count active tasks.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parentSysId</td>
<td>String</td>
<td>Sys_id of the parent ticket whose active child tickets to return.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>GlideRecord that contains all active child task records for the specified parent task.</td>
</tr>
</tbody>
</table>

The following example shows a function that obtains all active child tasks for the current sys_id and then uses getRowCount() to obtain the number of active child tasks.

```
(function executeRule(current, previous /*null when async*/) {
  var gr_ActiveTasks = new sn_uni_task.UniversalTaskUtils().getActiveChildTasks(current.sys_id);
  var count = gr_ActiveTasks.getRowCount();
})(current, previous);
```

UniversalTaskUtils - hasTasksToShow(GlideRecord current)

Checks whether the specified parent ticket has any universal tasks that are in the work-in-progress or complete state.
You can use this method to determine whether to show the Universal Task tab to a requester in a standard ticket configuration only if there are tasks that are work-in progress or complete.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>current</td>
<td>GlideRecord</td>
<td>GlideRecord of the parent ticket to check.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| sys_id | If universal tasks are associated with the parent ticket, the sys_id of the parent ticket; otherwise null.  
Data type: String |

The following example shows a function that calls this method to check for universal tasks.

```javascript
(function executeRule(current, previous /*null when async*/) {
  var sysId = new sn_uni_task.UniversalTaskUtils().hasTasksToShow(current);
})(current, previous);
```

**UniversalTaskUtils - markActiveChildTasksCancelled(String parentSysId)**

Changes the state of all active universal tasks under the specified parent ticket to "Cancelled".

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parentSysId</td>
<td>String</td>
<td>Sys_id of the parent ticket whose active child universal tasks' state should be changed to &quot;Cancelled&quot;.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
This code example shows how to use this method to update the states of all universal tasks associated with the specified parent task to cancelled.

```
(function executeRule(current, previous /*null when async*/) {
   new sn_uni_task.UniversalTaskUtils().markActiveChildTasksCancelled(current.sys_id);
})(current, previous);
```

**UniversalTaskUtils - markActiveChildTasksCompleted(String parentSysId)**

Changes the state of all active child universal tasks associated with the specified parent ticket to "Complete".

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parentSysId</td>
<td>String</td>
<td>Sys_id of the parent ticket whose active child universal tasks' state should be changed to &quot;Complete&quot;.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

This code example shows how to use this method to update the states of all universal tasks associated with the specified parent task to completed.

```
(function executeRule(current, previous /*null when async*/) {
   new sn_uni_task.UniversalTaskUtils().markActiveChildTasksCompleted(current.sys_id);
})(current, previous);
```

**UPSHandler - Global**

Handles SNMP classification and identification for UPSs.

Use this API for SNMP-related discovery.

**UPSHandler - classifyAndIdentify()**

Classifies and identifies the UPSs.
### UserCriteria - Scoped

The UserCriteria API enables you to create, modify, or delete user criteria records using scripts.

To use this class in a scoped application, use the `sn_uc` namespace identifier. The User Criteria Scoped API plugin (ID: com.glideapp.user_criteria.scoped.api) should be enabled to access the UserCriteria API.

**Scoped UserCriteria - create(Object columnValues, Boolean standardUpdate)**

Creates a user criteria with specified values in the user_criteria table. Values specified in columnValues override the values provided via setters.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columnValues</td>
<td>Object</td>
<td>Key and value pairs for a column and its value.</td>
</tr>
<tr>
<td>standardUpdate</td>
<td>Boolean</td>
<td>Set to true to enable the running of engines and workflow.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>sys_id of the created user criteria.</td>
</tr>
</tbody>
</table>

```javascript
var uc = new sn_uc.UserCriteria();
uc.setCompanies("31be3d53792000044e0bfc8bcbe5dec,0c441abb6112275000025157c651c89");
uc.setActive(true);
uc.setUsers("31be3d53792000044e0bfc8bcbe5dec,0c441abb6112275000025157c651c89");
```
var UserCriteriaId = uc.create();
gs.info(UserCriteriaId);

Output:
41bea3d53790200044e0bfc8bcbe5dec

Scoped UserCriteria - deleteRecord()
Deletes the current user criteria.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>If true, the user criteria is deleted.</td>
</tr>
<tr>
<td></td>
<td>If false, no user criteria is found to delete.</td>
</tr>
</tbody>
</table>

var uc = new sn_uc.UserCriteria("31bea3d53790200044e0bfc8bcbe5dec");
uc.deleteRecord();

Output:
true

Scoped UserCriteria - read(String columns)
Displays the mapping for the attribute and value pairs of the catalog item.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columns</td>
<td>String</td>
<td>Array of catalog item attributes.</td>
</tr>
</tbody>
</table>
**Scope UserCriteria - setActive(Boolean active)**

Specifies if the user criteria is active.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| active| Boolean | If true, the user criteria is active.  
If false, the user criteria is inactive. |

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var uc = new sn_uc.UserCriteria();
uc.setActive(true);
```

**Scope UserCriteria - setAdvanced(boolean advanced)**

Specifies if the user criteria has an advanced script.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>advanced</td>
<td>Boolean</td>
<td>If true, the user criteria has an advanced script. If false, the user criteria does not have an advanced script.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var uc = new sn_uc.UserCriteria();
uc.setAdvanced(true);
```

### Scoped UserCriteria - setCompanies(String companies)

Sets the company property for the user criteria.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>companies</td>
<td>String</td>
<td>Comma-separated list of the company sys_ids to be set for the user criteria.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var uc = new sn_uc.UserCriteria();
uc.setCompanies("31bea3d53790200044e0bfc8bcbe5dec,0c441abbc611227500025157c651c89");
```

### Scoped UserCriteria - setDepartments(String departments)

Sets the department property for the user criteria.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>departments</td>
<td>String</td>
<td>Comma-separated list of the department sys_ids to be set for the user criteria.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var uc = new sn_uc.UserCriteria();
uc.setDepartments("31bea3d53790200044e0bfc8bcbe5dec,0c441abbc6112275000025157c651c89");
```

Scoped UserCriteria - setGroups(String groups)

Sets the group property for the user criteria.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>groups</td>
<td>String</td>
<td>Comma-separated list of the group sys_ids to be set for the user criteria.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var uc = new sn_uc.UserCriteria();
uc.setGroups("31bea3d53790200044e0bfc8bcbe5dec,0c441abbc6112275000025157c651c89");
```

Scoped UserCriteria - setLocations(String locations)

Sets the location property for the user criteria.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>locations</td>
<td>String</td>
<td>Comma-separated list of the location sys_ids to be set for the user criteria.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var uc = new sn_uc.UserCriteria();
uc.setLocations("31bea3d53790200044e0bfc8bcbe5dec,0c441abbc6112275000025157c651c89");
```

### Scoped UserCriteria - setMatchAll(Boolean match_all)

Sets the match_all property for the user criteria.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>match_all</td>
<td>If true, all conditions set as properties for the user criteria should be fulfilled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If false, at least one condition set as a property for the user criteria should be fulfilled.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var uc = new sn_uc.UserCriteria();
uc.setMatchAll(true);
```

### Scoped UserCriteria - setName(String name)

Sets the name property for the user criteria.

```javascript
var uc = new sn_uc.UserCriteria();
uc.setName("My UserCriteria");
```
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the user criteria.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var uc = new sn_uc.UserCriteria();
uc.setName("Property1");
```

#### Scoped UserCriteria - setRoles(String roles)

Sets the role property for the user criteria.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>roles</td>
<td>String</td>
<td>Comma-separated list of the role sys_ids to be set for the user criteria.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var uc = new sn_uc.UserCriteria();
uc.setRoles("31bea3d53790200044e0bfc8bcbe5dec,0c441abbc6112275000025157c651c89");
```

#### Scoped UserCriteria - setScript(String script)

Sets the script for the user criteria.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>script</td>
<td>String</td>
<td>Script to be set for the advanced user criteria.</td>
</tr>
</tbody>
</table>
**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var uc = new sn_uc.UserCriteria();
uc.setScript("function scriptTest() {
    var rval;
    if (gs.getUser().getRecord().getDisplayValue('department') == 'Product Management') {
        rval = true;
    } else {
        rval = false;
    }
    return rval;
}"");
```

**Scoped UserCriteria - setUsers(String users)**

Sets the user property for the user criteria.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>users</td>
<td>String</td>
<td>Comma-separated list of the user sys_ids to be set for the user criteria.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var uc = new sn_uc.UserCriteria();
uc.setUsers("31bea3d53790200044e0bfc8bcbe5dec,0c441abbc6112275000025157c651c89");
```

**Scoped UserCriteria - update(Object columnValues, String reason)**

Updates the current catalog item with the specified values.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columnValues</td>
<td>Object</td>
<td>Mapping for the column name and the value pairs.</td>
</tr>
<tr>
<td>reason</td>
<td>String</td>
<td>Reason for updating the catalog item.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Returns the sys_id of the created user criteria.</td>
</tr>
</tbody>
</table>

```javascript
var uc = new sn_uc.UserCriteria();
uc.update("name": "Updated name", "The existing name is not relevant. Setting a relevant name");
```

Output:

31bea3d53790200044e0bfc8bcbe5dec

Scoped UserCriteria - UserCriteria()

Creates an instance of the UserCriteria class.

```javascript
var uc = new sn_uc.UserCriteria();
```

Scoped UserCriteria - UserCriteria(String sys_id)

Creates an instance of the UserCriteria class with the specified sys_id.

```javascript
var uc = new sn_uc.UserCriteria("31bea3d53790200044e0bfc8bcbe5dec");
```

UserSkillAnalyzer - Global

Analyzes user skills against skills required to complete work items.

Requires the Skills Management plugin (com.snc.skills_management), which you can activate if you have the admin role.
See Qualify and rank users based on skills and skill levels.

**UserSkillAnalyzer - analyzeUserByID(String requiredSkillsJson, String userID, Boolean isSkillLevelEnforced)**

Analyzes user skills against the required skills and sets the number of matching skills, skill level gap, and qualification that can be collected from an instance.

Analyzes whether an agent is qualified to work on a task by verifying if the agent has the skills and skill levels required to complete the task. If an agent is qualified to work on a task, the API analyzes the level gap between required skills and user’s skill level which can be further used by the application to choose the best agent based on level gap. It also analyzes the total number of skills with and without skill levels the agent has to execute that task.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requiredSkillsJson</td>
<td>String</td>
<td>List of required skills to compare with user. Each skill is listed as a JSON entry in the following format:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• is_mandatory: Boolean. Flag that indicates whether the skill is mandatory.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• level: String. Sys ID from the Skill Levels [cmn_skill_level] table.</td>
</tr>
<tr>
<td>userID</td>
<td>String</td>
<td>Sys ID from the User [sys_user] table.</td>
</tr>
<tr>
<td>isSkillLevelEnforced</td>
<td>Boolean</td>
<td>True if user must have a minimum skill level for all required mandatory skills, false otherwise. Default: false.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>JSON that contains all the required information about the user based on the analyzer skills map as follows:</td>
</tr>
<tr>
<td></td>
<td>• sys_id: String. Sys ID from the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>• is_qualified: Boolean. Flag that indicates whether the user is qualified for the required skills.</td>
</tr>
</tbody>
</table>
Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>◦ True if user has all the mandatory skills and skill level is enforced.</td>
<td></td>
</tr>
<tr>
<td>◦ False if user does not have all mandatory skills or does not meet level</td>
<td></td>
</tr>
<tr>
<td>requirements.</td>
<td></td>
</tr>
<tr>
<td>• num_skills: Number. Skills matched against the required skills.</td>
<td></td>
</tr>
<tr>
<td>• num_skills_matching_level: Number. User skills matching the required level.</td>
<td></td>
</tr>
<tr>
<td>• total_skill_level_gap: Number. Skill level gap helps ranking algorithm</td>
<td></td>
</tr>
<tr>
<td>find optimal user meeting minimum skill level requirements. Calculated</td>
<td></td>
</tr>
<tr>
<td>based on overall gap between task skill level and user skill level. User</td>
<td></td>
</tr>
<tr>
<td>must have required skill level for mandatory skills.</td>
<td></td>
</tr>
<tr>
<td>• optional_skill_level_gap: Number. Provides skill level gap for optional</td>
<td></td>
</tr>
<tr>
<td>only. Assists the end points in distinguishing between total skill level</td>
<td></td>
</tr>
<tr>
<td>and optional skill level gap.</td>
<td></td>
</tr>
</tbody>
</table>

Error if inaccurate parameters or malformed JSON provided.

```javascript
var skills = [{"sys_id":"48c9f873c0a8018b65c3814608b201e6", "is_mandatory": true,
    "level":"4e0ac4d6b3332300290ea943c6a8dc4e"},
    {"sys_id":"48c9fdddc0a8018b04bd8d7914c82c9d", "is_mandatory": false,
    "level":"4e0ac4d6b3332300290ea943c6a8dc4e"}];
var userSysID = "a8f98bb0eb32010045e1a5115206fe3a";
var userSkillAnalyzer = new SNC.UserSkillAnalyzer();
var result = userSkillAnalyzer.analyzeUserById(JSON.stringify(skills), userSysID, true);
gs.info('Results: ' + result);
```

Output:

Calling user analyzer
*** Script: Results:
{"sys_id":"a8f98bb0eb32010045e1a5115206fe3a","is_qualified":false,"num_skills":0,"num_skills_matching_level":0,"total_skill_level_gap":0,"optional_skill_level_gap":0}

**UserSkillAnalyzer - analyzeUserBySkills(String requiredSkillsJson, String userSkillsJson, Boolean isSkillLevelEnforced)**

Analyzes user skills against the required skills and sets the number of matching skills, skill-level gap, and qualification that can be collected from an instance.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requiredSkillsJson</td>
<td>String</td>
<td>List of required skills to qualify the users against. Each skill is listed as a JSON entry in the following format:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>is_mandatory</code>: Boolean. Flag that indicates whether the skill is mandatory.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>level</code>: String. Sys ID from the Skill Levels [cmn_skill_level] table.</td>
</tr>
<tr>
<td>userSkillsJson</td>
<td>String</td>
<td>List of user skills to determine qualification for required skills. Each skill is listed as a JSON entry in the following format:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>is_mandatory</code>: Boolean. Flag that indicates whether the skill is mandatory.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>level</code>: String. Sys ID from the Skill Levels [cmn_skill_level] table.</td>
</tr>
<tr>
<td>isSkillLevelEnforced</td>
<td>Boolean</td>
<td>True if user must have minimum skill level for all required mandatory skills, false otherwise. Default: false.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>JSON that contains all the required information about the user based on the analyzer skills map as follows:</td>
</tr>
<tr>
<td></td>
<td>• <code>sys_id</code>: String. Sys ID from the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>• <code>is_qualified</code>: Boolean. Flag that indicates whether the user is qualified for the required skills.</td>
</tr>
<tr>
<td></td>
<td>◦ True if user has all the mandatory skills and skill level is enforced.</td>
</tr>
<tr>
<td></td>
<td>◦ False if user does not have all mandatory skills or does not meet level requirements.</td>
</tr>
<tr>
<td></td>
<td>• <code>num_skills</code>: Number. Skills matched against the required skills.</td>
</tr>
</tbody>
</table>
### Returns (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• num_skills_matching_level: Number.</td>
<td>User skills matching the required level.</td>
</tr>
<tr>
<td>• total_skill_level_gap: Number.</td>
<td>Skill level gap helps ranking algorithm find optimal user meeting minimum skill level requirements. Calculated based on overall gap between task skill level and user skill level. User must have required skill level for mandatory skills.</td>
</tr>
<tr>
<td>• optional_skill_level_gap: Number.</td>
<td>Provides skill level gap for optional skills only. Assists the end points in distinguishing between total skill level and optional skill level gap.</td>
</tr>
</tbody>
</table>

Error if inaccurate parameters or malformed JSON provided.

```javascript
var skills = ["sys_id":"48c9f873c0a8018b65c3814608b201e6","is_mandatory":true,"level":"4e0ac4d6b3332300290ea943c6a8dc4e"],{"sys_id":"48c9fddcc0a8018b04bd8d7914c82c9d","is_mandatory":false,"level":"4e0ac4d6b3332300290ea943c6a8dc4e"};
var userSkills = ["sys_id":"48c9f873c0a8018b65c3814608b201e6","is_mandatory":true,"level":"4e0ac4d6b3332300290ea943c6a8dc4e"],{"sys_id":"k, th","is_mandatory":false,"level":"4e0ac4d6b3332300290ea943c6a8dc4e"};
var userSkillAnalyzer = new SNC.UserSkillAnalyzer();
var result = userSkillAnalyzer.analyzeUserBySkills(JSON.stringify(skills),JSON.stringify(userSkills));
gs.info('Results: ' + result);
```

### UserSkillAnalyzer - UserSkillAnalyzer()

Instantiates a new UserSkillAnalyzer object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var userSkillAnalyzer = new SNC.UserSkillAnalyzer();
```
**UserSkillRanking - Global**

Used to configure options for choosing the right user and getting a sorted list of qualified users based on number of matching skills and skill-level gap.

Requires the Skills Management plugin (com.snc.skills_management), which you can activate if you have the admin role.

See Qualify and rank users based on skills and skill levels.

**UserSkillRanking - UserSkillRanking()**

Instantiates a UserSkillRanking object.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**UserSkillRanking - getQualifiedRankedUsers(String requiredSkillsJson, String usersJson, String rankType, Boolean isSkillLevelEnforced)**

Provides a list of qualified users based on matching mandatory skills and levels if the isSkillLevelEnforced parameter is true).

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requiredSkillsJson</td>
<td>String</td>
<td>List of required skills to qualify users. Each skill is listed as a JSON entry in the following format: • sys_id: String. Sys ID of skill from the Skills [cmn_skill] table. • is_mandatory: Boolean. Flag that indicates whether the skill is mandatory. • level: String. Sys ID from the Skill Levels [cmn_skill_level] table.</td>
</tr>
<tr>
<td>usersJson</td>
<td>String</td>
<td>List of user Sys IDs from sys_user table (in JSON format representing users to be qualified against required skills).</td>
</tr>
<tr>
<td>rankType</td>
<td>String</td>
<td>Enables sorting qualified users based on ranking. Possible values are comparators to provide sorted list of qualified users based on number of matching skills and skill level gap:</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO_RANKING</td>
<td></td>
<td>(default) – Used if empty value provided.</td>
</tr>
<tr>
<td>NUM_SKILLS</td>
<td></td>
<td>Number of skills matched against required skills.</td>
</tr>
<tr>
<td>NUM_SKILL_MATCHING_LEVEL</td>
<td></td>
<td>Number of user skills matching the required skill level.</td>
</tr>
<tr>
<td>LEAST_SKILL_LEVEL_GAP</td>
<td></td>
<td>Finds user skill that most closely meets minimum skill level requirements.</td>
</tr>
<tr>
<td>MOST_SKILL_LEVEL_GAP</td>
<td></td>
<td>Finds user that least meets minimum skill level requirements.</td>
</tr>
<tr>
<td>BEST_OPTIMUM_USER</td>
<td></td>
<td>Finds optimum user for all skill levels.</td>
</tr>
</tbody>
</table>

| isSkillLevelEnforced | Boolean   | True if user must have minimum skill level for all required mandatory skills, false otherwise. Default: false. |

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>List of qualified users as JSON. Error if inaccurate parameters or malformed JSON provided.</td>
</tr>
</tbody>
</table>

```javascript
var skills = [
  {"sys_id": "48c9f873c0a8018b65c3814608b201e6", "is_mandatory": true,
   "level": "4e0ac4d6b3332300290ea943c6a8dc4e"},
  {"sys_id": "48c9fdddc0a8018b04bd8d7914c82c9d", "is_mandatory": false,
   "level": "4e0ac4d6b3332300290ea943c6a8dc4e"}];
var users = ["a8f98bb0eb32010045e1a515206fe3a", "62826bf03710200044e0bfc8bcbe5df1"];
var userSkillRanking = new SNC.UserSkillRanking();
var qualifiedUsers = userSkillRanking.getQualifiedRankedUsers(JSON.stringify(skills),
  JSON.stringify(users), "BEST_OPTIMUM_USER", true);
```

### v_query – Scoped, Global

Scriptable object that represents a query running against a remote table.

This API requires the Remote Tables plugin (com.glide.script.vtable) to be activated. Retrieving external data using remote tables and scripts.
Remote table rows are created using the `v_table` API.

**v_query - getCondition(String field)**

Gets an encoded query string for the specified field.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>field</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

The following example shows results for a field with an encoded query of `number=INC0001^active=true`.

```javascript
var result = v_query.getCondition("number");
gs.info(result);
```

Output:

```
number=INC0001
```

**v_query - getEncodedQuery()**

Returns the query against a remote table as an encoded query string

For details, see [Encoded query strings](#).

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

The following example is a snippet from [Retrieving specific records from a third-party source](#).
gs.info(v_query.getEncodedQuery());

Output:
active=true^priority=1

**v_query - getParameter(String field)**

Gets the value of a field in an equality query condition.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>String</td>
<td>Name of the field to be queried.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Value of the field in the query condition. For example, if name=John is the encoded query, then getParameter(&quot;name&quot;) returns &quot;John&quot;.</td>
</tr>
</tbody>
</table>

The following example is a snippet from *Retrieving specific records from a third-party source*.

v_query.getParameter("caller_id")

Output:
1234512345123451234512345123451234501

**v_query - getSysId()**

Returns the sys_id value in a get query.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Sys_id value in the get query, for example, if sys_id=123, this method returns 123.</td>
</tr>
</tbody>
</table>

The following example is a snippet from Retrieving specific records from a third-party source.

v_query.getSysId();

Output:

a3a7ffb6dba41010db2051735e4619b7

**v_query - getTextSearch()**

Gets a text search query parameter.

See also:
- Encoded query strings
- Querying tables in script

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Text search query parameter, for example email.</td>
</tr>
</tbody>
</table>

In the following example, the method returns true if the query contains a text query parameter, such as GOTO123TEXTQUERY321=email.

// for query value GOTO123TEXTQUERY321=email
v_query.getTextSearch();

Output:

email
**v_query - isGet()**

Determines if the query is a get query, that is, a query that retrieves a record by sys_id.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Boolean  | Flag indicates that query for a specific record has sys_id= as the query format. Valid values:  
|          | • true: Query contains equality query condition on the sys_id field.        |
|          | • false: Query does not include equality query condition on the sys_id field. |

The following example is a snippet from *Retrieving specific records from a third-party source*. If the encoded query is `sys_id=12345123`, the `v_query.isGet()` method returns true. If the encoded query is anything else, such as `Number=INC1234`, the `v_query.isGet()` method returns false.

```java
if (v_query.isGet()) {
    bySysId(v_table, v_query, v_query.getSysId());
} else if (v_query.getParameter("caller_id")) {
    byCallerId(v_table, v_query, v_query.getParameter("caller_id"));
} else {
    fetchAllIncidents(v_table, v_query);
}
```

**v_query - isTextSearch()**

Indicates if the query contains a text query parameter.

See also:
- Encoded query strings
- Querying tables in script

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the query contains a text query parameter.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Query contains a text query parameter.</td>
</tr>
<tr>
<td></td>
<td>• false: Query does not contain a text query parameter.</td>
</tr>
</tbody>
</table>

In the following example, the method returns true if the query contains a text query parameter, such as GOTO123TEXTQUERY321=email.

```java
v_query.isTextSearch();
```

**v_query** - **setLastErrorMessage(String message)**

Sets the last error message in the GlideRecord.

See also **Scoped GlideRecord - getLastErrorMessage()**.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Error message.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

The following example is a snippet from **Retrieving specific records from a third-party source**.
function fetchAllIncidents(v_table, v_query) {
  // Uses RestMessage with name 'Remote Instance Incidents' and function 'All Incidents'
  // Create a RestMessage first which calls an external REST service
  try {
    var restMessage = new sn_ws.RESTMessageV2('Remote Instance Incidents', 'All Incidents');
    var response = restMessage.execute();
    var responseBody = response.getBody();

    // if REST call ends up in an error, set the last error message which shows up
    // at the bottom of the list view
    if (response.haveError()) {
      v_query.setLastErrorMessage(response.getErrorMessage());
      // can use gs.error() or gs.addErrorMessage() while debugging
      // gs.debug() messages visible in session debugger
      // gs.debug(response.getErrorMessage());
      return;
    }
  } catch (ex) {
    v_query.setLastErrorMessage(ex.message);
    // gs.debug(ex.message);
    return;
  }
}

v_table – Scoped, Global
Scriptable object that enables you to add rows to a remote table.
This API requires the Remote Tables plugin (com.glide.script.vtable) to be activated. Retrieving external data using remote tables and scripts.
Use the v_query scriptable object to query remote tables.

v_table - addRow(Object row)
Adds rows to the remote table.

See also Create a script definition for a remote table.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| row    | Object   | JavaScript object containing field name and value map in which the key is the field name, for example, `{number: "INC0001", sys_id: "a34"}.

Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>row.&lt;field value&gt;</td>
<td>String</td>
<td>Represents the value of the selected field. Although no fields are mandatory, ServiceNow recommends sys_id. Example listing only sys_id field and value:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{ sys_id: &quot;&lt;uniqueID&gt;&quot; }</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the row was added to the remote table. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Success.</td>
</tr>
<tr>
<td></td>
<td>• false: Row was not added.</td>
</tr>
</tbody>
</table>

The following example is a snippet from Retrieving specific records from a third-party source.

```javascript
/**
 * Adds rows to `v_table` using `restMessage`
 */
function loadData(v_table, v_query, restMessage) {
    try {
        var response = restMessage.execute();
        var responseBody = response.getBody();

        // if REST call ends up in an error, set the last error message which shows up
        // at the bottom of the list view
        if (response.haveError()) {
            v_query.setLastErrorMessage(response.getErrorMessage());
            // can use gs.error() or gs.addErrorMessage() while debugging
            // gs.debug() messages visible in session debugger
            // gs.debug(response.getErrorMessage());
            return;
        }
    } catch (ex) {
    }
}
v_query.setLastErrorMessage(ex.message);
// gs.debug(ex.message);
return;
}

var transformerDefinition = getTransformerDefinition();
var transformer = new sn_tfrm.Transformer(transformerDefinition, responseBody);
// transformer parses the responseBody and extracts rows
while (transformer.transform()) {
  // row is field-value map e.g. { active:"true", number: "INC00000001"}
  var row = transformer.getRow();
  // you may do any additional transformations to the row like GlideDuration, GlideDataTime etc. For example,
  // row.duration = new GlideDuration(row.duration);

  // finally add the row to the remote table
  v_table.addRow(row);
}

---

**VMUtils - Global**

Provides utility functions for discovery and provisioning of virtual machines.

The `VMUtils` class is available in server-side scripts to convert between UUID and Correlation ID formats.

**VMUtils - turnCorrelationIdToUuid(String correlationId)**

Converts a correlation ID into a UUID for a virtual machine in the ServiceNow system.

---

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>correlationId</td>
<td>String</td>
<td>The correlation ID to convert</td>
</tr>
</tbody>
</table>

---

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The UUID, or an empty string if invalid.</td>
</tr>
</tbody>
</table>
var vmu = new VMUtils();
gs.print(vmu.turnCorrelationIdToUuid('42 10 82 62 35 ca 68-b5 1c 1e f8 5c 0a 0d 5b'));

VMUtils - turnUuidToCorrelationId(String uuid)
Converts a VMware universal unique identifier (UUID) to a format that matches the original format in the ServiceNow system.

vCenter works with UUIDs where the ESXi Linux console worked with this format.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uuid</td>
<td>String</td>
<td>The UUID to convert.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The identifier converted to the ServiceNow format.</td>
</tr>
</tbody>
</table>

var vmu = new VMUtils();
vmu.turnUuidToCorrelationId('13eb78d0-d504-11e1-9b23-0800200c9a66');

WalkWorkflow - Global
Walks the workflow and gets the list of successors for each activity along with a shortest path order that indicates the activity depth in the workflow.

Walkworkflow - computeFullSequences()
Computes the full sequence for every path.

Does not combine paths at Join activities. After calling this method, call getSequences() to yield the computed sequences.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Walkworkflow - computeSequences()**

Computes the sequences of activities.

All sequences leading up to a join end at the join and then a single sequence from the join is computed. This ensures that all sequences can be walked up to a join, and then the sequence can be walked from the join. After calling this method a call to `getSequences()` will yield the computed sequences.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Walkworkflow - dump()**

Outputs the internal state of this class to `gs.print()`.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
**Walkworkflow - dumpSequences()**

Outputs the sequences internal to this class to `gs.print()`.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

**Walkworkflow - dumpShortestPathOrders(description)**

Outputs the shortest path orders internal to this object to `gs.print()`.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

**Walkworkflow - getActivitiesAtOrder(Number order)**

Returns all the activities at the specified shortest path order.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>order</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>An array of activity sys_ids at the specified order. If an empty array is</td>
</tr>
<tr>
<td>Array</td>
<td>returned, there are no more activities.</td>
</tr>
</tbody>
</table>

**Walkworkflow - getSequences()**

Returns the activity sequences.

The `walk()` method must have been called prior to calling this method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Walkworkflow - walk(Boolean fullSequences)**

Walks the workflow and sets up the activity shortest path ordering.

After calling this method the client would call the various accessor methods to obtain relevant data about the workflow.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fullSequences</td>
<td>Boolean</td>
<td>(Optional) If <code>true</code>, computes the full sequences. Default is false.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### Walkworkflow - Walkworkflow(GlideRecord workflowVersion)

Constructor for **Walkworkflow** class.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>workflowVersion</td>
<td>GlideRecord</td>
<td>GlideRecord on table wf_workflow_version of the workflow version to analyze.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workflow</td>
<td>Workflow object just created.</td>
</tr>
</tbody>
</table>

### WFActivityHandler - Global

The base class for all workflow activities.

As the base class for all workflow activities, this code always executes as part of an activity.

When developing your own workflow activities, create your script object by deriving from this class and overriding methods as necessary to get the functionality you want. Client code must override method `onExecute()` to perform meaningful activity processing.

### WFActivityHandler - debug(String msg, String args)

Logs a debug message.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to send to the log.</td>
</tr>
</tbody>
</table>
Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>args</td>
<td>String</td>
<td>String to send to the log</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

`WFActivityHandler - debug(String msg, Array args)`

Logs a debug message.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to send to the log.</td>
</tr>
<tr>
<td>args</td>
<td>Array</td>
<td>Array of values to send to the log</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

`WFActivityHandler - generate(String activityId, String order, GlideDateTime startAtDspValue, Boolean noCreateFlag)`

Generates approvals and tasks before the activity runs for a preview of upcoming work.

Enables generating objects at the start of a workflow in a pending state. For related information, see Generate workflow activity.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>activityId</td>
<td>String</td>
<td>Sys_id of this running activity. Located in the Workflow Activities [wf_activity] table.</td>
</tr>
</tbody>
</table>
### Parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>order</td>
<td>String</td>
<td>The order number associated with the task or approval.</td>
</tr>
<tr>
<td>startAtDspValue</td>
<td>String</td>
<td>The starting time for the task or approval in GlideDateTime format.</td>
</tr>
<tr>
<td>noCreateFlag</td>
<td>Boolean</td>
<td>Flag that indicates whether to skip creating an approval record.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• true: Do not create an approval record.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• false: Create an approval record.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: True</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Pre-generation object saved in the workflow scratchpad. For example, the object might be an array of approval IDs. When the activity runs the onExecute function, it can request the pre-generation object to determine one of the following conditions:</td>
</tr>
<tr>
<td></td>
<td>• Approvals have already been created in a pending state.</td>
</tr>
<tr>
<td></td>
<td>• The activity must create the approvals using the onExecute function.</td>
</tr>
</tbody>
</table>

**WFActivityHandler - info(String msg, String args)**

Logs an information message.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to send to the log.</td>
</tr>
<tr>
<td>args</td>
<td>String</td>
<td>String to send to the log.</td>
</tr>
</tbody>
</table>
WFActivityHandler - info(String msg, Array args)

Logs an information message.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to send to the log.</td>
</tr>
<tr>
<td>args</td>
<td>Array</td>
<td>Array of values to send to the log</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

WFActivityHandler - js(String str)

Evaluates activity variables in the script.

Used to evaluate strings that contain el such as `${some script}` or strings starting with `javascript:`. See almost any existing activity for example usage.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>str</td>
<td>String</td>
<td>String that possibly contains embeded el.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The evaluated string.</td>
</tr>
</tbody>
</table>
WFActivityHandler - onCancel()

Event handler for cancel event.

The base class for the activity script sets this activity to be cancelled. Derived classes (activities) can override this method if additional processing is required to cancel this activity. Activity Manual Approvals is an example of overriding this method to perform additional processing to cancel this activity.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

WFActivityHandler - onExecute()

Virtual method. Activity subclasses must override this method to perform work appropriate to the activity.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

WFActivityHandler - runScript(String script)

Enables activities to run a script contained in an activity variable of type script.

Example activities that use this method include If, Wait for condition, and Approval - User.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>script</td>
<td>String</td>
<td>String containing valid Javascript.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>variable</td>
<td>If the script set the workflow variable answer then this value is returned. If not, then the result of the script execution is returned.</td>
</tr>
</tbody>
</table>

#### WFActivityHandler - setActivityOutput()

Sets the activity output property.

See activity [SOAP Message](#).

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### WFActivityHandler - setResultFailed(String reason)

Sets the activity result as failed with an optional reason string.

See Activity [SOAP Message](#).

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reason</td>
<td>String</td>
<td>(Optional) Description of the reason this activity failed.</td>
</tr>
</tbody>
</table>
**WFActivityHandler - setResultSucceeded()**
Sets the result of this activity as successful.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**WFActivityHandler - warn(String msg, String args)**
Logs a warning message.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to send to the log.</td>
</tr>
<tr>
<td>args</td>
<td>String</td>
<td>String to send to the log</td>
</tr>
</tbody>
</table>

**WFActivityHandler - warn(String msg, Array args)**
Logs a warning message.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to send to the log.</td>
</tr>
<tr>
<td>args</td>
<td>Array</td>
<td>Array of values to send to the log</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### WindowsOSNameHelper - Global

Handles the formatting of Windows OS names.

Use in any server-side discovery scripts where you need to format OS names.

#### WindowsOSNameHelper - formatWindowsOSName(String name)

Formats the given Windows OS name.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name to format</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The formatted name</td>
</tr>
</tbody>
</table>

### WindowsOSNameHelper - makeOSChoiceValid(String value)

Ensures that the OS choice is valid.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>String</td>
<td>The value to validate</td>
</tr>
</tbody>
</table>
WindowsOSNameHelper - osCleanupName(String name)

Cleans up the specified Windows OS name by stripping out incorrect characters, incorrect capitalization, extra spaces, or trailing spaces.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The name to clean up.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The cleaned up name</td>
</tr>
</tbody>
</table>

Workflow - Global

Workflow scripts provide an interface with the workflow engine. Use these methods to manipulate workflows.

Workflow - broadcastEvent(String contextId, String eventName)

Sends the specified event (message) into the workflow context to pass along to the executing activities.

Typical use of this method is to enable activities that wait for some action to occur before proceeding. For additional information on using broadcastEvent, refer to Workflow event-specific functions.

For a list of the available OOB events, refer to Workflow events in the base system.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextId</td>
<td>String</td>
<td>The context ID.</td>
</tr>
<tr>
<td>eventName</td>
<td>String</td>
<td>The name of the event.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
//where current is a task record with a workflow context
var wf = new Workflow().getRunningFlows(current);
while(wf.next()) {
    new Workflow().broadcastEvent(wf.sys_id, 'resume');
}
```

**Workflow - cancel(GlideRecord record)**

Cancels all running workflows on this record by broadcasting the `cancel` event to activities in all running workflows on this record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>GlideRecord</td>
<td>GlideRecord on any table. All workflows running on this record will be cancelled.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
//get workflow helper
var workflow = new Workflow();
//cancel all the workflows, where current is a task record with a workflow context
workflow.cancel(current);
```
Workflow - cancelContext(GlideRecord context)

Cancels this running context by broadcasting a cancel event to activities in this workflow context.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>context</td>
<td>GlideRecord</td>
<td>GlideRecord of the running context to cancel.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

// If a workflow has started for this item, cancel it, where current is a task record with a workflow context

```java
if ((current.stage == 'Request Cancelled') && current.context && !current.context.nil()) {
    var w = new Workflow();
    var now_GR = new GlideRecord('wf_context');

    if (now_GR.get(current.context))
        w.cancelContext(now_GR);
}
```

Workflow - deleteWorkflow(GlideRecord current)

Deletes all the workflows on the record.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>current</td>
<td>GlideRecord</td>
<td>GlideRecord for which the caller wants to delete all workflows. This can be any record on any table.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
//where current is a task record with a workflow context
var wkfw = new Workflow();
wkfw.deleteWorkflow(current);
```

**Workflow - fireEvent(GlideRecord eventRecord, String eventName)**

Fires the named event on the input record.

Used in Activities **Approval Coordinator, Timer, Lock**, and some others.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventRecord</td>
<td>GlideRecord</td>
<td>The event record.</td>
</tr>
<tr>
<td>eventName</td>
<td>String</td>
<td>The name of the event to send to the executing workflow.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
//where current is a task record with a workflow context
var w = new Workflow();
w.fireEvent(current, 'execute');
```

**Workflow - fireEventById(String eventRecordId, String eventName)**

Fires the named event on the record specified by record ID.

Used in Activities **Approval Coordinator, Timer, Lock**, and some others.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventRecordId</td>
<td>String</td>
<td>The sys_id of the glide record.</td>
</tr>
<tr>
<td>eventName</td>
<td>String</td>
<td>The name of the event to send to the executing workflow.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var wkfw = new Workflow();
wkwf.fireEventById('f2400ec10b0a3c1c00ca5bb5c6fae427','Timer');
```

### Workflow - getContexts(GlideRecord record)

Returns all workflow contexts for a specified record.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>GlideRecord</td>
<td>GlideRecord for which the caller wants a list of all workflow contexts. This can be any record on any table for which the caller wants the running workflow contexts.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>GlideRecord in the Workflow context [wf_context] table filtered for all workflow contexts for the specified record (in any state, such as running, cancelled, finished).</td>
</tr>
</tbody>
</table>

```javascript
//where current is a task record with a workflow context
var wkfw = new Workflow();
var context = wkfw.getContexts(current);
while (context.next())
  gs.print(context.started);
```
### Workflow - `getEstimatedDeliveryTime(String workflowId)`

 Gets the estimated time for a workflow to complete.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>workflowId</td>
<td>String</td>
<td>Sys_id of the workflow (table wf_workflow) to get the estimated run time.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Display value from a GlideDuration (e.g., 3 days), or blank if unknown.</td>
</tr>
</tbody>
</table>

```java
var wkfw = new Workflow();
gs.print(wkfw.getEstimatedDeliveryTime('b99a866a4a3623120074c033e005418f'));
```

2 Days

### Workflow - `getEstimatedDeliveryTimeFromWFVersion(GlideRecord wfVersion)`

 Get the estimated elapsed execution time for the workflow version.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wfVersion</td>
<td>GlideRecord</td>
<td>GlideRecord on table wf_workflow_version of a specific workflow version for which the caller wants the estimated duration of executing.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Display value from a GlideDuration (e.g., 3 days), or blank if unknown.</td>
</tr>
</tbody>
</table>

```java
//where current is a task record with a workflow context
var wkfw = new Workflow();
```
Workflow - `getReturnValue(String workflowID, Number amount, Boolean result)`

Gets the appropriate workflow return value for the input workflow ID. This is either the workflow checked out by the current user or the published workflow with the most recent date.

This is either the workflow checked out by the current user or the published workflow with the most recent date. This method is available starting with the Fuji release.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>workflowID</td>
<td>String</td>
<td>The sys_id of the workflow (table wf_workflow)</td>
</tr>
<tr>
<td>amount</td>
<td>Number</td>
<td>amount</td>
</tr>
<tr>
<td>result</td>
<td>Boolean</td>
<td><code>True</code>, if true</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>???</td>
<td>The return value of the workflow as specified by the Return Value activity. Workflows without a Return Value activity return a null value.</td>
</tr>
</tbody>
</table>

```javascript
var wkfw = new Workflow();
wkfw.getReturnValue('context');
```

**Output:**

```javascript
*** Script: b99a866a4a3623120074c033e005418f
```

Workflow - `getRunningFlows(GlideRecord record)`

Gets all the currently running workflow contexts for the input record.

The input record is any record on any table for which the caller wants the running workflow contexts.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>GlideRecord</td>
<td>GlideRecord of the record for which the caller wants a list of all running workflows.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>GlideRecord on table wf_context and filtered for all executing workflow contexts.</td>
</tr>
</tbody>
</table>

```javascript
//where current is a task record with a workflow context
var wf = new Workflow().getRunningFlows(current);
while(wf.next()) {
    new Workflow().broadcastEvent(wf.sys_id, 'pause');
}
```

**Workflow - getVersion(String workflowID)**

Gets the appropriate workflow version for the input workflow ID. This is either the workflow checked out by the current user or the published workflow with the most recent date.

This is either the workflow checked out by the current user or the published workflow with the most recent date.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>workflowID</td>
<td>String</td>
<td>The sys_id of the workflow (table wf_workflow)</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var wkfw = new Workflow();
wkfw.getVersion('b99a866a4a3623120074c033e005418f');
```
Workflow - `getVersionFromName(String workflowName)`

Returns the appropriate workflow version for the input workflow name.

See `getVersion()` for more information.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>workflowName</td>
<td>String</td>
<td>Name of the workflow (table wf_workflow)</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var wkfw = new Workflow();
wkfw.getVersionFromName('Emergency Change');
```

Workflow - `getWorkflowFromName(String workflowName)`

Returns the sys_id of the workflow associated with the specified workflow name.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>workflowName</td>
<td>String</td>
<td>Name of the workflow.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The sys_id of the workflow associated with the passed in name.</td>
</tr>
</tbody>
</table>

```javascript
var wflw = new Workflow();
gs.print(wflw.getWorkflowFromName('Emergency Change'));
```

Workflow - `hasWorkflow(GlideRecord record)`

Determines if a specified record has any workflow contexts associated to it.

This includes running and completed workflow contexts.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>GlideRecord</td>
<td>GlideRecord under scrutiny. This GlideRecord can be from any table.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True, if record has associated workflow; otherwise, returns False.</td>
</tr>
</tbody>
</table>

```javascript
var wkfw = new Workflow();
gs.print(wkfw.hasWorkflow('f2400ec10b0a3c1c00ca5bb5c6fae427'));
```

false

Workflow - `restartWorkflow(GlideRecord current, Boolean maintainStateFlag)`

Recalculates the approvals and tasks for a workflow by adding new approvals and tasks, while not resetting current approvals and tasks.

You can use this method to perform such tasks as adding a company to a change request, without resetting the current approvals for companies already in the workflow.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>current</td>
<td>GlideRecord</td>
<td>GlideRecord of the record this workflow is executing. This can by any record on any table.</td>
</tr>
</tbody>
</table>
| maintainStateFlag | Boolean | Flag that indicates whether to maintain all approvals and tasks in their current state. Valid values:  
  - true: Maintain all approvals and tasks in their current state.  
  - false: Update all approval and task states. |
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

This example shows the workflow being restarted with the approval file changing from Rejected to Requested.

```javascript
(function(){
    var comment = 'Workflow Restarted - the Approval Field changing from Rejected to Requested';
    var gLock = new GlideRecordLock(current);
    gLock.setSpinWait(50);
    if (gLock.get()) {
        new Workflow().restartWorkflow(current, false);
        current.setDisplayValue('approval_history', comment);
    }
})
```

**Workflow - runFlows(GlideRecord record, String operation)**

Runs all workflows for a given record in a given table and its descendant tables.

Sample usage can be seen in the Script Includes "SNC - Run parent workflows", and "SNC - Run parent workflows (Approval)".

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>GlideRecord</td>
<td>GlideRecord to run workflows against.</td>
</tr>
<tr>
<td>operation</td>
<td>String</td>
<td>Database operation. Valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• insert</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• update</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• delete</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
```javascript
var now_GR = new GlideRecord('wf_test');
now_GR.addQuery('parent', current.parent);
now_GR.addQuery('sys_id','!=',current.sys_id);
now_GR.query();
while(now_GR.next()) {
    new Workflow().runFlows(now_GR, 'update');
}
```

**Workflow - startFlow(String workflowId, GlideRecord current, String operation, Array vars)**

Starts a specified workflow.

See script include WorkflowScheduler and Business Rule "Start Workflow" on table sc_req_item for examples of use.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>workflowId</td>
<td>String</td>
<td>The sys_id of the workflow to start. This sys_id refers to table wf_workflow.</td>
</tr>
<tr>
<td>current</td>
<td>GlideRecord</td>
<td>The record to use as current in this workflow. This is normally from the Table field of the workflow properties for this workflow.</td>
</tr>
<tr>
<td>operation</td>
<td>String</td>
<td>The operation to perform on current. Possible values: insert, update, delete.</td>
</tr>
<tr>
<td>vars</td>
<td>Array</td>
<td>Collection of variables to add to the workflow</td>
</tr>
</tbody>
</table>

```javascript
///where current is a task record with a workflow context
var w = new Workflow();
var context = w.startFlow(id, current, current.operation(), getVars());
```

**Workflow - startFlowFromContextInsert(GlideRecord context, String operation)**

Helper method for business rule Auto start on context.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>context</td>
<td>GlideRecord</td>
<td>GlideRecord on table wf_context of a new record (the &quot;current&quot; record in the business rule).</td>
</tr>
<tr>
<td>operation</td>
<td>String</td>
<td>Database operation being performed. One of insert, update, delete.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
//where current is a task record with a workflow context
   current.name = current.workflow_version.name;
   current.started_by.setValue(gs.userID());

   if (gs.nil(current.id)) {
      var now_GR = new GlideRecord('wf_workflow_execution');
      now_GR.name = current.name;
      now_GR.insert();

      current.table = 'wf_workflow_execution';
      current.id = now_GR.sys_id;
   }

   var wf = new Workflow();
   wf.startFlowFromContextInsert(current, current.operation())
```

**Workflow - startFlowRetroactive(String workflowId, Number retroactiveMSecs, GlideRecord current, String operation, Array, ???)**

Used by business rule **Start Workflow** on table task_sla. This starts a workflow and the extra arguments to this method are used by activity "Timer" to pause the execution of the workflow for some duration.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>workflowID</td>
<td>String</td>
<td>The sys_id of the workflow to start. This sys_id refers to table wf_workflow.</td>
</tr>
<tr>
<td>retroactiveMSeconds</td>
<td>Number</td>
<td>Delay in milliseconds used by Activity Timer.</td>
</tr>
<tr>
<td>current</td>
<td>GlideRecord</td>
<td>GlideRecord of the record to use as current in this workflow. This is normally from the Table field of the workflow properties for this workflow</td>
</tr>
<tr>
<td>operation</td>
<td>String</td>
<td>Database operation being performed. One of insert, update, delete.</td>
</tr>
<tr>
<td>vars</td>
<td>Array</td>
<td>Collection of variables to add to the workflow.</td>
</tr>
<tr>
<td>withSchedule</td>
<td>???</td>
<td>Schedule used by Activity Timer.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>A GlideRecord on table wf_context on the inserted record for this newly created workflow context.</td>
</tr>
</tbody>
</table>

// is this a retroactive start?

```javascript
// where current is a task record with a workflow context
var msecs = new GlideDateTime().getNumericValue() -
current.start_time.glideObject().getNumericValue();

// treat this as a retroactive workflow start if the SLA started more than 5 seconds ago
var w = new Workflow();
if (msecs <= 5000)
    w.startFlow(id, current, current.operation());
else
    w.startFlowRetroactive(id, msecs, current, current.operation());

// update the record in case the workflow changed some values
current.update();
```
**Workflow - Workflow()**

Constructor for `Workflow` class.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
var w = new Workflow();
```

**Workflow - Scoped**

The scoped Workflow API provides methods that can be used in an activity definition script.

There are no constructors for creating an instance of a scoped workflow object. Instead, use the global `workflow` object available in activity scripts. This workflow object is available in any script location inside a workflow.

**Scoped Workflow - debug(String message, Object args)**

Adds a debug message to the log.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The message to add to the log.</td>
</tr>
<tr>
<td>args</td>
<td>Object</td>
<td>Arguments to add to the message.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The message added to the log.</td>
</tr>
</tbody>
</table>
var loggedMessage = workflow.debug("All is well");

**Scoped Workflow - error(String message, Object args)**

Adds an error message to the log.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The message to add to the log.</td>
</tr>
<tr>
<td>args</td>
<td>Object</td>
<td>Arguments to add to the message.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The logged message</td>
</tr>
</tbody>
</table>

var loggedMessage = workflow.error("An error has occurred. ");

**Scoped Workflow - getVariable(String name)**

Returns the specified variable's value.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The variable name</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The variable's value</td>
</tr>
</tbody>
</table>

var value = workflow.getVariable("task");

**Scoped Workflow - info(String message, Object args)**

Adds an informational message to the log.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The message to add to the log.</td>
</tr>
<tr>
<td>args</td>
<td>Object</td>
<td>Arguments to add to the message.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The message that is logged.</td>
</tr>
</tbody>
</table>

```javascript
var loggedMessage = workflow.info("All is well");
```

## Scoped Workflow - inputs

Object that contains the workflow variables.

This is not a method, but rather a way to reference the variables in a workflow object, such as the user. Workflow variables are defined in the properties of the workflow or by scripted activities.

<table>
<thead>
<tr>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>inputs</td>
</tr>
</tbody>
</table>

```javascript
var user = workflow.inputs.u_user;
```

## Scoped Workflow - name()

Returns the workflow name.

## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The workflow name</td>
</tr>
</tbody>
</table>

```javascript
var name = workflow.name();
```

**Scoped Workflow - removeVariable(String name)**

Removes the specified variable from the workflow.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The variable name</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var value = workflow.removeVariable("task");
```

**Scoped Workflow - result**

Element of the workflow object that contains the workflow results.

This is not a method, but rather the element in the workflow object that contains the results of a workflow.

**Field**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>String</td>
<td>Workflow results.</td>
</tr>
</tbody>
</table>

```javascript
var wfResult = workflow.result;
```
Scoped Workflow - scratchpad()

Returns the workflow's scratchpad object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The scratchpad object.</td>
</tr>
</tbody>
</table>

```javascript
var scratchpad = workflow.scratchpad();
```

Scoped Workflow - setResult(String result)

Sets the workflow's result.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>String</td>
<td>The workflow's result</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
workflow.setResult("Success");
```

Scoped Workflow - setVariable(String name, Object value)

Sets the specified variable to the specified value.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The variable name</td>
</tr>
<tr>
<td>value</td>
<td>Object</td>
<td>The value to be assigned to the variable.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```java
workflow.setVariable("task", "terrible");
```

### Scoped Workflow - `warn(String message, Object args)`

Adds a warning message to the log.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>The message to add to the log.</td>
</tr>
<tr>
<td>args</td>
<td>Object</td>
<td>Arguments to add to the message.</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The logged message</td>
</tr>
</tbody>
</table>

```java
var loggedMessage = workflow.warn("Check your permissions.");
```

### WorkflowDuration - Global

This class calculates the duration (in seconds) based on the variables of a workflow activity. It is an interface between `Workflow Timer()` and `DurationCalculator()`.
WorkflowDuration - addSeconds(GlideRecord record, Number amount)

Adds the number of seconds to the specified start date/time.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>GlideRecord</td>
<td>description</td>
</tr>
<tr>
<td>amount</td>
<td>Number</td>
<td>amount</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True, if object was successfully created.</td>
</tr>
</tbody>
</table>

WorkflowDuration - calculate(GlideRecord record)

Calculates the number of seconds and the due date.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>GlideRecord</td>
<td>The record that contains the fields with the schedule, timezone, and date/time information. This is almost always activity.vars.<strong>var_record</strong>.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

WorkflowDuration - calculateTimeLeft(GlideRecord record)

Calculates the number of seconds remaining to the specified end date/time.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record</td>
<td>GlideRecord</td>
<td>The record that contains the fields with the schedule, timezone, and date/time information. This is almost always <code>activity.vars.__var_record__</code>.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**WorkflowDuration - getEndDateTime()**

Returns the end date/time set by a call to `calculate(record)`.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideDateTime</td>
<td>The end date/time.</td>
</tr>
</tbody>
</table>

**WorkflowDuration - getSeconds()**

Returns the seconds value that was set by a call to `calculate(record)`.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The number of seconds set by a call to calculate(record).</td>
</tr>
</tbody>
</table>

**WorkflowDuration - getTotalSeconds()**

Returns the totalSeconds value that was set by a call to calculate(record).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>The total number of seconds set by a call to calculate(record).</td>
</tr>
</tbody>
</table>

**WorkflowDuration - setEndDateTime(String dt)**

Sets the end date/time to use when calculating the remaining time.

To convert the value into the GlideDateTime internal format, use GlideDateTime.getValue().

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dt</td>
<td>String</td>
<td>The end date/time to use.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**WorkflowDuration - setEndDateTime(GlideDateTime dt)**

Sets the end date/time to use when calculating the remaining time.
To convert the value into the GlideDateTime internal format, use `GlideDateTime.getValue()`.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dt</td>
<td>GlideDateTime</td>
<td>The end date/time to use.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**WorkflowDuration - setStartDateTime(String dt)**

Sets the start date/time to use in the calculations.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dt</td>
<td>String</td>
<td>The start date/time to use, in GMT.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**WorkflowDuration - setStartDateTime(GlideDateTime dt)**

Sets the start date/time to use in the calculations.

To convert the value into the GlideDateTime internal format, use `GlideDateTime.getValue()`.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dt</td>
<td>GlideDateTime</td>
<td>The start date/time to use.</td>
</tr>
</tbody>
</table>
WorkflowDuration - setUsedSecs(Number secs)

Sets the used seconds compensation, where the number of seconds is the number of seconds inside of any schedule.

Can be a -ve number, which extends the duration.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>secs</td>
<td>Number</td>
<td>The number of used seconds to set.</td>
</tr>
</tbody>
</table>

WorkflowDuration - setWorkflow(Object schedule, Object timezone)

Sets the workflow schedule/timezone. Used for schedule_type workflow_schedule and timezone_type workflow_timezone).

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>schedule</td>
<td>Object</td>
<td>The schedule to use. Usually from the workflow context.schedule.</td>
</tr>
<tr>
<td>timezone</td>
<td>Object</td>
<td>The timezone to use. Usually from the workflow context.timezone.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
WorkflowModelManager - Global

The WorkflowModelManager class provides a way to query the workflow model, to step backwards and forwards between specified *wf_history* items, and to query the history for activity and transition-specific information.

The interaction with the WorkflowModelManager class is through the ActivityHistoryRecord data object. The ActivityHistoryRecord is an inner class and is only used while interacting with WorkflowModelManager.

To acquire the executed history of the workflow activities:

```javascript
var model = new WorkflowModelManager('myContextId');
var activities = model.getExecutedHistory();
```

To output the playback of the workflow:

```javascript
var model = new WorkflowModelManager('myContextId');
model.getExecutedHistory();
model.playBack();
```

At this time, the playback is required to load the executed transitions. This will also play a role in walking backwards on a model to rollback to a specific activity.

To see the details of the cached model:

```javascript
var model = new WorkflowModelManager('myContextId');
model.getExecutedHistory();
model.dump();
```

Related information

**WFActivityHandler - Global**

**WorkflowModelManager - getActivityHistoryRecordById(String haRecordSysId)**

Retrieves the history activity that is cached by the *wf_history.sys_id* provided in the argument.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>haRecordSysId</td>
<td>String</td>
<td>The sys_id of the activity history (table wf_history).</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActivityHistoryRecord</td>
<td>The ActivityHistoryRecord JavaScript object requested.</td>
</tr>
</tbody>
</table>

**WorkflowModelManager - getAllTransitionedIntoActivity(Object description)**

Retrieves the history activities that executed and transitioned into the one represented by the `sys_id` in the argument.

The `next` status is based on the `wf_activity.sys_id` associated with the activity represented in the haRecord existing as a TO in a transition associated with any ActivityHistoryRecords that executed in the workflow's history. (This differs from `getPreviousByTransition`, which only returns TO transitions that come before the haRecord in the execution sequence (by time).) The return value is a collection of ActivityHistoryRecords that identify the argument haRecord.wfaId as their TO activity. The return values are based on all transitions in the executed history collection that transition To get the activity that executed prior to this activity in time use `getPreviousByExecutedOrder`.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>Object</td>
<td>ActivityHistoryRecord JavaScript object.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Array</td>
<td>Array of ActivityHistoryRecord JavaScript objects.</td>
</tr>
</tbody>
</table>

**WorkflowModelManager - getAllTransitionedIntoActivityId(String haRecordSysId)**

Retrieves the history activities that executed and transitioned into the one represented by the `sys_id` in the argument. The `next` status is based on the `wf_activity.sys_id` associated with the activity represented in the haRecord existing as a TO in a transition associated with any ActivityHistoryRecords that executed in the workflow's history. (This differs from `getPreviousByTransition`, which returns only TO transitions that come before the haRecord in the execution sequence (by time).) The return value is a collection of ActivityHistoryRecords that identify the argument haRecord.wfaId as their TO activity.
The next status is based on the `wf_activity.sys_id` associated with the activity represented in the `haRecord` existing as a TO in a transition associated with any `ActivityHistoryRecords` that executed in the workflow's history. (This differs from `getPreviousByTransition`, which returns only TO transitions that come before the `haRecord` in the execution sequence (by time).) The return value is a collection of `ActivityHistoryRecords` that identify the argument `haRecord.wfaId` as their TO activity.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>haRecordSysId</code></td>
<td>String</td>
<td>The sys_id of the activity history (table <code>wf_history</code>).</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Array</td>
<td>Array of <code>ActivityHistoryRecord</code> JavaScript objects requested.</td>
</tr>
</tbody>
</table>

```javascript
var model = new WorkflowModelManager('a143585c3b001000dada82c09ccf3d44');
    model.getExecutedHistory();
    var activity = model.begin;
    gs.print('activity: ' + activity.wfaName + ', transitions: ' +
               activity.transitions.length);
    while( activity != null){
        gs.print('activity: ' + activity.wfaName + ', transitions: ' +
               activity.transitions.length);
        var parents = model.getAllTransitionedIntoActivity(activity);
        for( var i = 0; i < parents.length; i++ ){
            gs.print('-------------- parent activity: ' + parents[i].wfaName);
        }
        activity = model.getNextByExecutedOrder( activity );
    }
```

**WorkflowModelManager - getExecutedHistory()**

This is the worker method that must be called to initialize the `WorkflowModelManager` object. In the process of initializing the `WorkflowModelManager` object, `getExecutedHistory()` creates and populates the `activityHistoryRecord` data object array (which is a member of the `WorkflowModelManager` class). Most `WorkflowModelManager` methods return a subset
of the `activityHistoryRecord` array. Your script can then call one or more of the `activityHistoryRecord` methods to accomplish further work.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String Array</td>
<td>An ordered array of sys_ids from table wf_history. It is unlikely client code would use the return value; it is kept internally to this object for use by subsequent method calls.</td>
</tr>
</tbody>
</table>

```javascript
var model = new WorkflowModelManager('myContextId');
var activities = model.getExecutedHistory();
var current = model.getActivityHistoryRecordById('d6681d573b130000dada82c09ccf3d10');
```

**WorkflowModelManager - getFinalExecutedActivityList()**

Queries the wf_history table by context and retrieves all the activities executed in the workflow given by the context set in the construction of this object.

This function produces a list of executed activities in the exact order each activity passed through the server side ActivityManager.java using the new activity_index to force the order coming out of the database. On its this call will not give the full picture; it needs to load and map the transitions.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Array</td>
<td>Array of ActivityHistoryRecord javascript objects.</td>
</tr>
</tbody>
</table>
WorkflowModelManager - getFinalExecutedActivityIdList()

Gets the list of `wf_history.sys_ids` of all activities that successfully executed and were not rolled back or skipped up to the moment the function was called.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String Array</td>
<td>Array of <code>wf_history.sys_id</code> values for executed activities.</td>
</tr>
</tbody>
</table>

```javascript
var model = new WorkflowModelManager('ee3e0a053b101000dada82c09ccf3d7c');
model.getExecutedHistory();
var finals = model.getFinalExecutedActivityIdList();
gs.print(' EXECUTION PATH IDs --------------- : ' + finals.length);

for ( var x = 0; x < finals.length; x++ ) {
    gs.print(finals[x] );
}
```

WorkflowModelManager - getNextByExecutedOrder(Object haRecord)

Retrieves the history activity that executed just after to the one provided in the argument.

The `next` status is based on the activity index reflecting the nearest prior activity in time and not necessarily the nearest next `wf_activity` the provided activity transitioned to. To get the nearest next activity that transitioned to this `haRecord` passed in, use `getNextByTransitionOrder(haRecord)`.

⚠️ **Note:** In the process of assembling the execution order array, blanks are left in the array where history objects have been deleted. This is done to ensure the index in the array and the activity_index of the object remain in sync. For that reason, all objects coming out of the execution order array should be tested for `nil()` and not assumed to be the previous record based on activity_index or array index value alone.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>haRecord</td>
<td>Object</td>
<td>ActivityHistoryRecord JavaScript object.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The ActivityHistoryRecord JavaScript object requested.</td>
</tr>
</tbody>
</table>

**WorkflowModelManager - getNextByExecutedOrderId(String haRecordSysId)**

Retrieves the history activity that is cached by the `wf_history.sys_id` provided in the argument and then calls into `getNextByExecutedOrder()` with the retrieved JavaScript object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>haRecordSysId</td>
<td>String</td>
<td>The sys_id of the activity history (table wf_history).</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The ActivityHistoryRecord JavaScript object requested.</td>
</tr>
</tbody>
</table>

```javascript
var model = new WorkflowModelManager('7b3e01573b130000dada82c09ccf3dcf');
    model.getExecutedHistory();
    var current = model.getActivityHistoryRecordById('d6681d573b130000dada82c09ccf3d10');
    current.debugDump();

    var results = model.getNextByExecutedOrderId(current.sys_id);
    results.debugDump();
```

**WorkflowModelManager - getNextByTransitionId(String haRecordSysId)**

Retrieves the history activity that executed just after to the one identified by the `sys_id` provided in the argument. This function retrieves the cached history record associated with the provided `wf_history.sys_id` and then calls...
getNextByTransition. The return values are based on which transitions came before the haRecord submitted and not necessarily the activities that executed just prior to the haRecord in time. To get the activity that executed prior to this activity in time, use getNextByExecutedOrder.

This function retrieves the cached history record associated with the provided wf_history.sys_id and then calls getNextByTransition. The return values are based on which transitions came before the haRecord submitted and not necessarily the activities that executed just prior to the haRecord in time. To get the activity that executed prior to this activity in time, use getNextByExecutedOrder.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>haRecordSysId</td>
<td>String</td>
<td>sys_id of the activity history (table wf_history).</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Array</td>
<td>Array of ActivityHistoryRecord JavaScript objects.</td>
</tr>
</tbody>
</table>

```javascript
var model = new WorkflowModelManager('7b3e01573b130000dada82c09ccf3dcf');
    model.getExecutedHistory();
    model.playBack();

    var current = model.getActivityHistoryRecordById('d6681d573b130000dada82c09ccf3d10');
    current.debugDump();
    var results = model.getNextByTransitionId(current.sys_id);
    gs.print('COMPLETED NEXT' + results.length);
    for( var i = 0; i < results.length; i++){
        results[i].debugDump();
    }
```

WorkflowModelManager - getPreviousByExecutedOrder(Object haRecord)

Retrieves the history activity that executed just previous to the one provided in the argument.

The previous status is based on the activity index reflecting the nearest prior activity in time and not necessarily the nearest prior activity with a valid transition.
to this activity. To get the nearest prior activity that transitioned to this haRecord passed in use `getPreviousByTransition(haRecord)`.

<i>Note:</i> In the process of assembling the execution order array, blanks are left in the array where history objects have been deleted. This ensures that the index in the array and the activity_index of the object remain in sync. For that reason, all objects coming out of the execution order array should be tested for `nil()` and not assumed to be the previous record based on activity_index or array index value.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>haRecord</td>
<td>Object</td>
<td>ActivityHistoryRecord JavaScript object</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>The ActivityHistoryRecord JavaScript object requested.</td>
</tr>
</tbody>
</table>

**WorkflowModelManager - getPreviousByExecutedOrderId(String haRecordSysId)**

Retrieves the history activity cached by the `wf_history.sys_id` provided in the argument, then calls `getPreviousByExecutedOrder()` with the retrieved JavaScript object.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>haRecordSysId</td>
<td>String</td>
<td>Workflow history system ID (wf_history.sys_id)</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActivityHistoryRecord</td>
<td>The ActivityHistoryRecord JavaScript object requested.</td>
</tr>
</tbody>
</table>

```javascript
var model = new WorkflowModelManager('contextId');
model.getExecutedHistory();
var current = model.getActivityHistoryRecordBySysId('wf_history.sys_id');
```
WorkflowModelManager - getPreviousByTransition(Object haRecord)

Retrieves the history activities that executed just prior to the one provided in the argument.

The **next** status is based on the `wf_activity.sys_id` associated with the activity represented in the `haRecord` existing as a TO in a transition associated with any ActivityHistoryRecords that come before the `haRecord` in the execution sequence. (This differs from `getAllTransitionedIntoActivity`, which returns all TO transitions up to the given `haRecord` in the execution sequence.) The return value is a collection of ActivityHistoryRecords that identify the argument `haRecord.wfaId` as their TO activity. The return values are based on which transitions came before the `haRecord` submitted and not necessarily the activities that executed just prior to the `haRecord` in time. To get the activity that executed prior to this activity in time, use `getPreviousByExecutedOrderId`.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>haRecord</td>
<td>Object</td>
<td>ActivityHistoryRecord JavaScript object.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Array</td>
<td>Array of ActivityHistoryRecord JavaScript objects requested.</td>
</tr>
</tbody>
</table>

WorkflowModelManager - getPreviousByTransitionId(String haRecordSysId)

Retrieves the history activities that executed just prior to the one provided in the argument.

The **next** status is based on the `wf_activity.sys_id` associated with the activity represented in the `haRecord` existing as a TO in a transition associated with any ActivityHistoryRecords that come before the `haRecord` in the execution sequence. The return value is a collection of ActivityHistoryRecords that identify the argument `haRecord.wfaId` as their TO activity. The return values are based on which transitions came before the `haRecord` submitted and not necessarily the activities that executed just prior to the `haRecord` in time. To get the activity that executed prior to this activity in time, use `getPreviousByExecutedOrderId`. 

```javascript
current.debugDump();

var results = model.getPreviousByExecutedOrderId(current.sys_id);
results.debugDump();
```
not necessarily the activities that executed just prior to the haRecord in time. To get the activity that executed prior to this activity in time use getPreviousByExecutedOrder.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>haRecordSysId</td>
<td>String</td>
<td>The sys_id of the activity history (table wf_history).</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Array of ActivityHistoryRecord JavaScript objects requested.</td>
</tr>
</tbody>
</table>

**WorkflowModelManager - WorkflowModelManager(String contextId)**

Creates a new WorkflowModelManager object (constructor) within the specified context.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextId</td>
<td>String</td>
<td>The ID for the context in which you want the WorkflowModelManager created.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActivityHistoryRecord</td>
<td>The ActivityHistoryRecord JavaScript object requested.</td>
</tr>
</tbody>
</table>

var model = new WorkflowModelManager('MyContextId');

**WorkflowModelManagerAjax - Global**

Provides an AjaxProcessor wrapper for the WorkflowModelManager. This class derives from AbstractAjaxProcessor.

Use this class in the Workflow Debug context menus.
WorkflowModelManagerAjax - getExecutedHistory()

Puts the execution history into the return parameter based on the query string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

WorkflowModelManagerAjax - getFinalExecutedActivityIdList()

Puts the final execution activity list into the return parameter based on the query string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

WorkflowModelManagerAjax - getRolledBackActivityIdList()

Puts the rollback activity list into the return parameter based on the query string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
<tr>
<td>Returns</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td></td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**WorkflowScheduler - Global**

This code is executed in business rule *Set workflow scheduler script* on table *wf_workflow_schedule*.

**WorkflowScheduler - run()**

Starts the workflow specified in current, which is a record in table *wf_workflow_schedule*.

This method is called from the Business Rule Set *workflow scheduler script* on table *wf_workflow_schedule*.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>void</td>
</tr>
</tbody>
</table>

**WorkflowTimeline - Global**

Generates a view that shows a workflow context on the timeline.

**WorkflowTimeline-getitems()**

Generates the Ajax response for the timeline.

This method is called by the Prototype JavaScript Framework. See the *AbstractTimelineSchedulePage* script include for more details.
<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>void</td>
<td>Current XML node.</td>
<td></td>
</tr>
</tbody>
</table>

**XMLDocument2 - Scoped**

XMLDocument2 is a JavaScript Object wrapper for parsing and extracting XML data from an XML string.

Use this JavaScript class to create an object from an XML string, usually a return value from a web-service invocation, or the XML payload of ECC Queue. Using the XMLDocument2 object in a JavaScript business rule lets you query values from the XML elements and attributes directly.

An XML string has a tree structure, and the parts of the structure are called nodes. An XMLDocument2 object deals with two node types, element, and document element. An element node is a node with a name and possibly attributes and child nodes. A document-element node is the root node of the XML tree. It is the only node without a parent node.

**Scoped XMLDocument2 - createElement(String name)**

Creates and adds an element node to the current node. The element name is the string passed in as a parameter. The new element has no text child nodes.

<table>
<thead>
<tr>
<th>Parameters</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The new element's name.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>XMLNode</td>
<td>Current XML node.</td>
<td></td>
</tr>
</tbody>
</table>
```javascript
var xmlString = "<test>
  <one>
    <two att="xxx">abcd1234</two>
    <three boo="yah" att="yyy">1234abcd</three>
    <two>another</two>
  </one>
  <number>1234</number>
";

var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
xmlDoc.createElement("new2");

gs.info(xmlDoc);
```

**Output:**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<test>
  <one>
    <two att="xxx">abcd1234</two>
    <three att="yyy" boo="yah">1234abcd</three>
    <two>another</two>
  </one>
  <number>1234</number>
  <new2></new2>
</test>
```

**Scoped XMLDocument2 - createElementWithTextValue(String name, String value)**

Creates and adds an element node with a text child node to the current node.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the element to add.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Element's text value.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLNode</td>
<td>Current XML node.</td>
</tr>
</tbody>
</table>
var xmlString = "<test>" +
  "  <one>" +
  "    <two att="xxx">abcd1234</two>" +
  "    <three boo="yah" att="yyy">1234abcd</three>" +
  "    <two>another</two>" +
  "  </one>" +
  "  <number>1234</number>" +
"</test>";

var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
xmlDoc.createElementWithTextValue("new", "test");
gs.info(xmlDoc);

Output:

<?xml version="1.0" encoding="UTF-8"?><test>
  <one>
    <two att="xxx">abcd1234</two>
    <three att="yyy" boo="yah">1234abcd</three>
    <two>another</two>
  </one>
  <number>1234</number>
  <new>test</new>
</test>

Scoped XMLDocument2 - getDocumentElement()

Gets the document element node of the XMLdocument2 object. The document element node is the root node.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLNode</td>
<td>The document element.</td>
</tr>
</tbody>
</table>

var xmlString = "<test>" +
  "  <one>" +
"    <two att="xxx">abcd1234</two>" +
"    <three boo="yah" att="yyy">1234abcd</three>" +
"    <two>another</two>" +
"    </one>" +
"    <number>1234</number>" +
"    </test>";

var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
//returns the root node of the document tree.
var rootNode = xmlDoc.getDocumentElement();
gs.info(rootNode.getTextContent());

Output:

abcd1234    1234abcd    another    1234

Scoped XMLDocument2 - getFirstNode(String xPath)
Gets the first node in the specified xPath.

This method only supports a subset of predicate xPath expressions. For example, assume the following document:

<store>
  <resources company="ABC Inc">
    <resources type="servers" />
  </resources>
  <resources company="XYZ Inc">
    <resource type="bookstore">
      <book>
        <title>Windows</title>
        <price>10</price>
      </book>
      <book year="2009">
        <title>Harry Potter</title>
        <price>50</price>
      </book>
      <book year="1999">
        <title>Learning XML</title>
        <price>120</price>
      </book>
      <book year="2019">
        <title>Learning Java</title>
        <price>99</price>
      </book>
    </resource>
  </resources>
</store>
This method supports the following xPath expressions with predicates:

```
"/store/resources/resource[@type='bookstore']/book[@year='1999'"]
"/store/resources/resource[@type='bookstore']/book[@year]
"/store/resources/resource[@type='bookstore']/book[price > 100]
```

However, it does not support the following xPath expressions with predicates:

```
"/store/resources/resource[@type='bookstore']/book[2]
"/store/resources/resource[@type='bookstore']/book[last()]
"/store/resources/resource[@type='bookstore']/book[position()>2]
```

To work around this, use xPath without predicates, such as 
`"/store/resources/resource[@type='bookstore']/book"` and then filter the nodes in the script using the `getFirstNode()` and `getNextNode()` methods.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xPath</td>
<td>String</td>
<td>The xPath to obtain the node from.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLNode</td>
<td>The first node.</td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>
  <one>
    <two att="xxx">abcd1234</two>
    <three boo="yah" att="yyy">1234abcd</three>
    <two>another</two>
  </one>
  <number>1234</number>
</test>";
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
var foo = xmlDoc.getFirstNode('/test/one/two');
gs.info(foo.getTextContent());
```
Scoped XMLDocument2 - getNextNode(Object current)

Gets the node after the specified node.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>current</td>
<td>Object</td>
<td>The current node.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLNode</td>
<td>The next node.</td>
</tr>
</tbody>
</table>

```java
var xmlString = "<test>" +
    "  <one>" +
    "    <two att="xxx">abcd1234</two>" +
    "    <three boo="yah" att="yyy">1234abcd</three>" +
    "    <two>another</two>" +
    "  </one>" +
    "  <number>1234</number>" +
"</test>";
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
var foo = xmlDoc.getFirstNode('/test/one/two');
var foo2 = xmlDoc.getNextNode(foo);
gs.info(foo.getTextContent());
gs.info(foo2.getTextContent());
```

Output:

```
abcd1234
another
```

Scoped XMLDocument2 - getNode(String xPath)

Gets the node specified in the xPath.
Note: This method is intended for use with small documents since it parses the entire document to match the XPath expression. If the document is too large, an out of memory error may occur. For large documents, use the `getFirstNode(String xpath)` and `getNextNode(Object current)` methods, which are designed to stream through the document without parsing the entire document.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xpath</td>
<td>string</td>
<td>XPath of the node to obtain.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLNode</td>
<td>Current XML node.</td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>
  <one>
    <two att="xxx">abcd1234</two>
    <three boo="yah" att="yyy">1234abcd</three>
    <two>another</two>
  </one>
  <number>1234</number>
</test>";
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
var node = xmlDoc.getNode("/test/one/two");
gs.info(node);
```

Output:

```
<two att="xxx">abcd1234</two>
```

Scoped XMLDocument2 - getNodeText(String xpath)

Gets all the text child nodes from the node referenced in the specified XPath.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xpath</td>
<td>string</td>
<td>XPath of the text to obtain.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Text children in the XPath.</td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>" +
    "  <one>" +
    "    <two att="xxx">abcd1234</two>" +
    "    <three boo="yah" att="yyy">1234abcd</three>" +
    "    <two>another</two>" +
    "  </one>" +
    "  <number>1234</number>" +
"</test>";
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
gs.info(xmlDoc.getNodeText("//two"));
```

Output: abcd1234

**Scoped XMLDocument2 - parseXML(String xmlDoc)**

Parses the XML string and loads it into the XMLDocument2 object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xmlDoc</td>
<td>String</td>
<td>The document to parse.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Flag that indicates whether the content was parsed.</td>
</tr>
</tbody>
</table>

This example parses the xmlString and loads it into the xmlDocument2 object.
Scoped XMLDocument2 - setCurrentElement(XMLNode element)

Makes the node passed in as a parameter the current node.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>element</td>
<td>XMLNode</td>
<td>The element node to set as the current node.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>" +
   "  <one>" +
   "    <two att="xxx">abcd1234</two>" +
   "    <three boo="yah" att="yyy">1234abcd</three>" +
   "    <two>another</two>" +
   "  </one>" +
   "  <number>1234</number>" +
"</test>";
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
//returns the root node of the document tree.
var rootNode = xmlDoc.getDocumentElement(); //returns org.w3c.dom.Element
// sets the root node as the current element
xmlDoc.setCurrentElement(rootNode);
```

Scoped XMLDocument2 - setNamespaceAware(Boolean aware)

When set to true, the XMLDocument2 object processes the document with XML namespaces.

If you don't set this, an XML document with namespaces won't be enumerated correctly, and an XPath search would fail.
## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aware</td>
<td>Boolean</td>
<td>When true, the XMLDocument2 object processes the document with XML namespaces.</td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### Scoped XMLDocument2 - toString()

Returns a string containing the XML.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>String</td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>
    <one>
        <two att="xxx">abcd1234</two>
        <three boo="yah" att="yyy">1234abcd</three>
        <two>another</two>
    </one>
    <number>1234</number>
</test>";
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
gs.info(xmlDoc.toString());
```

**Output - Line breaks were added to the output for readability.**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<test>
</test>
```
<one>
<two att="xxx">abcd1234</two>
<three att="yyy" boo="yah">1234abcd</three>
<two>another</two>
</one>
<number>1234</number>
</test>

Scoped XMLDocument2 - XMLDocument2()

Creates an XMLDocument2 object.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

Scoped XMLDocument2 - XMLDocument2( GlideScriptableInputStream inputStream)

Creates an XMLDocument2 object from an attachment stream.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputStream</td>
<td>GlideScriptableInputStream</td>
<td>The input stream the XMLDocument2 object encapsulates.</td>
</tr>
</tbody>
</table>

DOMNode - Scoped

The scoped XMLNode API allows you to query values from XML nodes. XMLNodes are extracted from XMLDocument2 objects, which contain XML strings.

There are no constructors for creating a stand alone instance of an XMLNode object. Instead, use the createElement() method of XMLDocument2, which adds a node to an existing document.

Scoped XMLNode - getAttribute(String attribute)

Gets the value of the attribute.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attribute</td>
<td>String</td>
<td>Name of the attribute.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The attribute’s value.</td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>" +
    "  <one>" +
    "    <two att="xxx">abcd1234</two>" +
    "    <three boo="yah" att="yyy">1234abcd</three>" +
    "    <two>another</two>" +
    "  </one>" +
    "  <number>1234</number>" +
"</test>";

var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
var node = xmlDoc.getNode('//two');
gs.info(node.getAttribute('att'));
```

Output: xxx

**Scoped XMLNode - getAttributes()**

Returns an object containing the node’s attributes as properties with values.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Contains name-value pairs where the name is the attribute and the value is the attribute’s value.</td>
<td></td>
</tr>
</tbody>
</table>
**Scoped XMLNode - getChildNodeIterator()**  
Gets a XMLNodeIterator object that can be used to walk through the list of child nodes.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLNodeIterator</td>
<td>The node iterator object.</td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>" +
  "  <one>" +
  "    <two att="xxx">abcd1234</two>" +
  "    <three boo="yah" att="yyy">1234abcd</three>" +
  "    <two>another</two>" +
  "  </one>" +
  "  <number>1234</number>" +
  "</test>";
var xmlDoc = new XMLDocument2();
xmldoc.parseXML(xmlString);
var node = xmlDoc.getNode('//one');
var iter = node.getChildNodeIterator();
gs.info(iter.hasNext());
```

**Scoped XMLNode - getFirstChild()**  
Gets the node's first child node.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLNode</td>
<td>The node's first child node.</td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>" +
    "<one>" +
    "<two att="xxx">abcd1234</two>" +
    "<three boo="yah" att="yyy">1234abcd</three>" +
    "<two>another</two>" +
    "</one>" +
    "<number>1234</number>" +
    "</test>";
var xmlDoc = new XMLDocument2();
xmDoc.parseXML(xmlString);
var node = xmlDoc.getNode('//one');
gs.info(node.getFirstChild());
```

Output:

```
<two att="xxx">abcd1234</two>
```

Scoped XMLNode - getLastChild()

Gets the node's last child node.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLNode</td>
<td>The node's last child.</td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>" +
    "<one>" +
    "<two att="xxx">abcd1234</two>" +
    "<three boo="yah" att="yyy">1234abcd</three>" +
    "<two>another</two>" +
    "</one>" +
    "<number>1234</number>" +
    "</test>";
var xmlDoc = new XMLDocument2();
xmDoc.parseXML(xmlString);
var node = xmlDoc.getNode('//one');
gs.info(node.getLastChild());
```
"<number>1234</number>" +
"</test>");
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
var node = xmlDoc.getNode('//one');
gs.info(node.getLastChild());

**Output:**

<two>another</two>

### Scoped XMLNode - getNodeName()

Gets the node's name. A node's name is determined by the node type. A document-element node's name is #document. A text node's name is #text. An element node's name is the element's name.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>Returns</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The node's name.</td>
</tr>
</tbody>
</table>

---

```
var xmlString = "<test>" +
    " <one>" +
    " <two att="xxx">abcd1234</two>" +
    " <three boo="yah" att="yyy">1234abcd</three>" +
    " <two>another</two>" +
  " </one>" +
    " <number>1234</number>" +
"</test>");
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
var node = xmlDoc.getNode('//two');
gs.info(node.getNodeName());
```

**Output:** two
Scoped XMLNode - getNodeValue()

Gets the node's value. A node's value is determined by the node type. Element and document-element nodes return null.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The node's value.</td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>" +
  "  <one>" +
  "    <two att="xxx">abcd1234</two>" +
  "    <three boo="yah" att="yyy">1234abcd</three>" +
  "    <two>another</two>" +
  "  </one>" +
  "  <number>1234</number>" +
"</test>";
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
var node = xmlDoc.getNode('//two');
gs.info(node.getNodeValue());
```

Output: null

Scoped XMLNode - getTextContent()

Gets the text content of the current node. The text content of a node consists of all the node's child text nodes

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The text content of the current node.</td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>" +
  "  <one>" +
  "    <two att="xxx">abcd1234</two>" +
  "    <three boo="yah" att="yyy">1234abcd</three>" +
  "    <two>another</two>" +
  "  </one>" +
  "  <number>1234</number>" +
"</test>";
var xmldoc = new XMLDocument2();
xmldoc.parseXML(xmlString);
var node = xmldoc.getNode('//one/two');
gs.info(node.getTextContent());
```

Output: abcd1234

**Scoped XMLNode - hasAttribute(String attribute)**

Determines if the node has the specified attribute.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attribute</td>
<td>String</td>
<td>The name of the attribute to check.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the node has the attribute.</td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>" +
  "  <one>" +
  "    <two att="xxx">abcd1234</two>" +
  "    <three boo="yah" att="yyy">1234abcd</three>" +
  "    <two>another</two>" +
  "  </one>" +
  "  <number>1234</number>" +
"</test>";
```
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
var node = xmlDoc.getNode('//two');
gs.info(node.hasAttribute('att'));

Output: true

Scoped XMLNode - toString()
Returns the string value of the current node.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The string value of the current node.</td>
</tr>
</tbody>
</table>

var xmlString = "<test>
  <one>
    <two att="xxx">abcd1234</two>
    <three att="yyy">1234abcd</three>
    <two>another</two>
  </one>
  <number>1234</number>
</test>";
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
var node = xmlDoc.getNode('//one');
gs.info(node.toString());

Output: Line breaks were added to the output.

<one>
  <two att="xxx">abcd1234</two>
  <three att="yyy" boo="yah">1234abcd</three>
  <two>another</two>
</one>
XMLNodeIterator - Scoped

The scoped XMLNodeIterator class allows you to iterate through a node of a XML document.

There are no constructors for creating a stand alone instance of a XMLNodeIterator object. To create a XMLNodeIterator object use the getChildNodeIterator() method of the XMLNode object.

Scoped XMLNodeIterator - hasNext()

Returns true if the iteration has more elements.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>True if the iteration has more elements.</td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>" +
  "  <one>" +
    "    <two att="xxx">abcd1234</two>" +
    "    <three boo="yah" att="yyy">1234abcd</three>" +
    "    <two>another</two>" +
  "  </one>" +
  "  <number>1234</number>" +
"</test>";
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
var node = xmlDoc.getNode('//one');
var iter = node.getChildNodeIterator();
gs.info(iter.hasNext());
```

Scoped XMLNodeIterator - next()

Gets the next element in the iteration. The returned element may be a #text node for the spaces/tabs if XML is "pretty formatted". 
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLNode</td>
<td>The next element in the iteration.</td>
</tr>
</tbody>
</table>

```javascript
var xmlString = "<test>" +
  "  <one>" +
  "    <two att="xxx">abcd1234</two>" +
  "    <three boo="yah" att="yyy">1234abcd</three>" +
  "    <two>another</two>" +
  "  </one>" +
  "  <number>1234</number>" +
  "</test>";
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(xmlString);
var node = xmlDoc.getNode('//one');
var iter= node.getChildNodeIterator();
while(iter.hasNext()) {
  var n = iter.next();
  gs.info('Node name: ' +  n.getNodeName());
  gs.info('Node value: ' +  n.getNodeValue());
}
```

### Output:

```
Node name: #text
Node value:
Node name: two
Node value: null
Node name: #text
Node value:
Node name: three
Node value: null
Node name: #text
Node value:
Node name: two
Node value: null
```
**XMLStreamingAPI - Scoped**

Build a large streaming XML payload for use in a REST or SOAP request to send bulk data to a third-party API. You can also create the payload as an XML string for a non-streaming option.

Use these methods in a Flow Designer Script step with the `sn_ih` namespace identifier. For example, you can use this API to create an XML payload in the Flow Designer Script step and pass the returned value to the REST step to send the request to a third-party service. For more information, see the Flow Designer Script step.

ℹ️ **Note**: You can only use this API within the Flow Designer environment.

There is no constructor for this class. Instead, you must call the `build()` method in the `XMLStreamingBuilder` class to return an `XMLStreamingAPI` object. For more information, see `XMLStreamingBuilder`.

**API call order**

Generate XML payloads by first instantiating a builder object with `XMLStreamingBuilder` and then calling the methods in the `XMLStreamingAPI` class:

1. **XMLStreamingBuilder**: Creates a builder object

   Use these method in the following order to create a builder object:

   1. `XMLStreamingBuilder()`: Instantiates the `XMLStreamingBuilder` object.
   2. `withAttachment()`: Optional. Creates an XML document as an attachment and stores it in the Streaming Attachments `[streaming_attachment]` table. If you don’t call this method, the API builds the payload as an XML string.
   3. `expiresAt()`: Optional. Sets a time when the attachment expires. False is the default. Must also call the `withAttachment()` method.
   4. `build()`: Returns an `XMLStreamingAPI` object.

2. **XMLStreamingAPI**: Builds the XML payload

   Use these methods in the following order to create the XML payload:
1. `startDocument()`: Creates the top-level parent element in the XML document.

2. Methods to generate child elements in the XML document, such as `writeTextElement()`, `startElement()`, and `writeArray()`.

3. Methods to generate attributes for an element, such as `writeAttribute()`, `writeNamespace()`, and `writeDtd()`.

4. `endElement()`: Closes an XML element.

5. `endDocument`: Closes the top-level parent element.

6. `getXMLString()` or `getAttachmentId()`: Returns the XML string or attachment ID that you created.

7. `close()`: Closes the XMLStreamingAPI object.

**Size limits**

Payloads generated through this API cannot exceed these size limits:

- Attachments: 200 MB
- Strings: 5 MB

The following example shows how to create an XML document and store it in the Streaming Attachments [streaming_attachment] table with a defined expiration date.

```javascript
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.XMLStreamingBuilder()
    .withAttachment() // Creates the XML document in streaming mode within an attachment.
    .expiresAt(ttl) // Sets an expiration date for the attachment.
    .build(); // Creates the XMLStreamingAPI object.

    builder.startDocument() // Begins generating the XML document.
    .writeTextElement("firstName","John") // Writes a "firstName" element and value.
    .writeTextElement("lastName","Smith")
    .writeTextElement("age","25")
    .startElement("address") // Adds an "address" parent element.
    .writeTextElement("streetAddress", "21 2nd Street") // Writes a child element and value.
    .writeTextElement("city", "Santa Clara")
    .writeTextElement("state", "CA")
    .writeTextElement("postalCode", "11111")
    .endElement() // Adds a closing tag for the "address" element.
}
```
Alternatively, this example shows how to use the API in the Script step and create the payload as an XML string. You can use this option to create payloads under 5 MB.

```javascript
(function execute(inputs, outputs) {

  var builder = new sn_ih.XMLStreamingBuilder().build();

  builder.startDocument()
    .enablePrettyPrint()
    .writeTextElement("firstName","John")
    .writeTextElement("lastName","Smith")
    .writeTextElement("age","25")
    .startElement("address")
      .writeTextElement("streetAddress", "21 2nd Street")
      .writeTextElement("city", "Santa Clara")
      .writeTextElement("state", "CA")
      .writeTextElement("postalCode", "11111")
    .endElement()
    .startElement("phoneNumber")
      .writeTextElement("type","home")
      .writeTextElement("number","212 555-1234")
      .writeTextElement("type","fax")
      .writeTextElement("number","646 555-4567")
    .endElement()
  .endDocument()

  outputs.payload = builder.getXMLString();
}
```
Output:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<firstName>John</firstName>
<lastName>Smith</lastName>
<age>25</age>
<address>
  <streetAddress>21 2nd Street</streetAddress>
  <city>Santa Clara</city>
  <state>CA</state>
  <postalCode>11111</postalCode>
</address>
<phoneNumber>
  <type>home</type>
  <number>212 555-1234</number>
  <type>fax</type>
  <number>646 555-4567</number>
</phoneNumber>
```

**XMLStreamingAPI - close()**

Closes the XMLStreamingAPI object. You must call this method to close the stream after building your XML document.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Example**

The following example shows how to create an XML document and store it in the Streaming Attachments [streaming_attachment] table with a defined expiration date.

```java
try {
```
```javascript
var ttl = new GlideDateTime("2011-01-01 12:00:00");
var builder = new sn_ih.XMLStreamingBuilder()
    .withAttachment() // Creates the XML document in streaming mode within an attachment.
    .expiresAt(ttl) // Sets an expiration date for the attachment.
    .build(); // Creates the XMLStreamingAPI object.

builder.startDocument() // Begins generating the XML document.
    .writeTextElement("firstName","John") // Writes a "firstName" element and value.
    .writeTextElement("lastName","Smith")
    .writeTextElement("age","25")
    .startElement("address") // Adds an "address" parent element.
        .writeTextElement("streetAddress", "21 2nd Street") // Writes a child element and
        value.
        .writeTextElement("city", "Santa Clara")
        .writeTextElement("state", "CA")
        .writeTextElement("postalCode", "11111")
    .endElement() // Adds a closing tag for the "address" element.
    .startElement("phoneNumber")
        .writeTextElement("type","home")
        .writeTextElement("number","212 555-1234")
        .writeTextElement("type","fax")
        .writeTextElement("number","646 555-4567")
    .endElement()
.endDocument() // Stops generating the XML document.

gs.log(builder.getAttachmentId()); // Returns the sys_id of the attachment.
} catch (err) {
    gs.log(err);
} finally {
    builder.close();
}
```

**XMLStreamingAPI - disablePrettyPrint()**

Ends pretty print XML formatting.

Before calling this method, you must first call `enablePrettyPrint()` to add XML formatting to a section.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>

The following example shows how to add pretty print formatting to the address element.

```javascript
try {
  var ttl = new GlideDateTime("2011-01-01 12:00:00");
  var builder = new sn_ih.XMLStreamingBuilder()
    .withAttachment()
    .expiresAt(ttl)
    .build();

  builder.startDocument()
    .writeTextElement("firstName", "John")
    .writeTextElement("lastName", "Smith")
    .writeTextElement("age", "25")
    .enablePrettyPrint()
    .startElement("address")
      .writeTextElement("streetAddress", "21 2nd Street")
      .writeTextElement("city", "Santa Clara")
      .writeTextElement("state", "CA")
      .writeTextElement("postalCode", "11111")
    .endElement()
    .disablePrettyPrint()
    .startElement("phoneNumber")
      .writeTextElement("type", "home")
      .writeTextElement("number", "212 555-1234")
      .writeTextElement("type", "fax")
      .writeTextElement("number", "646 555-4567")
    .endElement()
  .endDocument()

  gs.log(builder.getAttachmentId());
} catch (err) {
  gs.log(err);
} finally {
  builder.close();
}
```
XMLStreamingAPI - enablePrettyPrint()

Adds pretty print formatting to an XML element or tree of elements.

Use the disablePrettyPrint() method to end the formatting.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>

The following example shows how to add pretty print formatting to the address element.

```javascript
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.XMLStreamingBuilder()
        .withAttachment()
        .expiresAt(ttl)
        .build();

    builder.startDocument()
        .writeTextElement("firstName","John")
        .writeTextElement("lastName","Smith")
        .writeTextElement("age","25")
        .enablePrettyPrint()
        .startElement("address")
            .writeTextElement("streetAddress", "21 2nd Street")
            .writeTextElement("city", "Santa Clara")
            .writeTextElement("state", "CA")
            .writeTextElement("postalCode", "11111")
        .endElement()
        .disablePrettyPrint()
        .startElement("phoneNumber")
            .writeTextElement("type","home")
            .writeTextElement("number","212 555-1234")
            .writeTextElement("type","fax")
            .writeTextElement("number","646 555-4567")
} catch (err) {
    console.err(err);}
```
XMLStreamingAPI - endDocument()

Ends the structure of your XML document.

After calling the `startDocument()` method and organizing your streaming XML document, call the `endDocument()` method at the end of your document's structure. You must use these two methods together to successfully build your streaming XML document's structure.

## Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>

The following example shows how to create an XML document containing elements about a user.

```javascript
try {
  var ttl = new GlideDateTime('2011-01-01 12:00:00');
  var builder = new sn_ih.XMLStreamingBuilder().withAttachment().expiresAt(ttl);
  var streamingDocument = builder.build();

  streamingDocument.startDocument('Employee')
    .writeTextElement('firstName', 'John')
    .writeTextElement('lastName', 'Smith')
    .writeTextElement('age', '25')
    .endDocument();
  gs.log(streamingDocument.getAttachmentId());
}
```
XMLStreamingAPI - endElement()

Adds a closing tag to an XML element.

Use the following methods in this sequence to create a valid XML element:

1. Use the startElement() method to add a starting tag.
2. Use the endElement() method to add the closing tag.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>

The following example shows how to build a parent element named address and then write four child elements.

```javascript
try {
  var ttl = new GlideDateTime("2011-01-01 12:00:00");
  var builder = new sn_ih.XMLStreamingBuilder()
    .withAttachment()
    .expiresAt(ttl)
    .build();

  builder.startDocument()
    .startElement("address")
    .writeTextElement("streetAddress", "21 2nd Street")
    .writeTextElement("city", "Santa Clara")
    .writeTextElement("state", "CA")
    .writeTextElement("postalCode", "11111")
    .endElement()
  .endDocument();
}
```
XMLStreamingAPI - getXMLString()

Returns the XML document as a string.

To return the XML document as a string, don't call the getAttachmentId() method in the XMLStreamingBuilder class. For more information, see XMLStreamingBuilder - Scoped.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>XML document created using the XMLStreamingAPI methods, as a string.</td>
</tr>
</tbody>
</table>

The following example shows how to create an XML document and then return it as a string.

```javascript
(function execute(inputs, outputs) {

  var builder = new sn_ih.XMLStreamingBuilder().build();

  builder.startDocument()
    .enablePrettyPrint()
    .writeTextElement("firstName", "John")
    .writeTextElement("lastName", "Smith")
    .writeTextElement("age", "25")
    .startElement("address")
      .writeTextElement("streetAddress", "21 2nd Street")
      .writeTextElement("city", "Santa Clara")
      .writeTextElement("state", "CA")
      .writeTextElement("postalCode", "11111")
}
```
Output:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<firstName>John</firstName>
<lastName>Smith</lastName>
<age>25</age>
<address>
    <streetAddress>21 2nd Street</streetAddress>
    <city>Santa Clara</city>
    <state>CA</state>
    <postalCode>11111</postalCode>
</address>
<phoneNumber>
    <type>home</type>
    <number>212 555-1234</number>
    <type>fax</type>
    <number>646 555-4567</number>
</phoneNumber>
```

**XMLStreamingAPI - startDocument(String rootElement, Object namespaceDefinitionMap)**

Begins building an XML document.

After calling the `build()` method, call the `startDocument()` method to start organizing your XML document. You must also call the `endDocument` method at the end of your document's structure.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rootElement</td>
<td>String</td>
<td>Optional. Root element, or top-level parent element, for your XML document.</td>
</tr>
<tr>
<td>namespaceDefinitionMap</td>
<td>Object</td>
<td>Optional. Map of keys and values for the namespaces and their associated values in a subsequent list of elements. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{ 'namespaceOne':'namespaceValue', 'namespaceTwo':'namespaceValue' }</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>

The following example shows how to create an XML document containing elements with information about an employee.

```javascript
try {
    var ttl = new GlideDateTime('2011-01-01 12:00:00');
    var builder = new sn_ih/XMLStreamingBuilder().withAttachment().expiresAt(ttl);
    var streamingDocument = builder.build();

    streamingDocument.startDocument('Employee')
        .writeTextElement('firstName', 'John')
        .writeTextElement('lastName', 'Smith')
        .writeTextElement('age', '25')
        .endDocument();
    gs.log(streamingDocument.getAttachmentId());
} catch (err) {
    gs.log(err);
} finally {
    streamingDocument.close();
}
```
XMLStreamingAPI - startElement(String name, Object namespaceMap, Object attributeMap, String prefix)

Adds a starting tag for an XML element.

Use the following methods in this sequence to create a valid XML element:

1. Use the `startElement()` method to add a starting tag.
2. Use the `endElement()` method to add the closing tag.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the XML element.</td>
</tr>
<tr>
<td>namespaceMap</td>
<td>Object</td>
<td>Optional. Map of keys and values for the namespaces and their associated values in a subsequent list of elements. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'namespaceOne':'namespaceValue',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'namespaceTwo':'namespaceValue'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td>attributeMap</td>
<td>Object</td>
<td>Optional. Map of keys and values for the attributes and their associated values in a subsequent list of elements.</td>
</tr>
<tr>
<td>prefix</td>
<td>String</td>
<td>Optional. Prefix for the XML element.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>

The following example shows how to build a parent element named `address` and then write four child elements.

```java
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.XMLStreamingBuilder()
        .withAttachment()
        .expiresAt(ttl)
        .build();
```
XMLStreamingAPI - `writeArray(String elementName, Array data, String wrappingElement)`

Adds a list of nested elements with predefined text to your streaming XML document.

After calling the `startDocument()` method, you can call the `writeArray()` method to add a block of nested elements to your streaming XML document.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>elementName</code></td>
<td>String</td>
<td>Name of the XML element associated with each string listed in the <code>data</code> array.</td>
</tr>
<tr>
<td><code>data</code></td>
<td>Array</td>
<td>List of values to assign to each element nested inside <code>wrappingElement</code>.</td>
</tr>
<tr>
<td><code>wrappingElement</code></td>
<td>String</td>
<td>Parent element containing each <code>elementName</code>.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>

The following example shows how to build a parent element named `officeLocations`, and then nest an array of five `city` elements.
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.XMLStreamingBuilder()
        .withAttachment()
        .expiresAt(ttl)
        .build();

    builder.startDocument()
        .writeArray('city', ['Santa Clara','San Diego','Chicago','Sydney','London'],
        'officeLocations')
        .endDocument()

    gs.log(builder.getAttachmentId());
} catch (err) {
    gs.log(err);
} finally {
    builder.close();
}

**XMLStreamingAPI - writeAttribute(String name, String value)**

Adds an attribute to an element in your XML document.

After calling the `startDocument()`, `startElement()`, or `writeTextElement()` method, you can call the `writeAttribute()` method to add an attribute to the associated XML element.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the XML element’s attribute.</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>Value for the XML element’s attribute.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>

This example adds an attribute named `idNumber` to the `employee` element.

```javascript
try {
    var ttl = new GlideDateTime('2011-01-01 12:00:00');
    var builder = new sn_ih.XMLStreamingBuilder().withAttachment().expiresAt(ttl);
    builder.startDocument()
        .writeArray('city', ['Santa Clara','San Diego','Chicago','Sydney','London'],
        'officeLocations')
        .endDocument()
    gs.log(builder.getAttachmentId());
} catch (err) {
    gs.log(err);
} finally {
    builder.close();
}
```
var streamingDocument = builder.build();

streamingDocument.startDocument('Employee')
  .writeAttribute('idNumber', '12345')
  .writeTextElement('firstName', 'John')
  .writeTextElement('lastName', 'Smith')
  .writeTextElement('age', '25')
  .endDocument();

gs.log(streamingDocument.getAttachmentId());
} catch (err) {
  gs.log(err);
} finally {
  streamingDocument.close();
}

**XMLStreamingAPI - writeAttributes(Object attributeMap)**

Adds attributes to an element in your XML document.

After calling the `startDocument()`, `startElement()`, or `writeTextElement()` method, you can call the `writeAttributes()` method to add attributes to the associated XML element.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributeMap</td>
<td>Object</td>
<td>Map of keys and values containing attribute names and values to associate with the XML element. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>

This example adds attributes named `idNumber`, `officeLocation`, and `department` to the `employee` element.
try {
    var ttl = new GlideDateTime('2011-01-01 12:00:00');
    var builder = new sn_ih.XMLStreamingBuilder().withAttachment().expiresAt(ttl);
    var streamingDocument = builder.build();
    streamingDocument.startDocument('Employee')
        .writeAttributes({'idNumber':'12345', 'officeLocation':'San Diego',
                         'department':'Sales'})
        .writeTextElement('firstName', 'John')
        .writeTextElement('lastName', 'Smith')
        .writeTextElement('age', '25')
        .endDocument();
    gs.log(streamingDocument.getAttachmentId());
} catch (err) {
    gs.log(err);
} finally {
    streamingDocument.close();
}

XMLStreamingAPI - writeCData(String data)

Adds CDATA to your XML document.

After calling the writeCDataElement() method, you can call the writeCData() method to add CDATA within the element.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>String</td>
<td>Value to include after the CDATA keyword in your CDATA element.</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>

This example adds CDATA to the CDATA element timeWorked.

try {
    var ttl = new GlideDateTime('2011-01-01 12:00:00');
    var builder = new sn_ih.XMLStreamingBuilder().withAttachment().expiresAt(ttl);
    var streamingDocument = builder.build();
}
streamingDocument.startDocument('Employee')
    .writeTextElement('firstName', 'John')
    .writeTextElement('lastName', 'Smith')
    .writeTextElement('age', '25')
    .writeCDataElement('timeWorked')
    .writeCData('< 2 years')
    .endDocument();
gs.log(streamingDocument.getAttachmentId());
} catch (err) {
    gs.log(err);
} finally {
    streamingDocument.close();
}

XMLStreamingAPI - writeCDataElement(String name, String data, Object prefix)
Adds a CDATA element to your XML document.

After calling the startDocument() method, you can call the writeCDataElement() method to add a CDATA element to your XML document.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the CDATA element.</td>
</tr>
<tr>
<td>data</td>
<td>String</td>
<td>Optional. Type of data to parse the CDATA element as.</td>
</tr>
<tr>
<td>prefix</td>
<td>Object</td>
<td>Optional. Map of child elements and values that the CDATA element includes. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'prefixOne':'prefixValue',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'prefixTwo':'prefixValue'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You must associate an XML element's prefix with a namespace using writeNamespace().</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>
This example uses a document type definition named `address` to define an internal document type definition for the XML document.

```javascript
try {
  var ttl = new GlideDateTime("2011-01-01 12:00:00");
  var builder = new sn_ih.XMLStreamingBuilder()
    .withAttachment()
    .expiresAt(ttl)
    .build();

  builder.startDocument()
    .writeDtd('address')
    .writeCDataElement('home', '#PCDATA', {'streetAddress', 'city', 'state', 'postalCode'})
      .writeTextElement("streetAddress", "21 2nd Street")
      .writeTextElement("city", "Santa Clara")
      .writeTextElement("state", "CA")
      .writeTextElement("postalCode", "11111")
    .endDocument()

  gs.log(builder.getAttachmentId());
} catch (err) {
  gs.log(err);
} finally {
  builder.close();
}
```

**XMLStreamingAPI - writeCharacters(String text)**

Adds text to your XML document.

Use the `writeCharacters()` method to insert string data to a section in your XML document.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>Text to add to a section of your XML document.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>
The following example shows how to add text values to elements in your XML document.

```javascript
try {
    var ttl = new GlideDateTime('2011-01-01 12:00:00');
    var builder = new sn_ih.XMLStreamingBuilder().withAttachment().expiresAt(ttl);
    var streamingDocument = builder.build();

    streamingDocument.startDocument('Employee')
       .startElement('firstName')
       .writeCharacters('John')
       .EndElement()
       .startElement('lastName')
       .writeCharacters('Smith')
       .EndElement()
       .endDocument();
    gs.log(streamingDocument.getAttachmentId());
} catch (err) {
    gs.log(err);
} finally {
    streamingDocument.close();
}
```

**XMLStreamingAPI - writeComment(String comment)**

Adds a comment to your XML document.

After calling the `startDocument()` method, you can call the `writeComment()` method to add a comment to your XML document.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>comment</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
</tr>
<tr>
<td>XMLStreamingAPI</td>
</tr>
</tbody>
</table>

The following example shows how to add a comment to an XML document.

```javascript
try {
    var ttl = new GlideDateTime('2011-01-01 12:00:00');
```
var builder = new sn_ih.XMLStreamingBuilder().withAttachment().expiresAt(ttl);
var streamingDocument = builder.build();

streamingDocument.startDocument('Employee')
    .writeComment('Element for information related to active employees.')
    .writeTextElement('firstName', 'John')
    .writeTextElement('lastName', 'Smith')
    .writeTextElement('age', '25')
    .endDocument();
gs.log(streamingDocument.getAttachmentId());
} catch (err) {
    gs.log(err);
} finally {
    streamingDocument.close();
}

**XMLStreamingAPI - writeDtd(String dtd)**

Adds a document type definition to your XML document.

After calling the `startDocument()` method, you can call the `writeDtd()` method to add a valid XML document type definition to your XML document.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dtd</td>
<td>String</td>
<td>Name of a valid XML document type definition.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>

This example uses a document type definition named *address* to define an internal document type definition for the XML document.

```javascript
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.XMLStreamingBuilder()
        .withAttachment()
        .expiresAt(ttl)
        .build();
```
XMLStreamingAPI - writeNamespace(String prefix, String namespaceURI)

Adds a namespace to an element in your XML document.

After calling the startDocument(), startElement(), or writeTextElement() method, you can call the writeNamespace() method to add a namespace to the associated XML element.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>prefix</td>
<td>String</td>
<td>Prefix for the XML namespace.</td>
</tr>
<tr>
<td>namespaceURI</td>
<td>String</td>
<td>Optional. URI for the namespace.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>

The following example shows how to add a namespace and URI to the company root element, and then assign the prefix to the nested companyName element.

```javascript
try {
  var ttl = new GlideDateTime('2011-01-01 12:00:00');
  var builder = new sn_ih.XMLStreamingBuilder().withAttachment().expiresAt(ttl);
  var streamingDocument = builder.build();
...
```javascript
streamingDocument.startDocument('company')
   .writeNamespace('x','https://www.servicenow.com')
   .writeTextElement('companyName', 'ServiceNow')
   .writeNamespace('x')
   .endDocument();
gs.log(streamingDocument.getAttachmentId());
}
```

**XMLStreamingAPI - writeNamespaces(Object namespaceMap)**

Adds namespaces to the root element in your XML document.

After calling the `startDocument()` or `startElement()` method, you can call the `writeNamespaces()` method to declare namespaces for the associated XML element.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>namespaceMap</td>
<td>Object</td>
<td>Map of keys and values containing namespace prefixes and URIs to associate with the root element of the XML document. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{ 'namespaceOne': 'namespaceValue', 'namespaceTwo': 'namespaceValue' }</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>

This example adds two namespaces and URIs to the `company` root element, and then assigns the appropriate prefixes to the nested elements.

```javascript
try {
    var ttl = new GlideDateTime('2011-01-01 12:00:00');
    var builder = new sn_ih.XMLStreamingBuilder().withAttachment().expiresAt(ttl);  
    var streamingDocument = builder.build();
}
```
XMLStreamingAPI - writeTextElement(String name, String text, Object prefix)

Adds a single XML element to your XML document.

After calling the `startDocument()` method, you can call the `writeTextElement()` method to add a single XML element to your XML document’s structure.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of the XML element.</td>
</tr>
<tr>
<td>text</td>
<td>String</td>
<td>Value for the XML element.</td>
</tr>
<tr>
<td>prefix</td>
<td>Object</td>
<td>Optional. Map of prefixes and values associated with the XML element. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>`{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'prefixOne': 'prefixValue',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'prefixTwo': 'prefixValue'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

You must associate an XML element's prefix with a namespace using `writeNamespace()`.

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingAPI</td>
<td>Streaming XML object for constructing the payload.</td>
</tr>
</tbody>
</table>
The following example shows how to create an XML document containing three elements with information about an employee.

```javascript
try {
    var ttl = new GlideDateTime('2011-01-01 12:00:00');
    var builder = new sn_ih.XMLStreamingBuilder().withAttachment().expiresAt(ttl);
    var streamingDocument = builder.build();

    streamingDocument.startDocument('Employee')
        .writeTextElement('firstName', 'John')
        .writeTextElement('lastName', 'Smith')
        .writeTextElement('age', '25')
        .endDocument();
    gs.log(streamingDocument.getAttachmentId());
} catch (err) {
    gs.log(err);
} finally {
    streamingDocument.close();
}
```

**XMLStreamingBuilder - Scoped**

Create a builder object to build a large XML payload for use in a REST or SOAP request to send bulk data to a third-party API. You can also create the payload as an XML string for a non-streaming option.

Use these methods in a Flow Designer Script step with the `sn_ih` namespace identifier. For example, you can use this API to create an XML payload in the Flow Designer Script step and pass the returned value to the REST step to send the request to a third-party service. For more information, see the Flow Designer Script step.

⚠️ **Note:** You can only use this API within the Flow Designer environment.

**API call order**

Generate XML payloads by first instantiating a builder object with `XMLStreamingBuilder` and then calling the methods in the `XMLStreamingAPI` class:

1. `XMLStreamingBuilder`: Creates a builder object

   Use these method in the following order to create a builder object:
1. XMLStreamingBuilder(): Instantiates the XMLStreamingBuilder object.

2. withAttachment(): Optional. Creates an XML document as an attachment and stores it in the Streaming Attachments [streaming_attachment] table. If you don't call this method, the API builds the payload as an XML string.

3. expiresAt(): Optional. Sets a time when the attachment expires. False is the default. Must also call the withAttachment() method.

4. build(): Returns an XMLStreamingAPI object.

2. XMLStreamingAPI: Builds the XML payload

Use these methods in the following order to create the XML payload:

1. startDocument(): Creates the top-level parent element in the XML document.

2. Methods to generate child elements in the XML document, such as writeTextElement(), startElement(), and writeArray().

3. Methods to generate attributes for an element, such as writeAttribute(), writeNamespace(), and writeDtd().

4. endElement(): Closes an XML element.

5. endDocument(): Closes the top-level parent element.

6. getXMLString() or getAttachmentId(): Returns the XML string or attachment ID that you created.

7. close(): Closes the XMLStreamingAPI object.

Size limits

Payloads generated through this API cannot exceed these size limits:

- Attachments: 200 MB
- Strings: 5 MB

The following example shows how to create an XML document and store it in the Streaming Attachments [streaming_attachment] table with a defined expiration date.

```java
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.XMLStreamingBuilder()
```
builder.startDocument() // Begins generating the XML document.
.builder.writeHeadVersion()
.builder.writeStartElement("firstName","John") // Writes a "firstName" element and value.
.builder.writeTextElement("lastName","Smith")
.builder.writeTextElement("age","25")
.builder.writeStartElement("address") // Adds an "address" parent element.
.builder.writeTextElement("streetAddress", "21 2nd Street") // Writes a child element and value.
.builder.writeTextElement("city", "Santa Clara")
.builder.writeTextElement("state", "CA")
.builder.writeTextElement("postalCode", "11111")
.builder.endElement() // Adds a closing tag for the "address" element.
.builder.startElement("phoneNumber")
.builder.writeTextElement("type","home")
.builder.writeTextElement("number","212 555-1234")
.builder.writeTextElement("type","fax")
.builder.writeTextElement("number","646 555-4567")
.builder.endElement()
.builder.endDocument() // Stops generating the XML document.

gs.log(builder.getAttachmentId()); // Returns the sys_id of the attachment.
})
} catch (err) {
    gs.log(err);
} finally {
    builder.close();
}
Outputs.payload = builder.getXMLString();

Output:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<firstName>John</firstName>
<lastName>Smith</lastName>
<age>25</age>
<address>
  <streetAddress>21 2nd Street</streetAddress>
  <city>Santa Clara</city>
  <state>CA</state>
  <postalCode>11111</postalCode>
</address>
<phoneNumber>
  <type>home</type>
  <number>212 555-1234</number>
  <type>fax</type>
  <number>646 555-4567</number>
</phoneNumber>
```

**XMLStreamingBuilder - build()**

Returns an XMLStreamingAPI object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The following example shows how to create an XML document and store it in the Streaming Attachments [streaming_attachment] table with a defined expiration date.

```java
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.XMLStreamingBuilder()
        .withAttachment() // Creates the XML document in streaming mode within an attachment.
        .expiresAt(ttl) // Sets an expiration date for the attachment.
        .build(); // Creates the XMLStreamingAPI object.

    builder.startDocument() // Begins generating the XML document.
        .writeTextElement("firstName","John") // Writes a "firstName" element and value.
        .writeTextElement("lastName","Smith")
        .writeTextElement("age","25")
        .startElement("address") // Adds an "address" parent element.
            .writeTextElement("streetAddress","21 2nd Street") // Writes a child element and value.
                .writeTextElement("city","Santa Clara")
                .writeTextElement("state","CA")
                .writeTextElement("postalCode","11111")
            .endElement() // Adds a closing tag for the "address" element.
        .startElement("phoneNumber")
            .writeTextElement("type","home")
            .writeTextElement("number","212 555-1234")
            .writeTextElement("type","fax")
            .writeTextElement("number","646 555-4567")
        .endElement()
    .endDocument() // Stops generating the XML document.

    gs.log(builder.getAttachmentId()); // Returns the sys_id of the attachment.
} catch (err) {
    gs.log(err);
} finally {
    builder.close();
}
```
(XML: Streaming: Builder - expiresAt(Object expiresAt)

Sets a time when the attachment expires. Must also call the `withAttachment()` method.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>expiresAt</td>
<td>GlideDateTime</td>
<td>Object that is set when the attachment expires.</td>
</tr>
</tbody>
</table>

- Minimum value: 7200 seconds, or two hours, from the time the attachment is created. This is the default value if you don’t call the `expiresAt()` method.
- Maximum value: 172800 seconds, or 48 hours, from the time the attachment is created.

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingBuilder</td>
<td>Builder object used to initiate the XML payload.</td>
</tr>
</tbody>
</table>

The following example shows how to create an XML document and store it in the Streaming Attachments [streaming_attachment] table with a defined expiration date.

```java
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.XMLStreamingBuilder()
        .withAttachment() // Creates the XML document in streaming mode within an attachment.
        .expiresAt(ttl) // Sets an expiration date for the attachment.
        .build(); // Creates the XMLStreamingAPI object.

    builder.startDocument() // Begins generating the XML document.
        .writeTextElement("firstName","John") // Writes a "firstName" element and value.
        .writeTextElement("lastName","Smith")
        .writeTextElement("age","25")
        .startElement("address") // Adds an "address" parent element.
        .writeTextElement("streetAddress", "21 2nd Street") // Writes a child element and value.
        .writeTextElement("city", "Santa Clara")
        .writeTextElement("state", "CA")
```
XMLStreamingBuilder - withAttachment()

Creates an XML document as an attachment and stores it in the Streaming Attachments [streaming_attachment] table. If you don’t call this method, the API creates the XML document as a string.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLStreamingBuilder</td>
<td>Builder object used to initiate the XML payload.</td>
</tr>
</tbody>
</table>

The following example shows how to create an XML document and store it in the Streaming Attachments [streaming_attachment] table with a defined expiration date.

```javascript
try {
    var ttl = new GlideDateTime("2011-01-01 12:00:00");
    var builder = new sn_ih.XMLStreamingBuilder()
        .withAttachment() // Creates the XML document in streaming mode within an attachment.
        .expiresAt(ttl) // Sets an expiration date for the attachment.
    gs.log(builder.getAttachmentId()); // Returns the sys_id of the attachment.
} catch (err) {
    gs.log(err);
} finally {
    builder.close();
}
```
This example shows how to build the XML payload and save it as a string.

```javascript
(function execute(inputs, outputs) {

  var builder = new sn_ih.XMLStreamingBuilder().build();

  builder.startDocument();
  .enablePrettyPrint()
  .writeTextElement("firstName","John")
  .writeTextElement("lastName","Smith")
  .writeTextElement("age","25")
  .startElement("address")
  .writeTextElement("streetAddress", "21 2nd Street")
  .writeTextElement("city", "Santa Clara")
  .writeTextElement("state", "CA")
  .writeTextElement("postalCode", "11111")
  .endElement() // Adds a closing tag for the "address" element.
  .startElement("phoneNumber")
  .writeTextElement("type","home")
  .writeTextElement("number", "212 555-1234")
  .writeTextElement("type","fax")
  .writeTextElement("number", "646 555-4567")
  .endElement()
  .endDocument() // Stops generating the XML document.

  gs.log(builder.getAttachmentId()); // Returns the sys_id of the attachment.
} catch (err) {
    gs.log(err);
} finally {
    builder.close();
}
```

Output:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<firstName>John</firstName>
<lastName>Smith</lastName>
<age>25</age>
<address>
  <streetAddress>21 2nd Street</streetAddress>
  <city>Santa Clara</city>
  <state>CA</state>
  <postalCode>11111</postalCode>
</address>
<phoneNumber>
  <type>home</type>
  <number>212 555-1234</number>
  <type>fax</type>
  <number>646 555-4567</number>
</phoneNumber>
```

**XMLStreamingBuilder - XMLStreamingBuilder()**

Instantiates the XMLStreamingBuilder object.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This example shows how to instantiate a `builder` object.

```javascript
var builder = new sn_ih.XMLStreamingBuilder()
```
XMLUtilJS - Global

Provides XML utilities for JavaScript to be used with Discovery scripts.

Use this script include in any server-side discovery script where you need XML utilities.

Access these methods using the static variable XMLUtilJS.

XMLUtilJS - escapeForXMLText(String text)

Provides escape text for a given string.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>The text to format.</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The formatted text.</td>
</tr>
</tbody>
</table>

XMLUtilJS - stringToValue(String str)

Converts a string to an XML value.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>str</td>
<td>String</td>
<td>The string to convert</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The specified string converted to XML.</td>
</tr>
</tbody>
</table>

XMLUtilJS - unescapeForXMLText(String text)

Provides un-escaped text for a given string.
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>String</td>
<td>The text to clean up.</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The cleaned up string.</td>
</tr>
</tbody>
</table>

**XMLUtilJS - valueToString(String XMLvalue)**

Converts an XML value to a string.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLvalue</td>
<td>String</td>
<td>The XML to convert</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The XML value converted to a string.</td>
</tr>
</tbody>
</table>

### Scripts

Use scripts to extend your instance beyond standard configurations. With scripts, you may automate processes, add functionality, integrate your instance with an outside application and more.

APIs (Application Programming Interfaces) provide classes and methods that you can use in scripts to define functionality. ServiceNow provides APIs as JavaScript classes, web services, and other points of connection for integrations. Note that you cannot access commonly used JavaScript objects (such as DOM or Window). Jelly scripts are also used in some modules. Jelly is used to turn XML into HTML and may include both client-side and server-side scripts.

Scripts may be server-side (run on the server or database), client-side (run in the user's browser), or run on the MID server.

⚠️ **Note:** When you are writing scripts, you cannot use reserved words.
Understand JavaScript before you begin customizing your instance, and with Jelly if you intend to deploy Jelly scripts.

Server-side scripts

Perform database operations. For example, use a server-side script to update a record. Create a script in a scoped application or in the global scope. Each execution context includes a set of available APIs.

Scoped environment

Use scoped APIs when scripting in a scoped application.Scoped Glide APIs do not include all the methods included in the global Glide APIs, and you cannot call a global Glide API in a scoped application.

Global environment

The global scope is a special application scope that identifies applications developed prior to application scoping, or applications intended to be accessible to all other global applications. Use global APIs when scripting in the global scope.

To learn more about server-side scripting, see Server-side scripting. To learn more about application scope, see Application scope.

Client-side scripts

Make changes to the appearance of forms, display different fields based on values that are entered, or change other custom display options.

- **onLoad** client scripts run when the form or page is loaded
- **onChange** client scripts run when something specific gets changed AND also when the form or page loads
- **onSubmit** client scripts run when the form is submitted

Client Scripts can also be called by other scripts or modules, including UI policies. To learn more about client-side scripting, see Client-side scripting.

Available script types

Scripts can be used in many places. The most important detail is whether the script runs on the client or the server.
### Script types and where they run

<table>
<thead>
<tr>
<th>Script</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Access Control</strong></td>
<td>Determines whether access will be granted for a specified operation to a specific entity.</td>
</tr>
<tr>
<td></td>
<td>• type of entity being secured</td>
</tr>
<tr>
<td></td>
<td>• operation being secured</td>
</tr>
<tr>
<td></td>
<td>• unique identifier describing the object</td>
</tr>
<tr>
<td></td>
<td>Can be defined by roles, conditional expressions or scripts.</td>
</tr>
<tr>
<td><strong>Ajax Scripts</strong></td>
<td>Enables the client to get data from the server to dynamically incorporate into a page without reloading the whole page.</td>
</tr>
<tr>
<td></td>
<td>• Ajax Client Scripts request that information be returned, or that action be taken, or sometimes both</td>
</tr>
<tr>
<td></td>
<td>• Ajax Server Scripts fulfill Ajax Client Script requests</td>
</tr>
<tr>
<td><strong>Business Rules</strong></td>
<td>Customizes system behavior</td>
</tr>
<tr>
<td></td>
<td>• runs when a database action occurs (query, insert, update or delete)</td>
</tr>
<tr>
<td></td>
<td>• the script may run</td>
</tr>
<tr>
<td></td>
<td>◦ before or after the database action is performed (runs as part of the database operation)</td>
</tr>
<tr>
<td></td>
<td>◦ asynchronously (at some point after the database operation)</td>
</tr>
<tr>
<td></td>
<td>◦ on display (when displaying the data in a form)</td>
</tr>
<tr>
<td><strong>Service Catalog UI policies</strong></td>
<td>Defines the display of a variable set or a catalog item (from the service catalog)</td>
</tr>
</tbody>
</table>
### Script types and where they run (continued)

<table>
<thead>
<tr>
<th>Script</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Client Scripts**| Used for making changes to the appearance of forms, displaying different fields based on values that are entered or other custom display options.  
• onLoad means the Client Script runs when the form or page is loaded  
• onChange means the Client Script runs when something specific gets changed AND also when the form or page loads  
• onSubmit means the Client Script runs when the form is submitted  
Client Scripts can also be called by other scripts or modules, including UI policies. |
| **Script actions**| Contains scripts which run when an event occurs, for example  
• approval is cancelled  
• change is approved  
• problem is assigned  
Can have a condition which must be true for the script to run. Commonly used to call a Script Include. |
| **Script Includes**| Contains scripts which can be functions or classes. These scripts run only when called by other scripts (often Business Rules).  
Any server script which is complicated or reusable should be a Script Include (especially complicated Business Rules). |
| **Transform Maps**| Used for importing data.  
• defines mapping relationships between tables  
• can use Business Rules, other scripts and/or other options to import that data  
Do not always include scripts. |
| **UI Actions**    | Creates the ability to choose a specific action such as clicking a button or a link.  
UI Actions put these items on forms and lists:  
• buttons  
• links |
### Script types and where they run (continued)

<table>
<thead>
<tr>
<th>Script</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• context menu items</td>
</tr>
<tr>
<td></td>
<td>• list choices</td>
</tr>
</tbody>
</table>

**UI Context Menus**

Defines which "right-click menu" will pop-up in which area, and the menu choices that will be available
### Script types and where they run (continued)

<table>
<thead>
<tr>
<th>Script</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>client</strong> - action</td>
<td>Scripts run on the client.</td>
</tr>
<tr>
<td><strong>server</strong> - dynamic</td>
<td>Dynamic action scripts run on the server.</td>
</tr>
<tr>
<td><strong>server</strong> - all</td>
<td>All conditions run on the server.</td>
</tr>
</tbody>
</table>

**Note:** If you use a left-handed mouse configuration, right-click means “click the other button.”

#### UI Macros

Contains modular, reusable components that can contain Jelly and are called by UI pages. They also contain different types of scripts and may be called multiple times on the same page.

**Note:** Jelly turns XML into HTML.
### Script types and where they run (continued)

<table>
<thead>
<tr>
<th>Script</th>
<th>Description</th>
</tr>
</thead>
</table>
| UI Pages | Used to create and display pages, forms, dialogs, lists and other UI components. Can be displayed on a standalone basis, or called as a usable component, as part of a larger page. Can contain:  
- Client Scripts,  
- processing scripts (which are server scripts),  
- HTML,  
- Jelly,  
- UI Macros,  
- and also can call other scripts. |
### Script types and where they run (continued)

<table>
<thead>
<tr>
<th>Script</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong></td>
<td>Jelly turns XML into HTML.</td>
</tr>
<tr>
<td>UI Policies</td>
<td>Defines the behavior and visibility of fields on a form.</td>
</tr>
<tr>
<td></td>
<td>• mandatory</td>
</tr>
<tr>
<td></td>
<td>• visible</td>
</tr>
<tr>
<td></td>
<td>• read only</td>
</tr>
<tr>
<td></td>
<td>Use UI Policies rather than client scripts whenever possible.</td>
</tr>
<tr>
<td></td>
<td>• UI Policies are always attached to one table</td>
</tr>
<tr>
<td></td>
<td>• UI Policies often have a condition which must be true in order for them to run.</td>
</tr>
<tr>
<td>UI Properties</td>
<td>Designates what the instance will look like.</td>
</tr>
<tr>
<td>UI Scripts</td>
<td>Contains client scripts stored for re-use. Only used when called from other scripts.</td>
</tr>
<tr>
<td></td>
<td>Not recommended for use.</td>
</tr>
<tr>
<td>Validation</td>
<td>Validates that values are in a specified format.</td>
</tr>
<tr>
<td>Scripts</td>
<td>For example, a validation script can verify that the only value allowed in a specified field is an integer.</td>
</tr>
<tr>
<td>Workflow editor</td>
<td>Used to create or change a workflow. Scripts can be run at any point in a workflow or different scripts can be run at different points. Scripts also can be found inside every workflow activity and can be modified (although do so with extreme caution).</td>
</tr>
</tbody>
</table>
Glide class overview

The ServiceNow Glide classes expose JavaScript APIs that enable you to conveniently work with tables using scripts.

Using the Glide APIs, you can perform database operations without writing SQL queries, display UI pages, and define UI actions. The following tables provide brief descriptions of the Glide classes and links to detailed information.

Server-side Glide classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord</td>
<td>Use this class for database operations instead of writing SQL queries. GlideRecord is a special Java class that can be used in JavaScript exactly as if it were a native JavaScript class. A GlideRecord is an object that contains records from a single table. See GlideRecord.</td>
</tr>
<tr>
<td>GlideElement</td>
<td>Use this class to operate on the fields of the current GlideRecord. See GlideElement.</td>
</tr>
<tr>
<td>GlideSystem</td>
<td>Use this class to get information about the system. See GlideSystem.</td>
</tr>
<tr>
<td>GlideAggregate</td>
<td>Use this class to perform database aggregation queries, such as COUNT, SUM, MIN, MAX, and AVG, for creating customized reports or calculations in calculated fields. See GlideAggregate.</td>
</tr>
<tr>
<td>GlideDateTime</td>
<td>Use this class to perform date-time operations, such as date-time calculations, formatting a date-time, or converting between date-time formats. See GlideDateTime.</td>
</tr>
</tbody>
</table>

Client-side Glide Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideAjax</td>
<td>Use this class to execute server-side code from the client. See GlideAjax.</td>
</tr>
<tr>
<td>GlideDialogWindow</td>
<td>Use this class to display a dialog window. See GlideDialogWindow.</td>
</tr>
<tr>
<td>GlideForm</td>
<td>Use this class to customize forms. See GlideForm.</td>
</tr>
<tr>
<td>GlideList2</td>
<td>Use this class to customize (v2) lists, including normal lists and related lists. See GlideList2.</td>
</tr>
</tbody>
</table>
Client-side Glide Classes (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideMenu</td>
<td>Use this class to customize UI Context Menu items. See GlideMenu.</td>
</tr>
<tr>
<td>GlideUser</td>
<td>Use this class to get session information about the current user and current user roles. See GlideUser.</td>
</tr>
</tbody>
</table>

Glide stack

Glide is an extensible Web 2.0 development platform written in Java that facilitates rapid development of forms-based workflow applications (work orders, trouble ticketing, and project management, for example).

User interface stack technology map

<table>
<thead>
<tr>
<th>Java packages</th>
<th>Technologies used</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>User Interface (Browser)</td>
</tr>
<tr>
<td>com.glide.ui</td>
<td>• AngularJS</td>
</tr>
<tr>
<td>com.glide.jelly</td>
<td>• HTML</td>
</tr>
<tr>
<td>com.glide.script</td>
<td>• CSS</td>
</tr>
<tr>
<td>com.glide.db</td>
<td>• JavaScript</td>
</tr>
<tr>
<td>GlideServlet</td>
<td>Apache Jelly</td>
</tr>
<tr>
<td>Business Rules</td>
<td>Mozilla Rhino</td>
</tr>
<tr>
<td>Persistence</td>
<td>JDBC</td>
</tr>
</tbody>
</table>

User interface stack technology map descriptions

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideServlet</td>
<td>The primary driver of Glide, and the only servlet in the system, is found in GlideServlet.java. The GlideServlet:</td>
<td>• Handles inbound action requests</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Renders pages</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Merges data with forms</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td>Attributes</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Business Rules</td>
<td></td>
<td>• ECMA / JavaScript implementation based on Mozilla Rhino&lt;br&gt;• Interfaces with persistence layer using JDBC recordset interface&lt;br&gt;• Merges persistence layer meta-data with data for easy correlation</td>
</tr>
<tr>
<td>Persistence</td>
<td></td>
<td>• Persistence means any store&lt;br&gt;◦ RDBMS&lt;br&gt;◦ LDAP&lt;br&gt;◦ File system&lt;br&gt;• Uniform access regardless of store type&lt;br&gt;• Provides QUID and meta-data capabilities&lt;br&gt;• Interfaces presented to callers&lt;br&gt;◦ RecordSet&lt;br&gt;◦ TableDescriptor&lt;br&gt;◦ ElementDescriptor</td>
</tr>
</tbody>
</table>
Execution order of scripts and engines

Scripts, assignment rules, business rules, workflows, escalations, and engines all take effect in relation to a database operation, such as insert or update. In many cases, the order of these events is important.

⚠️ Note: Client-based code that executes in the browser, using Ajax or running as JavaScript, will always execute before the form submission to the server.

The order of execution is as follows:

1. **Before business rules**: Scripts configured to execute before the database operation with an order less than 1000.

2. **Before engines**: The following are not executed in any specific order:
   - Approval engine (for task and sys_approval_approver tables)
   - Assignment rules engine (for task tables)
   - Data policy engine
• Escalation engine
• Field normalization engine
• Role engine - keeps role changes in sync with sys_user_has_role table (for sys_user, sys_user_group, sys_user_grmember, and sys_user_role tables)
• Execution plan engine (for task tables)
• Update version engine - creates version entry when sys_update_xml entry is written (for sys_update_xml table)
• Workflow engine (for default workflows)

3. Before business rules: Scripts configured to execute before the database operation with an order greater than or equal to 1000.

4. The database operation (insert, update, delete).

5. After business rules: Scripts configured to execute after the database operation with an order less than 1000.

6. After engines. The following are not executed in any specific order:
   • Label engine
   • Listener engine
   • Table notifications engine
   • Role engine - keeps role changes in sync with sys_user_has_role table (for sys_user, sys_user_group, sys_user_grmember and sys_user_role tables)
   • Text indexing engine
   • Update sync engine
   • Data lookup engine inserts or updates
   • Workflow engine (for deferred workflows)
   • Trigger engine (for all Flow Designer flows)

7. Email notifications. The following are executed based on the weight of the notification record:
   • Notifications sent on an insert, update, or delete
   • Event-based notifications

8. After business rules. Scripts configured to execute after the database operation with an order greater than or equal to 1000.

**Script evaluation of fields by data type**

Script fields evaluate data based on the field type of the input.
### Evaluation of fields by data types

<table>
<thead>
<tr>
<th>Type</th>
<th>Evaluates to in script</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>The string</td>
<td>&quot;dog&quot; &gt; &quot;dog&quot;</td>
</tr>
<tr>
<td>Decimal</td>
<td>A number with up to two decimal points</td>
<td>12.34 &gt; 12.34</td>
</tr>
<tr>
<td>Integer</td>
<td>A number with zero decimal points</td>
<td>12 &gt; 12</td>
</tr>
<tr>
<td>True / False</td>
<td>true or false</td>
<td>☑️ &gt; true</td>
</tr>
<tr>
<td>Date</td>
<td>A date formatted as yyyy-mm-dd</td>
<td>2008-11-04</td>
</tr>
<tr>
<td>Date-time</td>
<td>A day and time formatted as yyyy-mm-dd hh:mm:ss</td>
<td>2008-11-04 06:46:20</td>
</tr>
</tbody>
</table>
| Duration     | A date that is equal to January 1st 1970 00:00:00 + the amount of time of the duration being stored | Days 00 Hours 00:00:00 > "1970-01-01 00:00:00"  
Days 1 Hours 02:03:04 > "1970-01-02 02:03:04" |
### Evaluation of fields by data types (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Evaluates to in script</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Choice</td>
<td>Returns the contents of the value field for the <code>sys_choice</code> record associated with that choice. See: Choice List for more information on returning the value associated with a particular item in a choice list.</td>
<td><code>Incident state: New</code> &gt; &quot;2&quot; (Note that this value is a string)</td>
</tr>
<tr>
<td>Journal</td>
<td>Returns a string of all entries made to that journal field. See Journal Fields for scripting of journal type fields</td>
<td>The web server is down &gt; The web server is down</td>
</tr>
</tbody>
</table>

**Note:** This date corresponds to the system time zone. If a different user time zone has been specified, the date and time value may appear different for that user.
## Evaluation of fields by data types (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Evaluates to in script</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference</td>
<td>Returns the sys_id of the record that is referenced</td>
<td><img src="http://www.service-now.com" alt="Database" /> &gt; &quot;287ee6fe9fe198100ada7950db1b73&quot;</td>
</tr>
<tr>
<td>Image</td>
<td>Returns the path to the image</td>
<td><img src="http://images/icons/image_name.gif" alt="Image" /> &gt; images/icons/image_name.gif</td>
</tr>
<tr>
<td>URL</td>
<td>Returns a string</td>
<td><img src="http://www.service-now.com" alt="URL" /> &gt; &quot;<a href="http://www.service-now.com">http://www.service-now.com</a>&quot;</td>
</tr>
<tr>
<td>Glide Lists</td>
<td>Returns a string of comma-separated Sys IDs</td>
<td><img src="http://example.com" alt="Watch list" /> &gt; 5137153cc611227c000bd1bd8cd2007d14f049a9fe19810</td>
</tr>
</tbody>
</table>

## Scripting alert, info, and error messages

You can send messages to customers as alerts, informative messages, or error messages.

### Business rule and other general use scripts

<table>
<thead>
<tr>
<th>Script</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>current.field_name.setError(&quot;Hello World&quot;);</code></td>
<td>Will put &quot;Hello World&quot; below the specified field.</td>
</tr>
<tr>
<td><code>gs.addInfoMessage(&quot;Hello World&quot;);</code></td>
<td>Will put &quot;Hello World&quot; on the top of the screen.</td>
</tr>
<tr>
<td><code>gs.print(&quot;Hello World&quot;);</code></td>
<td>Will write to the text log on the file system but not to the sys_log table in the database.</td>
</tr>
<tr>
<td><code>gs.log(&quot;Hello World&quot;);</code></td>
<td>Will write to the database and the log file.</td>
</tr>
</tbody>
</table>

**Note:** Too much of this can adversely affect performance.

### Important: The methods in this table are only for use in client scripts.
Client side scripts

<table>
<thead>
<tr>
<th>Script</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>alert(&quot;Hello World&quot;);</code></td>
<td>Will pop up a window with &quot;Hello World&quot; and an 'OK' button.</td>
</tr>
<tr>
<td><code>confirm(&quot;Hello World&quot;);</code></td>
<td>Will pop up a window with &quot;Hello World?&quot; and a 'Ok' and 'Cancel' buttons.</td>
</tr>
<tr>
<td><code>g_form.showFieldMsg(&quot;field_name&quot;, &quot;Hello World&quot;, &quot;error&quot;);</code></td>
<td>Puts &quot;Hello World&quot; in an error message below the specified field.</td>
</tr>
<tr>
<td><code>g_form.hideFieldMsg(&quot;field_name&quot;);</code></td>
<td>Hides an error message that is visible under the specified field.</td>
</tr>
</tbody>
</table>

It is also possible to add other custom messages to your forms if necessary using client scripting.

The text size of info and error messages at the top of the screen is customizable. Two properties control this. If you configured your forms, you may need to add these properties.

Error and alert text size properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>css.outputmsg.info.text.font-size</code></td>
<td>Sets the size for info messages. Default is 11pt.</td>
</tr>
<tr>
<td><code>css.outputmsg.error.text.font-size</code></td>
<td>Sets the size for error messages. Default is 11pt.</td>
</tr>
</tbody>
</table>

Using regular expressions in scripts

JavaScript regular expressions automatically use an enhanced regex engine, which provides improved performance and supports all behaviors of standard regular expressions as defined by Mozilla JavaScript. The enhanced regex engine supports using Java syntax in regular expressions.

The `SNC.Regex` API is not available for scoped applications. For scoped applications, remove the `SNC.Regex` API and use standard JavaScript regular expressions.

For more information on JavaScript regular expressions, see the Mozilla JavaScript documentation on regular expressions and `RegExp`. 
Using Java syntax in JavaScript regular expressions

The enhanced regex engine includes an additional flag to allow Java syntax to be used in JavaScript regular expressions.

Regular expressions with the additional flag work in all places that expect a regular expression, such as `String.prototype.split` and `String.prototype.replace`. To use Java syntax in a regular expression, use the Java inline flag `j`, for example:

```
/(?ims)ex(am)ple/j
```

### Extended regular expression flags

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>j</td>
<td>Defines a regular expression that executes using the Java regular expression engine. It can be used to access Java-only features of regular expressions (such as look behind, negative look behind) or to use Java regular expressions without translating them into JavaScript regular expressions. For example: <code>var regex = /ex(am)ple/j;</code></td>
</tr>
</tbody>
</table>

Convert SNC Regex expressions to enhanced regex expressions

When you upgrade to Eureka Patch 5 or later releases, you should convert scripts that use the `SNC.Regex` API to use regular JavaScript expressions.

**Procedure**

1. From the original expression, such as: `SNC.Regex("/expr/is")`, create a new regular expression object using the pattern with the slashes stripped.

   **Example**
   ```
   new RegExp('expr');
   ```

2. Move the `SNC.Regex` flags to the start of the expression using Java’s inline flag special construct.

   **Example**
   ```
   new RegExp('(?is)expr');
   ```

3. Add the `j` flag to the `RegExp` to tell the engine to treat the expression as a Java expression.

   **Note:** If you know that the script being converted does not use Java syntax, it is not necessary to use the `j` flag.
Example

```
new RegExp('(?is)expr', 'j');
```

4. Add the `g` flag to handle multiple matches or a global replace.

Example

```
new RegExp('(?is)expr', 'jg');
```

Example:
Using SNC.Regex

```
var r = new SNC.Regex('/world/');
var str = 'helloworld';
var replaced = r.replaceAll(str, 'there');
// replaced == 'hellothere'
```

Using a JavaScript regular expression

```
var r = new RegExp('world', 'jg');
var str = 'helloworld';
var replaced = str.replace(r, 'there');
// replaced == 'hellothere'
```

**JavaScript syntax editor**

The syntax editor provides support for editing JavaScript scripts.

The syntax editor has these features.

- JavaScript syntax coloring, indentation, line numbers, and automatic creation of closing braces and quotes
- JavaScript support
- Linting using the ESLint utility

ℹ️ Note: Modify or view default linting configurations by accessing the `glide.ui.syntax_editor.linter.eslint_config` property in the System Property [sys_properties] table. See Available system properties for more information.

- Context menu for script includes, API, and tables
- Script macros for common code shortcuts

This feature requires the Syntax Editor (com.glide.syntax_editor) plugin.
Script macro maintenance

Administrators can define new script macros or modify existing script macros.

Before you begin
Role required: admin

About this task
Script macros provide shortcuts for typing commonly used code. Several script macros are available by default. Administrators can define new or modify existing script macros.

Procedure
1. Navigate to **System Definition > Syntax Editor Macros**.
2. Click **New** or select the macro to edit.
3. Define the macro details with the fields listed in the table below.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Macro keyword text users type to insert macro text.</td>
</tr>
<tr>
<td>Comments</td>
<td>Description of the macro. This text appears when the user types help.</td>
</tr>
<tr>
<td>Text</td>
<td>Full macro text that replaces the name in the editor.</td>
</tr>
</tbody>
</table>
Syntax editor plugin

Enable the syntax editor plugin to use the syntax editor. The syntax editor enables the following features for all script fields:

- JavaScript syntax coloring, indentation, line numbers, and automatic creation of closing braces and quotes
- Code editing functions
- Code syntax checking
- Script macros for common code shortcuts

JavaScript syntax editor

The syntax editor can be disabled or enabled by modifying the `glide.ui.javascript_editor` property in the `sys_properties.list`. In addition, administrators can configure the syntax editor to show error and warning indicators next to a line of code that contains an error by modifying the `glide.ui.syntax_editor.show_warnings_errors` property. For information on the `sys_properties.list`, refer to Available system properties.

**Note:** Administrators can disable or enable the syntax editor for all users, regardless of user preference.

Searching for errors by line

To locate the exact position of the error in a large script, click the Go to line icon. This feature is particularly useful when you are encounter a syntax error in a log file rather than in the ServiceNow record itself. In this case, you can navigate to the record and search for errors by line number. In the dialog box that appears, enter the line number of an error, and then click OK. Your view moves to the site of the error, and the cursor marks the correct line and column.
Note: For this feature to function, you must disable the Syntax Editor.

Navigate to a line number
When the syntax editor is disabled, users can navigate to a specific line in the code using the Go to line icon (î).

Procedure
1. Click the Go to line icon (î).

   Note: This icon is not available when the editor is enabled.

2. Enter a number in the field and then press Enter.

Syntax editor JavaScript support
The syntax editor provides editing functions to support editing JavaScript scripts.

JavaScript editing functions

<table>
<thead>
<tr>
<th>Icon</th>
<th>Keyboard Shortcut</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>♻</td>
<td>N/A</td>
<td>Toggle Syntax Editor</td>
<td>Disables the syntax editor. Click the button again to enable the syntax editor.</td>
</tr>
<tr>
<td>📝</td>
<td>Access Key + R</td>
<td>Format Code</td>
<td>Applies the proper indentation to the script.</td>
</tr>
<tr>
<td>🕒</td>
<td>Access Key + C</td>
<td>Comment Selected Code</td>
<td>Comments out the selected code.</td>
</tr>
<tr>
<td>Icon</td>
<td>Keyboard Shortcut</td>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>------------------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>🌐</td>
<td>Access Key + U</td>
<td>Uncomment Selected Code</td>
<td>Removes comment codes from the selected code.</td>
</tr>
<tr>
<td>🌐</td>
<td>N/A</td>
<td>Check Syntax</td>
<td>Checks the code for syntax errors. By default, the system automatically checks for syntax errors as you type in a script field. If an error or warning is found, the syntax editor displays a bullet beside the script line containing the error or warning. This check occurs on all script fields.</td>
</tr>
<tr>
<td>🌐</td>
<td>Access Key + \</td>
<td>Start Searching</td>
<td>Highlights all occurrences of a search term in the script field and locates the first occurrence. Click the icon, then enter the search term and press <strong>Enter</strong>. You can use regular expressions enclosed in slashes to define the search term. For example, the term <code>/a{3}/</code> locates <code>aaa</code>.</td>
</tr>
<tr>
<td>🌐</td>
<td>Access Key + [</td>
<td>Find Next</td>
<td>Locates the next occurrence of the current search term in the script field. Use <strong>Start Searching</strong> to change the current search term.</td>
</tr>
<tr>
<td>🌐</td>
<td>Access Key + ]</td>
<td>Find Previous</td>
<td>Locates the previous occurrence of the current search term in the script field. Use <strong>Start Searching</strong> to change the current search term.</td>
</tr>
</tbody>
</table>
| 🌐 | Access Key + W | Replace | Replaces the next occurrence of a text string in the script field.  
1. Click the icon, then enter the string to replace and press **Enter**. You can use regular expressions enclosed in slashes to define the string to replace. For example, the term `/a{3}/` locates `aaa`.  
2. Enter the replacement string and press **Enter**. |
| 🌐 | Access Key + ; | Replace All | Replaces all occurrences of a text string in the script field.  
1. Click the icon, then enter the string to replace and press **Enter**. You can use regular expressions enclosed in slashes to define the string to replace. For example, the term `/a{3}/` locates `aaa`.  
2. Enter the replacement string and press **Enter**. |
<table>
<thead>
<tr>
<th>Icon</th>
<th>Keyboard Shortcut</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>N/A</td>
<td>Save</td>
<td>Saves changes without leaving the current view. Use this button in full screen mode to save without returning to standard form view.</td>
</tr>
<tr>
<td></td>
<td>Access Key + L</td>
<td>Toggle Full Screen Mode</td>
<td>Expands the script field to use the full form view for easier editing. Click the button again to return to standard form view. This feature is not available for Internet Explorer.</td>
</tr>
<tr>
<td></td>
<td>Access Key + P</td>
<td>Help</td>
<td>Displays the keyboard shortcuts help screen.</td>
</tr>
</tbody>
</table>

**JavaScript editing tips**

- To fold a code block, click the minus sign beside the first line of the block. The minus sign only appears beside blocks that can be folded. To unfold the code block, click the plus sign.
- To insert a fixed space anywhere in your code, press Tab.
- To indent a single line of code, click in the leading white space of the line and then press Tab.
- To indent one or more lines of code, select the code and then press Tab. To decrease the indentation, press Shift + Tab.
- To remove one tab from the start of a line of code, click in the line and press Shift + Tab.

**JavaScript resources**

Scripts use ECMA 262 standard JavaScript. Helpful resources include:

- Mozilla: [http://developer.mozilla.org/en/docs/Core_JavaScript_1.5_Reference](http://developer.mozilla.org/en/docs/Core_JavaScript_1.5_Reference)
- History and overview: [http://javascript.crockford.com/survey.html](http://javascript.crockford.com/survey.html)

**Context menu**

Enable the context menu for script includes, Glide APIs, and tables in the JavaScript editor.
With the context menu options, your users can navigate to:

- Script include definitions
- Glide API documentation
- System and custom table definitions and data

In the syntax editor, bold font is used for tokens that have a context menu. Right-click the token to view context menu options. If you use a Mac, you can use the Command-click shortcut.

### Context menu options

<table>
<thead>
<tr>
<th>Token type</th>
<th>Context menu option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Script include</td>
<td>Open Definition</td>
<td>Definition of the script include in a new window.</td>
</tr>
<tr>
<td></td>
<td>Find References</td>
<td>List of files referencing the script include.</td>
</tr>
<tr>
<td>Glide API</td>
<td>Show Documentation</td>
<td>Documentation page of the Glide API.</td>
</tr>
<tr>
<td>Table</td>
<td>Show Definition</td>
<td>Definition of the system or custom table in a new window.</td>
</tr>
</tbody>
</table>

**Note:**
- To view the preview of the file, click the preview script icon 🔄.
  - To open the file in a new window, click **Open File**.
- To view all files that reference the script include, click **Show All Files**.
Context menu options (continued)

<table>
<thead>
<tr>
<th>Token type</th>
<th>Context menu option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Show Data</td>
<td>Records in the table that are based on the role of the current user.</td>
</tr>
</tbody>
</table>

Enable or disable the context menu in the script editor using the `glide.ui.syntax_editor.context_menu` property in the System Property [sys_properties] table. See Available system properties for more information.

⚠️ Note: Context menu options can be accessed only if the browser supports SharedWorker. For example, Google Chrome and Mozilla Firefox.

Syntax editor macros

Script macros provide shortcuts for typing commonly used code. To insert macro text into a script field, enter the macro keyword followed by the Tab.

**vargr**

- **Description**: Inserts a standard `GlideRecord` query for a single value.
- **Output**:

```javascript
var now_GR = new GlideRecord('');
gr.addQuery("name", "value");
gr.query();
if (gr.next()) {
}
```

**vargror**

- **Description**: Inserts a `GlideRecord` query for two values with an OR condition.
- **Output**:

```javascript
var now_GR = new GlideRecord('');

var qc = gr.addQuery('field', 'value1');
qc.addOrCondition('field', 'value2');
```
gr.query();
while (gr.next()) {
}

for

- **Description**: Inserts a standard recursive loop with an array.
- **Output**:

```javascript
for (var i=0; i< myArray.length; i++) {
    //myArray[i];
}
```

info

- **Description**: Inserts a GlideSystem information message.
- **Output**:

```javascript
gs.addInfoMessage('"');
```

method

- **Description**: Inserts a blank JavaScript function template.
- **Output**:

```javascript
/*_________________________________________________________________
 * Description:
 * Parameters:
 * Returns:
 *________________________________________________________________*/
 : function() {
 },
```

doc

- **Description**: Inserts a comment block for describing a function or parameters.
- **Output**:
Script syntax error checking

All script fields provide controls for checking the syntax for errors and for locating the error easily when one occurs. The script editor places the cursor at the site of a syntax error and lets you search for errors in scripts by line number.

The script editor notifies you of syntax errors in your scripts in the following situations.

- Save a new record or update an existing record. A banner appears at the bottom of the editor showing the location of the first error (line number and column number), and the cursor appears at the site of the error. Warnings presented at Save or Update show only one error at a time.
Click the syntax checking icon before saving or updating a record. A banner appears at the bottom of the editor showing the location of all errors in the script, and the cursor appears at the site of the first error.

**Script syntax error**

```
Error:
Problem at line 23 character 15: Missing `}` before `continue`.
continue;
Problem at line 25 character 52: Expected `)` and instead saw `:`
item.setAttribute(name, gr.getValue(name));
Problem at line 25 character 53: Missing `:`
item.setAttribute(name, gr.getValue(name));
```

**Syntax editor keyboard shortcuts and actions**

The syntax editor offers keyboard shortcuts and actions to assist in writing code.

<table>
<thead>
<tr>
<th>Keyboard shortcut or action</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write code</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scripting assistance</td>
<td>Displays a list of valid elements at the insertion point such as:</td>
<td></td>
</tr>
<tr>
<td>Control+Spacebar</td>
<td>• Class names</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Function names</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Object names</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Variable names</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Double-click an entry to add it to the script.</td>
<td></td>
</tr>
<tr>
<td>Enter a period character after a valid</td>
<td>Displays a list methods for the class.</td>
<td></td>
</tr>
<tr>
<td>class name</td>
<td>Double-click an entry to add it to the script.</td>
<td></td>
</tr>
<tr>
<td>Enter an open parenthesis character</td>
<td>Displays the expected parameters for the class or method.</td>
<td></td>
</tr>
<tr>
<td>after a valid class</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Keyboard shortcut or action</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------</td>
<td>---------</td>
</tr>
<tr>
<td>function, or method name.</td>
<td>Enter the expected parameters as needed.</td>
<td></td>
</tr>
<tr>
<td>Toggle full screen mode</td>
<td>Switches between displaying the form with the full screen and displaying it normally.</td>
<td></td>
</tr>
<tr>
<td>Control+M</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Format code</td>
<td>Formats the selected lines to improve readability.</td>
<td></td>
</tr>
</tbody>
</table>
| • **Windows**: Control+Shift+B  
  • **Mac**: Command+Shift+B |             |         |
| Toggle comment             | Adds or removes the comment characters // from the selected lines. | |
| • **Windows**: Control+/  
  • **Mac**: Command+/ |             |         |
| Insert macro text          | Inserts macro text at the current position. | |
  1. In the **Script** field, type the macro keyword text. For example `help`.  
  2. Press Tab. |
| Search                     |             |         |
| Start search               | Highlights all occurrences of a search term in the script field and locates the first occurrence. | |
| • **Windows**: Control+F  
  • **Mac**: Command+F | You can create regular expressions by enclosing the search terms between slash characters / . For example, the search term `/a(3)/` locates the string `aaa`. | |
| Find next                  | Locates the next occurrence of the current search term in the script field. Use **Start Searching** to change the current search term. | |
| • **Windows**: Control+G  
  • **Mac**: Command+G |             |         |
| Find previous              | Locates the previous occurrence of the current search term in the script field. Use | |
### Syntax editor keyboard shortcuts and actions for writing code (continued)

<table>
<thead>
<tr>
<th>Keyboard shortcut or action</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| • **Windows**: Control + Shift + G  
  • **Mac**: Command + Shift + G | **Start Searching** to change the current search term. |        |

Replace

| • **Windows**: Control+E  
  • **Mac**: Command+E | | |
|-----------------------|----------------------------------------------------------------|---------|
| Replace all
  • **Windows**: Control++;  
  • **Mac**: Command++; | Replaces all occurrences of a text string in the script field. |        |

Help

| • **Windows**: Control+H  
  • **Mac**: Command+H | | |
|-----------------------|----------------------------------------------------------------|---------|
| Show description
  • **Windows**: Control+J  
  • **Mac**: Command+J | Displays API documentation for the scripting element at the cursor’s current location. |        |

**Show macros**

1. In the **Script** field, type `help.`
2. Press Tab.

Displays the list of available syntax editor macros as text within the script field.

---

**HTML syntax editor**

The HTML syntax editor provides support for editing HTML and Jelly scripts and defines what's rendered when the page is displayed. The HTML syntax editor can contain either static XHTML or dynamically generated content defined as Jelly, and can call script includes and UI Macros.

The syntax editor has these features.
• HTML and Jelly script support
• HTML and Jelly syntax coloring, indentation, line numbers, and automatic creation of closing braces and quotes
• Auto-suggestions for HTML and Jelly tags

⚠️ Note: The keyboard shortcut is Ctrl+Space.
• Script macros for common code shortcuts

### HTML syntax editor

```xml
<%=jelly trim="true" xmlns=":jellycore" xmlns=":glide" xmlns:i=":null" xmlns:m=":null">
  <input type="hidden" name="selection_result" id="selection_result" value=""/>
  <input type="hidden" name="my_sys_id" id="my_sys_id" value="${cmp.getProperty('sys_id')}"/>
  <div style="text-align:center;">
    <div>${cmp.getMessage('Are you sure you want to close this interaction?')}</div>
  </div>

<%=/*
  comments the selected code.
*/%>

<%=//
  replaces the next occurrence of a text string in the script field.
  1. Click the Replace icon ( ), then enter the string to replace, and press Enter. You can use regular expressions enclosed in slashes to define the string to replace. For example, the term /a{3}/ locates aaa.
  2. Enter the replacement string and press Enter.

<%=/jelly%>
```

### HTML and Jelly editing functions

<table>
<thead>
<tr>
<th>Icon</th>
<th>Keyboard shortcut</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🚄</td>
<td>N/A</td>
<td>Toggle syntax editor</td>
<td>Disables the syntax editor. Click the Toggle syntax editor icon ( ) again to enable the syntax editor.</td>
</tr>
<tr>
<td>🚄</td>
<td>Cmd +/</td>
<td>Toggle comment</td>
<td>Comments the selected code.</td>
</tr>
</tbody>
</table>
| 🚄   | Cmd +E            | Replace               | Replaces the next occurrence of a text string in the script field.  
  1. Click the Replace icon ( ), then enter the string to replace, and press Enter. You can use regular expressions enclosed in slashes to define the string to replace. For example, the term /a{3}/ locates aaa.  
  2. Enter the replacement string and press Enter. |
| 🚄   | Cmd               | Replace All           | Replaces all occurrences of a text string in the script field. |
### HTML and Jelly editing functions (continued)

<table>
<thead>
<tr>
<th>Icon</th>
<th>Keyboard shortcut</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="icon" /></td>
<td><strong>Cmd + F</strong></td>
<td>Start Searching</td>
<td>Highlights all occurrences of a search term in the script field and locates the first occurrence. Click the <strong>Start searching icon</strong> (🔍), then enter the search term and press Enter.</td>
</tr>
<tr>
<td><img src="image" alt="icon" /></td>
<td><strong>Cmd + G</strong></td>
<td>Find Next</td>
<td>Locates the next occurrence of the current search term in the script field. Click the <strong>Start searching icon</strong> (🔍) to change the current search term.</td>
</tr>
<tr>
<td><img src="image" alt="icon" /></td>
<td><strong>Cmd + Shift + G</strong></td>
<td>Find Previous</td>
<td>Locates the previous occurrence of the current search term in the script field. Click the <strong>Start searching icon</strong> (🔍) to change the current search term.</td>
</tr>
<tr>
<td><img src="image" alt="icon" /></td>
<td><strong>Ctrl + M</strong></td>
<td>Toggle Full Screen</td>
<td>Expands the script field to use the full form view for easier editing. Click the <strong>Toggle full screen icon</strong> (🗖) again to return to standard form view. This feature is not available for Internet Explorer.</td>
</tr>
<tr>
<td><img src="image" alt="icon" /></td>
<td><strong>Cmd + H</strong></td>
<td>Help</td>
<td>Displays the keyboard shortcuts help screen.</td>
</tr>
<tr>
<td><img src="image" alt="icon" /></td>
<td><strong>N/A</strong></td>
<td>Save</td>
<td>Saves changes without leaving the current view. Click the <strong>Save icon</strong> (💾) in full screen mode to save without returning to standard form view.</td>
</tr>
</tbody>
</table>

### Editing tips

- To insert a fixed space anywhere in your code, press Tab.
- To indent a single line of code, click in the leading white space of the line and then press Tab.
• To indent one or more lines of code, select the code and then press Tab. To decrease the indentation, press Shift+Tab.

• To remove one tab from the start of a line of code, click in the line and press Shift+Tab.

**Code editor**

The code editor provides support to use programming language services in a text editor and is used in scripts.

The code editor has these features for the supported language services and scripts:

• Syntax coloring, indentation, line numbers, and automatic creation of closing braces and quotes

• Auto-suggestions and auto-completions

**Editing tips**

• To insert a fixed space anywhere in your code, press Tab.

• To indent a single line of code, click in the leading white space of the line and then press Tab.

• To indent one or more lines of code, select the code and then press Tab. To decrease the indentation, press Shift+Tab.

• To remove one tab from the start of a line of code, click in the line and press Shift+Tab.

• To declare variables, use the `var` keyword so that they remain within the proper JavaScript scope.

**Server-side scripting**

Server scripts run on the server or database. They can change the appearance or behavior of ServiceNow or run as business rules when records and tables are accessed or modified.
Server-side Glide APIs (Application Programming Interfaces) provide classes and methods that you can use in scripts to perform server-side tasks.

**Immediately invoked function expressions**

The system uses immediately invoked function expressions when a script runs in a single context, such as in a transform map script. Functions that run from multiple contexts use Script includes instead.

By enclosing a script in an immediately invoked function expression, you can:

- Ensure that the script does not impact other areas of the product, such as by overwriting global variables.
- Pass useful variables or objects as parameters.
- Identify function names in stack traces.
- Eliminate having to make separate function calls.

An immediately invoked function expression follows this format:

```javascript
(function functionName(parameter){
    //The script you want to run
})('value');//Note the parenthesis indicating this function should run.
```

You can declare functions within the immediately invoked function expression. These inner functions are accessible only from within the immediately invoked function expression.

```javascript
(function functionName(parameter){
    function helperFunction(parameter){//return some value}
    var value = helperFunction(parameter);//Valid function call.
    //perform any other script actions
})('value');

var value2 = helperFunction(parameter);//Invalid. This function is not accessible from outside the self-executing function.
```

**Glide Server APIs**

ServiceNow provides APIs for the Glide Server.
GlideAggregate

The GlideAggregate class is an extension of GlideRecord and allows database aggregation (COUNT, SUM, MIN, MAX, AVG) queries to be done. This can be helpful in creating customized reports or in calculations for calculated fields.

ℹ️ Note: This functionality requires a knowledge of JavaScript.

For additional information, refer to GlideAggregate API.

GlideAggregate examples

GlideAggregate is an extension of GlideRecord and its use is probably best shown through a series of examples.

ℹ️ Note: This functionality requires a knowledge of JavaScript.

Here is an example that simply gets a count of the number of records in a table:

```javascript
var count = new GlideAggregate('incident');
count.addAggregate('COUNT');
count.query();
var incidents = 0;
if(count.next())
    incidents = count.getAggregate('COUNT');
```

There is no query associated with the preceding example. If you want to get a count of the incidents that were open, simply add a query as is done with GlideRecord. Here is an example to get a count of the number of active incidents.

```javascript
var count = new GlideAggregate('incident');
count.addQuery('active','true');
count.addAggregate('COUNT');
count.query();
var incidents = 0;
if(count.next())
    incidents = count.getAggregate('COUNT');
```

To get a count of all the open incidents by category the code is:

```javascript
var count = new GlideAggregate('incident');
count.addQuery('active','true');
count.addAggregate('COUNT','category');
count.query();
while(count.next())
    var category = count.category;
```
```javascript
var categoryCount = count.getAggregate('COUNT','category');
gs.log("The are currently "+ categoryCount +" incidents with a category of "+ category);
```

The output is:

```javascript
*** Script: The are currently 1.0 incidents with a category of Data
*** Script: The are currently 11.0 incidents with a category of Enhancement
*** Script: The are currently 1.0 incidents with a category of Implementation
*** Script: The are currently 197.0 incidents with a category of inquiry
*** Script: The are currently 13.0 incidents with a category of Issue
*** Script: The are currently 47.0 incidents with a category of request
```

The following is an example that uses multiple aggregations to see how many times records have been modified using the MIN, MAX, and AVG values.

```javascript
var count = new GlideAggregate('incident');
count.addAggregate('MIN','sys_mod_count');
count.addAggregate('MAX','sys_mod_count');
count.addAggregate('AVG','sys_mod_count');
count.groupBy('category');
count.query();
while(count.next()){
  var min = count.getAggregate('MIN','sys_mod_count');
  var max = count.getAggregate('MAX','sys_mod_count');
  var avg = count.getAggregate('AVG','sys_mod_count');
  var category = count.category.getDisplayValue();
  gs.log(category +" Update counts: MIN = "+ min +" MAX = "+ max +" AVG = "+ avg);
}
```

The output is:

```javascript
*** Script: Data Import Update counts: MIN = 4.0 MAX = 21.0 AVG = 9.3333
*** Script: Enhancement Update counts: MIN = 1.0 MAX = 44.0 AVG = 9.6711
*** Script: Implementation Update counts: MIN = 4.0 MAX = 8.0 AVG = 6.0
*** Script: inquiry Update counts: MIN = 0.0 MAX = 60.0 AVG = 5.9715
*** Script: Inquiry / Help Update counts: MIN = 1.0 MAX = 3.0 AVG = 2.0
*** Script: Issue Update counts: MIN = 0.0 MAX = 63.0 AVG = 14.9459
*** Script: Monitor Update counts: MIN = 0.0 MAX = 63.0 AVG = 3.6561
*** Script: request Update counts: MIN = 0.0 MAX = 53.0 AVG = 5.0987
```

The following is a more complex example that shows how to compare activity from one month to the next.

```javascript
var agg = new GlideAggregate('incident');
agg.addAggregate('count','category');
agg.orderByAggregate('count','category');
agg.orderBy('category');
```
agg.addQuery('opened_at','>=','javascript:gs.monthsAgoStart(2)');
agg.addQuery('opened_at','<=','javascript:gs.monthsAgoEnd(2)');
agg.query();
while(agg.next()){  
    var category = agg.category;
    var count = agg.getAggregate('count','category');
    var query = agg.getQuery();
    var agg2 = new GlideAggregate('incident');
    agg2.addAggregate('count','category');
    agg2.orderByAggregate('count','category');
    agg2.orderBy('category');
    agg2.addQuery('opened_at','>=','javascript:gs.monthsAgoStart(3)');
    agg2.addQuery('opened_at','<=','javascript:gs.monthsAgoEnd(3)');
    agg2.addEncodedQuery(query);
    agg2.query();
    var last ="";
    while(agg2.next()){  
        last = agg2.getAggregate('count','category');
    }
    gs.log(category +": Last month:"+ count +" Previous Month:"+ last);
}

The output is:

*** Script: Monitor: Last month:6866.0 Previous Month:4468.0
*** Script: inquiry: Last month:142.0 Previous Month:177.0
*** Script: request: Last month:105.0 Previous Month:26.0
*** Script: Issue: Last month:8.0 Previous Month:7.0
*** Script: Enhancement: Last month:5.0 Previous Month:5.0
*** Script: Implementation: Last month:1.0 Previous Month:0

The following is an example to obtain distinct count of a field on a group query.

var agg = new GlideAggregate('incident');
agg.addAggregate('count');
agg.addAggregate('count(distinct','category');
agg.addQuery('opened_at', '>=', 'javascript:gs.monthsAgoStart(2)');
agg.addQuery('opened_at', '<=', 'javascript:gs.monthsAgoEnd(2)');
//
agg.groupBy('priority');
agg.query();
while (agg.next()) {  
    // Expected count of incidents and count of categories within each priority value (group)
    gs.info('Incidents in priority ' + agg.priority + ' - ' + agg.getAggregate('count') +

You can implement the SUM aggregate with or without the use of the `groupBy()` method. If you do not use the `groupBy()` method, the result of the SUM is the cumulative value for each different value of the field for which you request the SUM. For example, if you SUM the `total_cost` field in the Fixed Asset table, and the Fixed Asset table contains 12 total records:

- Three records with a `total_cost` of $12
- Four records with a `total_cost` of $10
- Five records with a `total_cost` of $5

When you SUM the record set, the `getAggregate()` method returns three different sums: $36, $40, and $25.

The following code illustrates implementing the SUM aggregate without using the `groupBy()` method:

```javascript
var totalCostSum = new GlideAggregate('fixed_asset');
totalCostSum.addAggregate('SUM', 'total_cost');
totalCostSum.query();

while (totalCostSum.next()) {
    var allTotalCost = 0;
    allTotalCost = totalCostSum.getAggregate('SUM', 'total_cost');
    aTotalCost = totalCostSum.getValue('total_cost');
    gs.print('Unique field value: ' + aTotalCost + ', SUM = ' + allTotalCost + ', ' + allTotalCost/aTotalCost + ' records');
}
```

The output for this example is:

- **Script**: Unique field value: 12, SUM = 36, 3 records
- **Script**: Unique field value: 10, SUM = 40, 4 records
- **Script**: Unique field value: 5, SUM = 25, 5 records

Using the same data points as the prior example, if you use the `groupBy()` method, the SUM aggregate returns the sum of all values for the specified field.
The following example illustrates implementing the SUM aggregate using the `groupBy()` method:

```javascript
var totalCostSum = new GlideAggregate('fixed_asset');
totalCostSum.addAggregate('SUM', 'total_cost');
totalCostSum.groupBy('total_cost');
totalCostSum.query();
if(totalCostSum.next()){ // in case there is no result
    var allTotalCost = 0;
    allTotalCost = totalCostSum.getAggregate('SUM', 'total_cost');
    gs.print('SUM of total_cost: = ' + allTotalCost);
}
```

The output for this example is:

```plaintext
*** Script: SUM of total_cost: 101
```

**GlideRecord**

`GlideRecord` is a special Java class (`GlideRecord.java`) that can be used in JavaScript exactly as if it was a native JavaScript class.

**GlideRecord:**
- is used for database operations instead of writing SQL queries.
- is an object that contains zero or more records from one table. Another way to say this is that a GlideRecord is an ordered list.

A GlideRecord contains both records (rows) and fields (columns). The field names are the same as the underlying database column names. For additional information, refer to [GlideRecord - Scoped](#).

ℹ️ **Note:** Use of `gs.sql()` scripting syntax was discontinued in Geneva. Use standard GlideRecord syntax in its place.

**Using GlideRecordSecure**

`GlideRecordSecure` is a class inherited from `GlideRecord` that performs the same functions as `GlideRecord`, and also enforces ACLs.

**Non-writable fields**

Be aware that, when using `GlideRecordSecure`, non-writable fields are set to NULL when trying to write to the database. By default, `canCreate()` on the column is replaced with `canWrite()` on the column. If that returns false, the column value is set to NULL.
Checking for NULL values
If an element cannot be read because an ACL restricts access, a NULL value is created in memory for that record. With GlideRecord, you must explicitly check for any ACLs that might restrict read access to the record. To do so, an if statement such as the following is required to check if the record can be read:

if ( !grs.canRead() ) continue;

With GlideRecordSecure, you do not need to explicitly check for read access using canRead(). Instead, you can use next() by itself to move to the next record. The following example provides a comparison between GlideRecord and GlideRecordSecure.

```javascript
var count  = 0;
var now_GR  = new GlideRecord('mytable');
now_GR. query();
while (now_GR. next()) {
    if (!now_GR. canRead()) continue;
    if (!now_GR. canWrite()) continue;
    if (!now_GR. val. canRead() || !now_GR. val. canWrite())
        now_GR. val = null;
    else
        now_GR. val = "val-" + now_GR. id;
    if (now_GR. update())
        count ++;
}

var count  = 0;
var grs  = new GlideRecordSecure('mytable');
grs. query();
while (grs. next()) {
    grs. val = "val-" + grs. id;
    if (grs. update())
        count ++;
}
```

Examples
These are two simple examples using GlideRecordSecure.

```javascript
var att  = new GlideRecordSecure ('sys_attachment');
att. get('$[sys_attachment.sys_id]');
var sm  = GlideSecurityManager.get();
var checkMe  = 'record/sys_attachment/delete';
var canDelete  = sm.hasRightsTo(checkMe,att);
```
gs.log('canDelete: ' + canDelete);
canDelete;

var grs = new GlideRecordSecure('task_ci');
grs.addQuery();
grs.query();
var count = grs.getRowCount();
if (count > 0) {
    var allocation = parseInt(10000/count) / 100;
    while (grs.next()) {
        grs.u_allocation = allocation;
        grs.update();
    }
}

GlideSystem

The GlideSystem API provides methods for retrieving information.
The GlideSystem (referred to by the variable name 'gs' in business rules) provides a number of convenient methods to get information about the system, the current logged in user, etc. For example, the method addInfoMessage() permits communication with the user.

    gs.addInfoMessage('Email address added for notification');

Many of the GlideSystem methods facilitate the easy inclusion of dates in query ranges and are most often used in filters and reporting.

For additional information, see GlideSystem.

GlideDateTime

The GlideDateTime class provides methods for performing operations on GlideDateTime objects, such as instantiating GlideDateTime objects or working with glide_date_time fields.

In addition to the instantiation methods described below, a GlideDateTime object can be instantiated from a glide_date_time field using the getGlideObject() method (for example, var gdt =
    gr.my_datetime_field.getGlideObject();).

Some methods use the Java Virtual Machine time zone when retrieving or modifying a date and time value. Using these methods may result in unexpected behavior. Use equivalent local time and UTC methods whenever possible.

For additional information, refer to GlideDateTime.
GlideDate and GlideDateTime examples

The GlideDate and GlideDateTime APIs are used to manipulate date and time values.

Note: This functionality requires a knowledge of JavaScript.

For additional information, refer to GlideDate API and GlideDateTime API.

You can create a GlideDateTime object from a GlideDate object by passing in the GlideDate object as a parameter to the GlideDateTime constructor. By default, the GlideDateTime object is expressed in the internal format, yyyy-MM-dd HH:mm:ss and the system time zone UTC.

```javascript
var gDate = new GlideDate();
gDate.setValue('2015-01-01');
gs.info(gDate);

var gDT = new GlideDateTime(gDate);
gs.info(gDT);
```

Output:

```
2015-01-01
2015-01-01 00:00:00
```

Set a duration field value in script

Examples of JavaScript that can be used to set the value of a duration field.

Note: Negative duration values are not supported.

Using the GlideDateTime.subtract() method

The `subtract(GlideDateTime start, GlideDateTime end)` method in GlideDateTime enables you to set the duration value using a given start date/time and end date/time. An example on how to set the duration for the time a task was opened is:

```javascript
var duration = GlideDateTime.subtract(start, end);
```

If you want to work with the value returned as a number to use in date or duration arithmetic, convert the return to milliseconds:

```javascript
var time = GlideDateTime.subtract(start,end).getNumericValue();
```

If you want to set a duration to the amount of time between some event and the current date/time:
The time values presented to `GlideDateTime.subtract` are expected to be in the user's time zone and in the user's format.

### Setting a default value of a duration field

Setting the default value for a duration field is similar to the method used in the previous topic.

### Setting the duration field value in a client script

This script sets a `duration_field` value in a client script. Replace `duration_field` with the field name from your instance.

```javascript
g_form.setValue('<duration_field>','11 01:02:03');
```

### Calculating and setting a duration using a client script

Here is an example of how to return a value and populate it using a client script.

Create an `onChange` client script that includes the following code. You can modify this script if you need the calculation to happen in an `onLoad` script or some other way.

```javascript
function onChange(control, oldValue, newValue, isLoading){
    var strt = g_form.getValue('<start_field>');
    var end = g_form.getValue('<end_field>');
    var ajax = new GlideAjax('AjaxDurCalc');
    ajax.addParam('sysparm_name','durCalc');
    ajax.addParam('sysparm_strt',strt);
    ajax.addParam('sysparm_end',end);
    ajax/XMLWait();
    var answer = ajax.getAnswer();
    g_form.setValue('<duration_field>', answer);
}
```

Create a system script include file called `AjaxDurCalc` that handles the request. It may be reused for other functions as well.

```javascript
var AjaxDurCalc = Class.create();
AjaxDurCalc.prototype = Object.extendsObject(AbstractAjaxProcessor,{
    durCalc:function(){return
        GlideDuration.subtract(this.getParameter('sysparm_strt'),this.getParameter('sysparm_end'));
    };
});
```

### Changing the duration field value

If you manipulate a duration value with addition/subtraction of some amount of time, use the functions that allow you to get and set the numeric value of ...
the duration. A unit of measure for a duration numeric value is milliseconds. The following is an example that adds 11 seconds to the duration field in the current record.

```javascript
var timems = current.duration.dateNumericValue();
timems = timems + 11*1000;
current.duration.setDateNumericValue(timems);
```

**Formatting the Resolve Time**

To format the **Resolve Time** or the **Business Resolve Time** fields as durations, which displays them as a duration instead of a large integer, add the following attribute to those fields:

```
format=glide_duration
```

Modify the dictionary entry for the field and add the attribute. If there is an existing attribute, separate multiple attributes with commas.

**Setting the maximum unit of measurement**

The `max_unit` dictionary attribute defines the maximum unit of time used in a duration. For example, if `max_unit=minutes`, a duration of 3 hours 5 minutes 15 seconds appears as 185 minutes 15 seconds. To set the maximum unit of duration measurement, add the following dictionary attribute to the `duration` field:

```
max_unit=<unit>
```

**Business rules**

A business rule is a server-side script that runs when a record is displayed, inserted, updated, or deleted, or when a table is queried.

Use business rules to accomplish tasks like automatically changing values in form fields when certain conditions are met, or to create events for email notifications and script actions.

**Note:** Business rules can make use of scripts to take actions on records in the database. However, there are several other scripting options available on the platform, such as client scripts and UI actions.

**How business rules work**

To configure business rules, you first need to determine when the business rule should run and what action it should take.
When business rules run

Business rules run based on two sets of criteria:

- The time that the business rule is configured to run relative to a record being modified or accessed.
- The database operation that the system takes on the record.

The following options are provided to determine the time the business rule should run:

<table>
<thead>
<tr>
<th>Time the business rule should run</th>
<th>Option</th>
<th>When the rule runs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before</td>
<td>Before</td>
<td>After the user submits the form but before any action is taken on the record in the database.</td>
</tr>
<tr>
<td>After</td>
<td>After</td>
<td>After the user submits the form and after any action is taken on the record in the database.</td>
</tr>
<tr>
<td>Async</td>
<td>Async</td>
<td>When the scheduler runs the scheduled job created from the business rule. The system creates a scheduled job from the business rule after the user submits the form and after any action is taken on the record in the database.</td>
</tr>
</tbody>
</table>

**Note:** Newly created business rules will run during upgrades.

<table>
<thead>
<tr>
<th>Database operation that the system takes on the record</th>
<th>Option</th>
<th>When the rule runs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insert</td>
<td>Insert</td>
<td>When the user creates a new record and the system inserts it into the database.</td>
</tr>
</tbody>
</table>

**Note:** Asynchronous business rules do not have access to the previous version of a record. Therefore, the changes(), changesTo(), and changesFrom() GlideElement methods do not work with async rule script. However, the condition builder and condition field (advanced view) both support the changes(), changesTo(), and changesFrom() methods.
### Database operation that the system takes on the record (continued)

<table>
<thead>
<tr>
<th>Option</th>
<th>When the rule runs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update</td>
<td>When the user modifies an existing record.</td>
</tr>
<tr>
<td>Query</td>
<td>Before a query for a record or list of records is sent to the database. Typically you should use query for before business rules.</td>
</tr>
<tr>
<td>Delete</td>
<td>When the user deletes a record.</td>
</tr>
</tbody>
</table>

This image shows when different types of business rules run:

#### Business rule processing flow

- **User or system query**
- **Query rules**
- **Display Rules**
- **Before Rules**
- **After Rules**
- **Database query**
- **Database update**

#### Note:
Business rules apply consistently to records regardless of whether they are accessed through forms, lists, or web services. This is one major difference between business rules and client scripts, which apply only when the form is edited.

#### Business rule actions
Business rules can perform a variety of actions. Common types of actions are:

- Changing field values on a form that the user is updating. Field values can be set to specific values available for that field, values copied from other fields, and relative values determined by the user’s role.
- Displaying information messages to the user.
• Changing values of child tasks based on changes to parent tasks.
• Preventing users from accessing or modifying certain fields on a form.
• Aborting the current database transaction. For example, if certain conditions are met, prevent the user from saving the record in the database.

Administrators can set field values, create information messages, and abort transactions without writing a script.

**Prevent recursive business rules**

Avoid using `current.update()` in a business rule script. The `update()` method triggers business rules to run on the same table for insert and update operations, leading to a business rule calling itself over and over. Changes made in before business rules are automatically saved when all before business rules are complete, and after business rules are best used for updating related, not current, objects. When a recursive business rule is detected, the system stops it and logs the error in the system log. However, `current.update()` causes system performance issues and is never necessary.

You can prevent recursive business rules by using the `setWorkflow()` method with the false parameter. The combination of the `update()` and `setWorkflow()` methods is only recommended in special circumstances where the normal before and after guidelines mentioned above do not meet your requirements.

**Related information**

Precedence between data lookup, assignment, and business rules

Business rules in scoped applications

Every business rule is assigned to either a private application scope or to the global scope.

The types of business rules you can create and how you can access those rules varies depending on the scope of the business rule and the scope of the table it runs on.

**Note:** The term *global* can refer to two different aspects of a business rule: the table it runs on and the scope it runs in. Business rules can either run on specific tables or be global. In addition, they can be in the global scope or in a private application scope.
Business rules on specific tables

Most business rules run on a specific table, which is defined in the Table field. You can create business rules on tables in the same scope and on tables that allow configuration records from another application scope. For tables that are in a different scope than the business rule record, the types of rules are limited.

- You can create a rule where When is async with any of the following options:
  - Insert, Update, and Delete database operations. You cannot select Query.
  - Set field values actions and scripts (the Script field).
- You can create a rule where When is before with any of the following options:
  - Insert, Update, and Delete database operations. You cannot select Query.
  - Set field values actions only. You cannot write scripts and you cannot abort the database transaction.
- You cannot create any other types of business rules on tables in a different scope.

Business rules on specific tables cannot be accessed by other business rules or scripts.

Global business rules

⚠️ Warning: Consider using script includes instead of global business rules. Script includes load only on request while global business rules load on every page in the system.

Global business rules are business rules where the Table field is set to Global. Global business rules may be accessible on multiple tables and from other scripts, depending on their scope protection. For a global business rule, define the scope protection by setting the Accessible from field:

- This application scope only: prevents applications in a different scope than the business rule from calling this business rule.
- All application scopes: allows any application to call this business rule.

⚠️ Note: Global business rules do not support domain separation.

Scripts in scoped business rules

When you write a script in a business rule, you can access:
• Any script includes and global business rules in the same scope as the business rule.
• Script includes and global business rules that allow applications in a different scope to call them. To call functions from another scope, you must specify the scope of the function.
• For business rules in a unique scope, you can access the scoped system APIs only.

Create a business rule
You can create any type of business rule to run when a record is displayed, inserted, updated, or deleted, or when a table is queried.

Procedure
1. Navigate to System Definition > Business Rules.
2. Click New.
3. Fill in the fields, as appropriate.

⚠️ Note: You might need to configure the form to see all fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a name for the business rule.</td>
</tr>
<tr>
<td>Table</td>
<td>Select the table that the business rule runs on.</td>
</tr>
<tr>
<td></td>
<td>⚠️ Note: The list shows only tables and database views that meet the scope protections for business rules. Business rules defined for a database view can run only on Query. A business rule for a database view cannot run on insert, update, or delete.</td>
</tr>
<tr>
<td>Application</td>
<td>Application that contains this business rule.</td>
</tr>
<tr>
<td>Accessible from</td>
<td>Scope protection for a global business rule.</td>
</tr>
<tr>
<td></td>
<td>⚠️ Note: This field is visible only when the Table field is set to Global. It does not apply to rules that run on specific tables.</td>
</tr>
<tr>
<td>Active</td>
<td>Select this check box to enable the business rule.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Advanced</td>
<td>Select this check box to see the advanced version of the form.</td>
</tr>
<tr>
<td>When to run</td>
<td></td>
</tr>
<tr>
<td>When</td>
<td>[Advanced] Select when this business rule should execute: display, before, async, or after the database operation is complete.</td>
</tr>
<tr>
<td></td>
<td>Note: Consider setting the Priority for async business rules as the system uses this value when creating the associated scheduled job.</td>
</tr>
<tr>
<td></td>
<td>Note: Newly created async business rules run automatically on upgrade.</td>
</tr>
<tr>
<td>Order</td>
<td>[Advanced] Enter a number indicating the sequence in which this business rule should run. If there are multiple rules on a particular activity, the rules run in the order specified here, from lowest to highest.</td>
</tr>
<tr>
<td>Insert</td>
<td>Select this check box to execute the business rule when a record is inserted into the database.</td>
</tr>
<tr>
<td>Update</td>
<td>Select this check box to execute the business rule when a record is update.</td>
</tr>
<tr>
<td>Delete</td>
<td>[Advanced] Select this check box to execute the business rule when a record is deleted from the database.</td>
</tr>
<tr>
<td>Query</td>
<td>[Advanced] Select this check box to execute the business rule when a table is queried.</td>
</tr>
<tr>
<td>Filter Conditions</td>
<td>Use the condition builder to determine when the business rule should run based on the field values in the selected Table. You can also use the Condition field to build a condition with a script.</td>
</tr>
<tr>
<td></td>
<td>Note: Filters based on string compares are case-sensitive.</td>
</tr>
<tr>
<td>Role Conditions</td>
<td>Select the roles that users who are modifying records in the table must have for this business rule to run.</td>
</tr>
<tr>
<td>Actions</td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Set field values   | Set values for fields in the selected Table using the choice lists:  
• The field  
• The assignment operator:  
  ◦ *To*: An exact value  
  ◦ *Same as*: The value of another field  
  ◦ *To (dynamic)*: A value relative to the user configuring the business rule or a user with a specific role  
• The value |
| Add message        | Select this check box and enter a message that appears when this business rule is run                                                        |
| Abort action       | Select this check box to abort the current database transaction. For example, on a before insert business rule, if the conditions are met, do not insert the record into the database.  
If you select this option, you cannot perform additional actions on the record, such as setting field values and running scripts. You can still display a message to users by selecting the Add message check box and composing the message. |
| Advanced           |                                                                                                                                            |
| Condition          | Create a JavaScript conditional statement to specify when the business rule should run. By adding the condition statement to this field, you tell the system to evaluate the condition separately and run the business rule only if the condition is true.  
If you decide to include the condition statement in the Script field or if you use the condition builder, leave this field blank. To have the instance reevaluate the condition statement a second time before running an async business rule, add the system property `glide.businessrule.async_condition_check` and set the value to true. |
| Script             | [Advanced] Create a script that runs when the defined condition is true. The system automatically populates this field with a function name that matches the *When* value.  
• onAfter  
• onAsync  
• onBefore  
• onDisplay |
Field | Description
---|---
| | **Note:** The function name must match the **When** value. For more information and examples, see Scripting in Business Rules.

Related list: Versions

| Versions | Shows all versions of the business rule. Use this list to compare versions or to revert to a previous version.

4. Click **Submit**.

**Global variables in business rules**

Predefined global variables are available for use in business rules.

Use the following predefined global variables to reference the system in a business rule script.

<table>
<thead>
<tr>
<th>Global variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>current</strong></td>
<td>The current state of the record being referenced. Check for null before using this variable.</td>
</tr>
<tr>
<td><strong>previous</strong></td>
<td>The state of the referenced record prior to any updates made during the execution context, where the execution context begins with the first update or delete operation and ends after the script and any referenced business rules are executed. If multiple updates are made to the record within one execution context, previous will continue to hold the state of the record before the first update or delete operation. Available on update and delete operations only. Not available on asynch operations. Check for null before using this variable.</td>
</tr>
<tr>
<td><strong>g_scratchpad</strong></td>
<td>The scratchpad object is available on display rules, and is used to pass information to the client to be accessed from client scripts.</td>
</tr>
<tr>
<td><strong>gs</strong></td>
<td>References to GlideSystem functions.</td>
</tr>
</tbody>
</table>

The variables **current**, **previous**, and **g_scratchpad** are global across all business rules that run for a transaction.
Prevent null pointer exceptions
In some cases, there may not be a current or previous state for the record when a business rule runs, which means that the variables will be null. To check for null before using a variable, add the following code to your business rule:

```java
if (current == null) // to prevent null pointer exceptions.
    return;
```

Define variables
User-defined variables are globally scoped by default. If a new variable is declared in an order 100 business rule, the business rule that runs next at order 200 also has access to the variable. This may introduce unexpected behavior.

To prevent such unexpected behavior, always wrap your code in a function. This protects your variables from conflicting with system variables or global variables in other business rules that are not wrapped in a function. Additionally, variables such as current must be available when a function is invoked in order to be used.

The following script is vulnerable to conflicts with other code. If the variable gr is used in other rules, the value of the variable may unexpectedly change.

```java
var now_GR = new GlideRecord('incident');
gr.query();
while(gr.next()) {
    //do something
}
```

When this script is wrapped in a function, the variable is available only within the function and does not conflict with other functions using a variable named gr.

```javascript
myFunction();

function myFunction() {
    var now_GR = new GlideRecord('incident');
gr.query();
while(gr.next()) {
    //do something
}
}
```
Use business rules and client scripts to control field values

Implement both business rules and client scripts for a field to enable users to set record values properly using both forms and lists, and to see immediate changes to the values in forms as edits are made.

The problem with using only a client script or a business rule to control updates to a field is that fields can be changed on either a form or a list. Client scripts and UI policies run on forms only (client-side) and do not apply to list editing. Allowing list editing with client scripts running on fields in a form can result in incorrect data being saved to the record. For systems in which client scripts or UI policies apply to forms, either disable list editing or create appropriate business rules or access control to control the setting of values in the list editor. A side effect of this is that security measures implemented in client scripts are easy to circumvent. The user only needs to edit the field in a list.

Business rules on a form are not dynamic, the user must update the record for the change to be seen. This makes using client scripts the preferred method for controlling field values on forms.

When using both a business rule and client script to control field values, the update behavior is the same across the system. This means that updated values are not different depending on whether a list of form is used to make the change. This means that the same functionality must be implemented twice, once in a client script and once in a business rule or access control.

Example: Use a business rule to create email addresses during user record import

An organization has a client script that sets the email address for a user to first.last@company.com. The administrator likes this because he can see the email address immediately when he enters the user's information. The administrator then performs a bulk import of users from a spreadsheet containing the users' first and last names. His expectation is that each user's email address will be set automatically, as they are when he edits the form. Since the client script runs only on the form (the interface to the record), it has no effect on data imported into the record from outside that interface, and no email addresses are created. To solve this problem, the administrator implements a business rule that runs when the import occurs and creates the email addresses.

Example: Prevent list edit for a field that is not editable in the form

An organization wants to hide the Priority field on an incident form if the assignment group is Development. They create a UI policy on the incident form to do this, but their users can still see and edit the Priority field using the list editor. To rectify this, apply an access control to prevent read access to the Priority field when the assignment group is Development.
Using NULL as a field value

The string NULL has a particular role in scripts and is a reserved word. The reserved word is NULL in all capital letters. A field with the value Null or null, for example, is acceptable. Only use NULL to clear out a particular field.

Any NULL field values obtained from an import set data source are inserted into the staging table as empty field values. You should not use the term NULL as a field value in import set transform maps or anywhere in the First name or Last name fields. Also, do not use NULL in reference fields as the system interprets the value as a string containing the word NULL, not as a reserved word.

Display business-rules

Display rules are processed when a user requests a record form.

The data is read from the database, display rules are executed, and the form is presented to the user. The current object is available and represents the record retrieved from the database. Any field changes are temporary since they are not yet submitted to the database. To the client, the form values appear to be the values from the database; there is no indication that the values were modified from a display rule. This is a similar concept to calculated fields.

The primary objective of display rules is to use a shared scratchpad object, g_scratchpad, which is also sent to the client as part of the form. This can be useful when you need to build client scripts that require server data that is not typically part of the record being displayed. In most cases, this would require a client script making a call back to the server. If the data can be determined prior to the form being displayed, it is more efficient to provide the data to the client on the initial load. The form scratchpad object is an empty object by default, and used only to store name:value pairs of data.

To populate the form scratchpad with data from a display rule:

```javascript
// From display business rule
g_scratchpad.someName = "someValue";
g_scratchpad.anotherName = "anotherValue";

// If you want the client to have access to record fields not being displayed on the form
g_scratchpad.created_by = current.sys_created_by;

// These are simple examples, in most cases you'll probably perform some other
// queries to test or get data
```

To access the form scratchpad data from a client script:
Task Active State Management business rule

This business rule determines whether the active field value needs to change based on changes to the State field.

The Task Active State Management business rule is executed when the State is changed for a task record. Its execution order is 50, and it runs before most other task business rules.

If the current task table has the close_states attribute defined on its table, or if it is inherited from a higher-level table, then the rule determines whether the active field needs to change. This is done by comparing the previous and current state values.

- If the state changes from an active state to an inactive state, the Active field is set to false.
- If the state changes from an inactive state to an active state, the Active field is set to true, effectively re-activating or re-opening the task.

It is recommended that you leverage the (current.active.changesTo([true/false])) action in your business rule, as opposed to creating rules on each task table that mark tasks as inactive or active.

Example business rule scripts

Find an example business rule script that helps you with a requirement of your organization.

Compare date fields in a business rule

It is possible to compare two date fields or two date and time fields in a business rule, and abort a record insert or update if they are not correct.

For example, you may want a start date to be before an end date. The following is an example script:

```javascript
if (!current.u_date1.nil() && !current.u_date2.nil()) {
    var start = current.u_date1.getGlideObject().getNumericValue();
    var end = current.u_date2.getGlideObject().getNumericValue();
    if (start > end) {
        gs.addInfoMessage('start must be before end');
        current.u_date1.setError('start must be before end');
    }
}
```
This example has been tested in global scripts, and may need changes to work in scoped scripts. In addition to possibly needing API changes, security is more strict in scoped scripts.

As a good practice, make the business rule a before rule for insert and update actions. In the example script:

• `u_date1` and `u_date2` are the names of the two date fields. Replace these names with your own field names.

• The first line checks that both fields actually have a value.

• The next two lines create variables that have the dates' numerical values.

• The next two lines create different alert messages for the end user: one at the top of the form and one by the `u_date1` field in the form.

• The last line aborts the insert or update if the date fields are not correct.

Here is a more complex example of the above comparison. If you have more than one pair of start and end dates, you can use arrays as shown. Additionally, this script requires the input dates to be within a certain range, in this case, no fewer than 30 days in the past and no more than 365 days in the future.

```javascript
// Enter all start and end date fields you wish to check, as well as the previous values // Make sure that you keep the placement in the sequence the same for all pairs
var startDate = new Array(current.start_date, current.work_start);
var prevStartDate = new Array(previous.start_date, previous.work_start);
var endDate = new Array(current.end_date, current.work_end);
var prevEndDate = new Array(previous.end_date, previous.work_end);

// The text string below is added to the front of 'start must be before end'
var userAlert = new Array('Planned', 'Work');

// Set the number of Previous Days you want to check
var pd = 30;
// Set the number of Future Days you want to check
var fd = 365;

// You shouldn't have to modify anything below this line

var nowdt = new GlideDateTime();
nowdt.setDisplayValue(gs.nowDateTime());
var nowMs = nowdt.getNumericValue();
var pdMs = nowMs;
```
// Subtract the product of previous days to get value in milliseconds
pdms -= pd * 24 * 60 * 60 * 1000;
var fdms = nowMs;

// Add the product of future days to get value in milliseconds
fdms += fd * 24 * 60 * 60 * 1000;
var badDate = false;

// Iterate through all start and end date / time fields
for (x = 0; x < startDate.length; x++) {
    if (((startDate[x].nil()) && (endDate[x].nil()))) {
        var start = startDate[x].getGlideObject().getNumericValue();
        var end = endDate[x].getGlideObject().getNumericValue();
        if (start > end) {
            gs.addInfoMessage(userAlert[x] + ' start must be before end');
            startDate[x].setError(userAlert[x] + ' start must be before end');
            badDate = true;
        } else if (prevStartDate[x] !== startDate[x]) {
            if (start < pdms) {
                gs.addInfoMessage(userAlert[x] + ' start must be fewer than ' + pd + ' days ago');
                startDate[x].setError(userAlert[x] + ' start must be fewer than ' + pd + ' days ago');
                badDate = true;
            }
            else if (prevEndDate[x] !== endDate[x]) {
                if (end > fdms) {
                    gs.addInfoMessage(userAlert[x] + ' end must be fewer than ' + fd + ' days ahead');
                    endDate[x].setError(userAlert[x] + ' end must be fewer than ' + fd + ' days ahead');
                    badDate = true;
                }
            }
        }
    }
}
if (badDate == true) {
    current.setAbortAction(true);
}

Parse XML payloads

Fields in XML format can be parsed with the system's `getXMLText` function.

Fields that get inserted into the database in XML format, such as the payload of an `ecc_event` row, can be parsed with the system’s `getXMLText` function. The `getXMLText` function takes a string and an XPATH expression. For example:

```javascript
var name = gs.getXMLText("<name>joe</name>", "//name");
```

returns the string 'joe'.
Assuming that the field "payload" contains XML, the function call might look like:

```javascript
var name = gs.getXMLText(current.payload, "//name");
```

For information on XPATH, visit w3schools.

**Abort a database action in a before business-rule**

In a before business rule script, you can cancel or abort the current database action using the `setAbortAction()` method.

For example, if the before business rule is executed during an insert action, and you have a condition in the script that calls `current.setAbortAction(true)`, the new record stored in current is not created in the database. The business rule continues to run after calling `setAbortAction()` and all subsequent business rules will execute normally. Calling this method only prevents the database action from occurring.

You can use the `isActionAborted()` method to determine if the current database action (insert, update, delete) is going to be aborted. `isActionAborted()` is initialized for new threads and the `next()` method explicitly sets its value to false.

> Note: `setAbortAction()` can only be executed from the same scope as the record whose action is being aborted. `current.setAbortAction` is not honored if executed in a business rule that is defined in a different scope.

**Determine the operation that triggered the business rule**

You can write a script for a business rule that is triggered on more than one database action.

If you want the business rule script to dynamically branch depending on the action that triggered the event, you can use the `operation()` function. For example:

```javascript
if(current.operation() == "update") {
    current.updates ++;
} else if(current.operation() == "insert") {
    current.updates = 0;
}
```

**Use an OR condition in a business rule**

An **OR** condition can be added to any query part within a business rule.

An **OR** condition can be added to any query part within a business rule with the `addOrCondition()` method. The example below shows a query for finding all the incidents that have either a 1 or a 2 priority. The first `addQuery()` condition is defined as a variable and is used in the **OR** condition.
var inc = new GlideRecord('incident');
var qc = inc.addQuery('priority','1');
qc.addOrCondition('priority','2');
inc.query();
while(inc.next()) {
    // processing for the incident goes here
}

The following script is a more complex example, using two query condition variables doing the equivalent of (priority = 1 OR priority = 2) AND (impact = 2 OR impact = 3). The results of the OR condition are run with two variables, qc1 and qc2. This allows you to manipulate the query condition object later in the script, such as inside an IF condition or WHILE loop.

var inc = new GlideRecord('incident');
var qc1 = inc.addQuery('priority','1');
qc1.addOrCondition('priority','2');
var qc2 = inc.addQuery('impact','2');
qc2.addOrCondition('impact','3');
inc.query();
while(inc.next()) {
    // processing for the incident goes here
}

Reference a Glide list from a business rule

A field defined as a glide list is an array of values stored in a single field.

Here are some examples of how to process a glide_list field when writing business rules. Generally a glide_list field contains a list of reference values to other tables.

Examples

For example, the Watch list field within tasks is a glide_list containing references to user records.

The code below shows how to reference the field.

```javascript
// list will contain a series of reference (sys_id) values separated by a comma
// array will be a javascript array of reference values
var list = current.watch_list.toString();
var array = list.split(",");
for (var i=0; i < array.length; i++) {
    gs.print("Reference value is: " + array[i]);
}
```
You can also get the display values associated with the reference values by using the `getDisplayValue()` method as shown below.

```javascript
// list will contain a series of display values separated by a comma
// array will be a javascript array of display values
var list = current.watch_list.getDisplayValue();
var array = list.split(",");
for (var i=0; i < array.length; i++) {
    gs.print("Display value is: " + array[i]);
}
```

Output:

- Display value is: Abel Tuter
- Display value is: Ashley Leonesio
- Display value is: Charles Beckley
- Display value is: Cherie Fuhri

**Use `indexOf("searchString")` to find a string in a Glide list**

Use `indexOf("searchString")` to return the location of the string passed into the method if the glide list field, such as a Watch list, has at least one value in it.

If the field is empty, it returns `undefined`. To avoid returning an undefined value, do any of the following:

- Force the field to a string, such as:
  ```javascript
  watch_list.toString().indexOf("searchString")
  ```
- Check for an empty Glide list field with a condition before using `indexOf()`, such as: `if (watch_list.nil() || watch_list.indexOf("searchString") == -1)`

**Lock user accounts**

You can lock user accounts if the user is not active.

The following business rule script locks user accounts if the user is not active in the LDAP directory or the user does not have self-service, itil, or admin access to the instance.
/ Lock accounts if bcNetIDStatus != active in LDAP and user does not have self-service, itil or admin role
var rls = current.accumulated_roles.toString();
if(current.u_bcnetidstatus == 'active' && (rls.indexOf(',itil,') > 0 ||
    rls.indexOf(',admin,') > 0 ||
    rls.indexOf(',ess,') > 0 )) {
    current.locked_out = false; }
else {
    current.locked_out = true; }

var now_GR = new GlideRecord("sys_user");
gr.query();
while(gr.next()) {
    gr.update();
    gs.print("updating " + gr.getDisplayValue());
}

Default before-query business rule
You can use a query business rule that executes before a database query is made.
Use this query business rule to prevent users from accessing certain records. Consider the following example from a default business rule that limits access to incident records.
• Name: incident query
• Table: Incident
• When: before, query
• Script:

if(!gs.hasRole("itil") && gs.isInteractive()) {
    var u = gs.getUserId();
    var qc =
current.addQuery("caller_id",u).addOrCondition("opened_by",u).addOrCondition("watch_list", "CONTAINS",u);
    gs.print("query restricted to user: " + u); }

This example prevents users from accessing incident records unless they have the itil role, or are listed in the Caller or Opened by field. So, for example, when self-service users open a list of incidents, they can only see the incidents they submitted.
Note: You can also use access controls to restrict the records that users can see.

Script includes

Script includes are used to store JavaScript that runs on the server.

Create script includes to store JavaScript functions and classes for use by server scripts. Each script include defines either an object class or a function.

Consider using script includes instead of global business rules because script includes are only loaded on request.

Script include form

Script includes have a name, description and script. They also specify whether they are active or not, and whether they can be called from a client script.

Procedure

To access script includes, navigate to System Definitions > Script Includes.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the script include. If you are defining a class, this must match the name of the class, prototype, and type. If you are using a classless (on-demand) script include, the name must match the function name.</td>
</tr>
<tr>
<td>Client callable</td>
<td>Makes the script include available to client scripts, list/report filters, reference qualifiers, or if specified as part of the URL.</td>
</tr>
<tr>
<td>Application</td>
<td>The application where this script include resides.</td>
</tr>
<tr>
<td>Accessible from</td>
<td>Sets which applications can access this script include:</td>
</tr>
<tr>
<td></td>
<td><strong>All application scopes</strong>&lt;br&gt;Can be accessed from any application scope.</td>
</tr>
<tr>
<td></td>
<td><strong>This application scope only</strong>&lt;br&gt;Can be accessed only from the current application scope.</td>
</tr>
<tr>
<td>Active</td>
<td>Enables the script include when selected. Uncheck the active field to disable the script include.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Description</td>
<td>Provides descriptive content regarding the script include.</td>
</tr>
<tr>
<td>Script</td>
<td>Defines the server side script to run when called from other scripts. The script must define a single JavaScript class or a global function. The class or function name must match the Name field.</td>
</tr>
<tr>
<td>Package</td>
<td>The package that contains this script include.</td>
</tr>
<tr>
<td>Created by</td>
<td>The user who created this script include.</td>
</tr>
<tr>
<td>Updated by</td>
<td>The user who most recently updated this script include.</td>
</tr>
<tr>
<td>Protection policy</td>
<td>Sets the level of protection for the script include:</td>
</tr>
<tr>
<td></td>
<td><strong>None</strong></td>
</tr>
<tr>
<td></td>
<td>Allows anyone to read and edit this downloaded or installed script include.</td>
</tr>
<tr>
<td></td>
<td><strong>Read-only</strong></td>
</tr>
<tr>
<td></td>
<td>Allows anyone to read values from this downloaded or installed script include. No one can change script values on the instance on which they download or install the script include.</td>
</tr>
<tr>
<td></td>
<td><strong>Protected</strong></td>
</tr>
<tr>
<td></td>
<td>Provides intellectual property protection for application developers. Customers who download the script include cannot see the contents of the script field. The script is encrypted in memory to prevent unauthorized users from seeing it in plain text.</td>
</tr>
</tbody>
</table>

**Related lists on the form view:**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Versions</td>
<td>Shows all versions of the script include. Use this list to compare versions or to revert to a previous version. See Versions.</td>
</tr>
</tbody>
</table>

**Use script includes**

Script includes are found under System Definition or System UI. You can call existing script includes from a script.

To create an entirely new script include, you can follow the format of any of the existing script includes. In the example, the name of your Script Include is 'NewInclude' and there is a single function called 'myFunction.' It is important that the name of the script include match the name of the class, prototype,
and type. When you create a new script include and give it a name, the system provides you a code snippet with the class and prototype set up properly.

```javascript
var NewInclude = Class.create();

NewInclude.prototype = {
  initialize: function() {},

  myFunction: function() { // Put function code here,
  type: 'NewInclude';

You could then use the 'myFunction' line like this:

```javascript
var foo = new NewInclude();
foo.myFunction();
```

**Note:** Try not to modify a ServiceNow supplied script include. If you want a script include that does something similar to an existing one, copy it and make changes to the copy or consider extending the object. This is a common practice when using GlideAjax.

### Privacy settings

The privacy setting for a client-callable script-include can be public or private. Most client-callable script-includes are marked private by default.

The private privacy-setting means that guests who access public pages cannot access the client-callable script-include. A private script cannot be executed by a non-logged-in user.

A public privacy-setting means that the client script can be executed by non-logged-in users that create an appropriate HTTP request. This can create a security problem if the client script provides confidential information.

The following script includes remain public by default because public pages need to access them:

- GlideSystemAjax
- SysMessageAjax
- KnowledgeMessagingAjax
- KnowledgeAjax
- PasswordResetAjax
Change privacy on all client-callable script includes
Change the privacy setting on all client-callable script includes.

To provide further control over all client-callable script includes, administrators can add the `glide.script.ccsi.ispublic` property. This property changes the visibility of client-callable script includes by making them all public or private. Configure the property as follows:

<table>
<thead>
<tr>
<th>Title</th>
<th>Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>glide.script.ccsi.ispublic</td>
</tr>
<tr>
<td>Type</td>
<td>true</td>
</tr>
<tr>
<td>Value</td>
<td>false</td>
</tr>
</tbody>
</table>

**Note:** To learn more about this property, see [Privacy on client-callable script includes](#) in Instance Security Hardening Settings.

Change privacy on a single client callable script include
Change the privacy setting for a single client-callable script include by adding the `isPublic()` function.

The `isPublic()` setting takes precedence over the `glide.script.ccsi.ispublic` property. For example, if the property is set to `false`, making all client-callable script-includes private, and a script sets `isPublic()` to `true`, the script is public.

To change the privacy for a single client-callable script include, add the following method to the script include:

```javascript
isPublic:function(){return[true/false];},
```

Make the NewInclude client script private.

```javascript
var NewInclude =Class.create();
NewInclude.prototype={
    initialize:function(){},
    myFunction:function()//Put function code here),
    isPublic:function(){return false;},
    type:'NewInclude'};
```
Using discovery script includes

Discovery script includes define JavaScript classes that you can use to accomplish Discovery tasks.

Using GlideRecordUtil to Work with GlideRecords

GlideRecordUtil is a utility class that provides methods that are useful for working with GlideRecords during Discovery.

Getting a GlideRecord Instance

To get a GlideRecord instance for a given configuration item, and of the correct class and table, use the `getCIGR(sys_id)` method. For example, the following code gets the GlideRecord of a CI with the sys_id of 2dfd7c8437201000deeabfc8bcbe5d56:

```javascript
var now_GR = new GlideRecordUtil().getCIGR("2dfd7c8437201000deeabfc8bcbe5d56");
```

To retrieve any hierarchical table without knowing its class type, use the `getGR(base_table, sys_id)` method. For instance, if you need to get a GlideRecord for a computer class CI, you may not know if it is a computer class or more specifically a Windows server class or Linux server class. Using this method guarantees that you have a GlideRecord with the correct class. This is important because different classes have different attributes—in this case, a Windows server has attributes different from those of a Linux server. You must get a GlideRecord in the correct class or attributes may be missing. The following is a typical example of how to do this:

```javascript
var now_GR = new GlideRecordUtil().getGR("cmdb_ci_computer", "2dfd7c8437201000deeabfc8bcbe5d56");
```

Getting All the Fields In a GlideRecord

The `getFields(now_GR)` method returns a JavaScript object, such as a hash map, of all the fields or attributes that exist in a given GlideRecord.

```javascript
var now_GR = new GlideRecordUtil().getGR("cmdb_ci_computer", "2dfd7c8437201000deeabfc8bcbe5d56");
var fields = new GlideRecordUtil().getFields(now_GR);
gs.log(fields.join(" ")); // List all the fields that are in a computer CI
```

Populating GlideRecord Object Fields

The `populateFromGR(hashmap, gr, ignore)` method allows you to take a GlideRecord object and populate its fields and values into a JavaScript object. The third argument (`ignore`) is an optional JavaScript object that allows you to exclude certain fields. For example, you may not care about `sys_created_by` or `sys_updated_by` fields in a GlideRecord.
The `mergeToGR(hashmap, gr, ignore)` method allows you to populate a GlideRecord with a field/value-paired object. The ignore argument stops specified fields from being updated. The following code example updates a computer record's `name` and `os` fields, but does not update the `sys_created_by` field:

```javascript
var now_GR = new GlideRecordUtil().getGR("cmdb_ci_computer", "2dfd7c843721000deeabfcbcbce5d56");
var obj = {"name": "xyz", "os": "windows 2000", "sys_created_by": "aleck.lin"};
var ignore = {"sys_created_by": true};
new GlideRecordUtil().mergeToGR(obj, gr, ignore);
gr.update();
```

**Getting Table Hierarchies**

The `getTables(table)` method returns a list of table hierarchies, as shown in the following example:

```javascript
var tables = new GlideRecordUtil().getTables("cmdb_ci_linux_server");
gs.log(tables.join(','));
// The result would be "cmdb_ci, cmdb_ci_computer, cmdb_ci_server, cmdb_ci_linux_server".
```

**Using DiscoveryException and AutomationException**

When writing Discovery sensors and sensor-related scripts, you may want to use DiscoveryException or AutomationException to indicate that an exception has come from Discovery.

The `DiscoveryException` script include extends `AutomationException`, which extends the `GenericException` class. The following example uses `DiscoveryException` to throw an exception:

```javascript
function foo() {
    if(//condition matches) throw new DiscoveryException("The message", "The cause");
}
```

The first argument takes the message of the exception and the second argument (optional) takes the cause of the exception. You may also want to catch the exception and log it as shown in the example below:

```javascript
try {
    foo();
}
```
The above example also applies for `AutomationException`. `DiscoveryException` is typically used to provide exception processing specifically for Discovery, while `AutomationException` is used for exception processing that applies to both Orchestration and Discovery.

**UI pages**

UI pages can be used to create and display forms, dialogs, lists and other UI components.

Use UI pages as widgets on dashboards. To find the UI pages, navigate to **System UI > UI Pages**.

This functionality requires a knowledge of HTML or Jelly. You can also create simple AngularJS applications using UI pages. To learn more, see the Building Apps with AngularJS developer training.
The UI page form provides the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Input Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name used to invoke the page via a URL (must not contain spaces).</td>
</tr>
<tr>
<td>HTML</td>
<td>Main component of the page, and it defines what will be rendered when the page is shown. It can contain either static XHTML or dynamically generated content defined as Jelly, and it can call script includes and UI Macros.</td>
</tr>
</tbody>
</table>
| Client Script  | Client-side JavaScript that runs in the browser (e.g., functions called by buttons, etc.). It is intended to handle any client-side processing needed, for example setting focus to a field, or other interactive DHTML features after a page is loaded. Ultimately, a UI page's Client Scripts are deployed to the browser within a `<script/>` tag,
UI page (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Input Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>so it could be defined within the page's HTML field to achieve the same effect. Using the Client Script field instead to define these scripts makes things much more tidy and readable though, and it keeps the Jelly and HTML from becoming unmanageable.</td>
</tr>
<tr>
<td>Processing Script</td>
<td>Script that runs on the server when the page is submitted. This is useful if your page has a form (defined with the <code>&lt;g:ui_form/&gt;</code> or <code>&lt;g:form/&gt;</code> tags).</td>
</tr>
<tr>
<td>obsolete-custom-processors</td>
<td>Note: This feature is deprecated. While legacy, custom processors will continue to be supported, creating new custom processors has been deprecated. Instead, please use the Scripted REST APIs. The following information is left in the documentation for existing processors only.</td>
</tr>
</tbody>
</table>

Related lists on the form view:

<table>
<thead>
<tr>
<th>Field</th>
<th>Input Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Versions</td>
<td>Shows all versions of the UI page. Use this list to compare versions or to revert to a previous version.</td>
</tr>
</tbody>
</table>

A UI page can be secured by creating an ACL with the following parameters:

- **Type**: ui_page
- **Operation**: read
- **Name**: name of the UI page to be protected

For details on creating an ACL rule, see Create an ACL rule.

**UI page access**

Each UI page has a URL computed from the application scope, page name, and the .do file extension.

For example, to display the page called glidewindow_example on the demo system, you would navigate to https://<instance name>.service-now.com/glidewindow_example.do. If the page was part of a custom application called example_app, you would instead navigate to https://<instance name>.service-now.com/x_example_app_glidewindow_example.do.

You can also add additional parameters to a URL that can be accessed within a page's HTML section as jelly variables. That is, appending arguments to the URL as follows: /my_test_page.do?sysparm_verbose=true creates jelly variables called *verbose* that can be accessed as follows:
A common practical example of this might be retrieving a database record for display. To build a list of a user's roles, pass in a parameter with the user's sys_id. 

Invoke the following UI page to display a list of roles for that user with Jelly code:

```
role_select.do?sysparm_user=5137153cc611227c000bbd1bd8cd2007
```

```
<j:set var = "jvar_user_id" value = "${sysparm_user}" />

<g:evaluate> var userRoles = new GlideRecord('sys_user_has_role');
    userRoles.addQuery('user','${jvar_user_id}');
    userRoles.query();
</g:evaluate>

<select id='select_role'>
    <j:while test = "${userRoles.next()}">
        <option value = "${userRoles.sys_id}" > ${userRoles.role.name} </option>
    </j:while>
</select>
```

An exception to be careful of, though, is the reserved variable name sys_id. This variable always contains the ID of the UI page itself, regardless of what is specified in the URL. A common substitute variable name is sysparm_id.

Do not use URL parameters to load client scripts in UI pages. The system no longer evaluates scripts that are passed by URL parameter. If your implementation depends on this behavior, you can add the system property [glide.security.disable_ui_pages_sysparm_client_script] and set it to false to temporarily allow the evaluation of URL parameters passing scripts in UI pages.

**UI page process scripts**

If your UI page contains a form (uses the <g:form> tag), you can submit the form and have the process script run.

The processing script can naturally access fields on the form. For example, if your form contained the application_sys_id field:

```
<g:ui_form><p>Click OK to run the processing script.</p> <g:dialog_buttons_ok_cancel
    ok = "return true"/> <input type = "hidden" name = "application_sys_id" value = "499836460a0a0b1700003e7ad950b5da"/>
</g:ui_form>
```

You could then access it using just application_sys_id:

ℹ️ Note:

- GlideSession replaces Packages.com.glide.sys.GlideSession
var application = new GlideRecord('hr_application');
application.get(application_sys_id);
application.status = "Rejected";
application.update();
var urlOnStack = GlideSession.get().getStack().bottom();
response.sendRedirect(urlOnStack);

If you are using the UI page for a dialog, you can also reference the most recent URL on the stack using the code above and then send the response to that location. This is useful if you want to have the dialog's processing script update something and then redisplay the screen that brought up the dialog.

**UI Macros**

UI macros are discrete scripted components administrators can add to the user interface.

UI macros are typically controls that provide inputs or information not provided by existing field types. By default, the system provides UI macros for a variety of user interface elements such as:

- All formatters
- The Service Catalog cart
- The action icons next to fields
- The action icons on forms and lists
- The widgets of a content management system
- The Orchestration activity designer

Administrators can create their own UI macros to provide custom controls or interfaces. Creating UI macros requires knowledge of *Jelly script*. Review the existing UI macros for examples and suggested approaches. Those who want to build custom interfaces with JavaScript technologies should consider Service Portal as an alternative.

**Calling UI macros**

Administrators can call UI macros from certain record types associated with the user interface.
### Calling UI Macros by record type

<table>
<thead>
<tr>
<th>Record type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dictionary attribute</td>
<td>Display an icon for a reference field:</td>
</tr>
<tr>
<td></td>
<td><code>ref_contributions=ui_macro_name</code></td>
</tr>
<tr>
<td>UI page</td>
<td>Display something on a UI page:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;g:macro_invoke macro=&quot;ui_macro_name&quot; /&gt;</code></td>
</tr>
<tr>
<td>UI macro</td>
<td>Call a UI macro from another UI macro:</td>
</tr>
</tbody>
</table>
|                   | `<?xml version= "1.0" encoding= "utf-8"&nbsp;?><j:jelly trim= "false" xmlns:j="jelly:core" xmlns:g="glide"
xmlns:j2="null" xmlns:g2="null">
  <g:ui_macro_name />
  <g:ui_macro_name_2 />
</j:jelly>` |

### UI macro form

Each UI macro record consists of a name and an XML document written in Jelly code.

### UI macro fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A unique and descriptive name for this macro.</td>
</tr>
<tr>
<td>Active</td>
<td>Select the check box to render the element as defined. Clear the check box to disable the element without deleting the code. For example, the <code>email_reply</code> macro is inactive by default.</td>
</tr>
<tr>
<td>Description</td>
<td>Describe the purpose of the macro and parameters passed to it.</td>
</tr>
<tr>
<td>XML</td>
<td>Jelly script that defines the macro.</td>
</tr>
</tbody>
</table>

### Processors

Processors provide a customizable URL endpoint that can execute arbitrary server-side JavaScript code and produce output such as TEXT, JSON, or HTML. Creating custom processors is deprecated.
Note: This feature is deprecated. While legacy, custom processors will continue to be supported, creating new custom processors has been deprecated. Instead, please use the Scripted REST APIs. The following information is left in the documentation for existing processors only.

Warning: When creating a processor, ensure that you use parameter names that are specific to your processor. For example, if your processor exports a list of legal records, and a necessary parameter is the recipient's email address, don't use “email” as the parameter name. Create a more processor specific parameter name, such as legal_export_recipient_email. Failure to do so, and using instance parameter names, such as id, table, sys_id, service, catalog_id, or view (and others), can cause unexpected results.

When to create processors

Do not create custom processors. This feature is deprecated. Please use the REST APIs instead of creating custom processors. The remaining information is left for existing processors only.

Processor form

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Unique name of the processor.</td>
</tr>
<tr>
<td>Type</td>
<td>Programming language of the processor script. Options include:</td>
</tr>
<tr>
<td></td>
<td>• java: do not select this option</td>
</tr>
<tr>
<td></td>
<td>• script</td>
</tr>
<tr>
<td>Application</td>
<td>Application containing this record.</td>
</tr>
<tr>
<td>Active</td>
<td>Flag to enable or disable the record.</td>
</tr>
<tr>
<td>CSRF protect</td>
<td>Option to protect the processor from running unless the instance uses a CSRF token.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the processor's function or purpose.</td>
</tr>
<tr>
<td>Parameters</td>
<td>List of available input parameters. Specify parameter values in the URL as &lt;parameter name&gt;=&lt;parameter value&gt;.</td>
</tr>
</tbody>
</table>
### Processor API components

Processors have access to dedicated API classes, objects, and methods.

#### Processor API components

<table>
<thead>
<tr>
<th>Class, object, or method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>g_response</td>
<td>An object of type HttpServletResponse. See GlideServletResponse.</td>
</tr>
</tbody>
</table>
Processor API components (continued)

<table>
<thead>
<tr>
<th>Class, object, or method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>setContentType('text/html;charset=UTF-8')</td>
<td>A GlideServletResponse method to set the content type of the response being sent to the client.</td>
</tr>
<tr>
<td>g_request</td>
<td>An object of type HttpServletRequest. See HttpServletRequest.</td>
</tr>
<tr>
<td>getParameter()</td>
<td>A glide method to get the value of a URL parameter.</td>
</tr>
<tr>
<td>canRead()</td>
<td>A GlideRecord method to determine if the user can read data from a table. See GlideRecord.</td>
</tr>
<tr>
<td>g_processor</td>
<td>A simplified servlet for processors. See GlideScriptedProcessor.</td>
</tr>
<tr>
<td>writeOutput()</td>
<td>A GlideScriptedProcessor method to display information on the client.</td>
</tr>
<tr>
<td>g_target</td>
<td>An object containing the target table name of a processor URL. For example, a processor containing the URI incident.do applies to the Incident table.</td>
</tr>
</tbody>
</table>

Secure and protect a processor

You can protect your processor against unauthorized use by using role restrictions, and protect it by requiring a CSRF token.

About this task

⚠️ Note: This feature is deprecated. While legacy, custom processors will continue to be supported, creating new custom processors has been deprecated. Instead, please use the Scripted REST APIs. The following information is left in the documentation for existing processors only.

You can re-use a table’s user role restrictions to protect it from access by your processor. This protection method assumes the processor will access table data.

Procedure

1. Create or select a user role that has access to the table the processor script calls.

2. Navigate to System Definition > Processors.

3. In Script, add the following code block.
4. Update the code block to use other access restrictions as needed.
   Available access functions include:
   • canCreate()
   • canRead()
   • canWrite()
   • canDelete()

5. Click Update.

Protect a processor with a CSRF token
You can protect a processor by requiring a CSRF token.

About this task
Script type processors can require a CSRF token check before the processor runs.

Procedure
1. Navigate to System Definition > Processors.
2. Open a processor record.
3. Select the CSRF protect option.
4. Click Update.

Create a simple processor
Create a simple processor to execute a script from a URL query. This feature is deprecated.
About this task

⚠ **Note:** This feature is deprecated. While legacy, custom processors will continue to be supported, creating new custom processors has been deprecated. Instead, please use the Scripted REST APIs. The following information is left in the documentation for existing processors only.

The following steps assume that you have your own demonstration instance.

**Procedure**

1. Navigate to **System Definition > Processors**.
2. Click **New**.
3. Enter the following information.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Hello</td>
</tr>
<tr>
<td>Type</td>
<td>Script</td>
</tr>
<tr>
<td>Path</td>
<td>Hello</td>
</tr>
</tbody>
</table>
| Script | var name= g_request.getParameter("name"); 
g_processor.writeOutput("text/plain","Hello +name"); |

4. Click **Submit**.
5. Enter a URL query to the instance with the following format: https://instance.service-now.com/processor_name.do?parameter=value. For example: https://instancename.service-now.com/Hello.do?name=world.

**Create a multi-table processor**

Create a multi-table processor that reports the number of rows in any table on your instance. This feature is deprecated.

About this task

⚠ **Note:** This feature is deprecated. While legacy, custom processors will continue to be supported, creating new custom processors has been deprecated. Instead, please use the Scripted REST APIs. The following information is left in the documentation for existing processors only.
The multi-table processor protects itself from performance and security violations by confirming that the user is authorized to read the table. It does not report on certain tables that are too large to query safely.

**Procedure**

1. Navigate to *System Definition > Processors.*
   The list of processors appears.

2. Click **New.**

3. Enter the following information.

<table>
<thead>
<tr>
<th>Name</th>
<th>TableSize</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Choose Javascript</td>
</tr>
<tr>
<td>Description</td>
<td>Return number of records in a table</td>
</tr>
<tr>
<td>Parameters</td>
<td>SIZE</td>
</tr>
<tr>
<td>Path</td>
<td>&lt;leave empty&gt;</td>
</tr>
</tbody>
</table>

**Script**

```javascript
G_response.setContentType('text/html;charset=UTF-8');
if(g_target === 'sys_email' || g_target === 'sys_log') {
    g_processor.writeOutput(g_target + ' table is too large to quickly count');
} else {
    var count = new GlideAggregate(g_target);
    if( count.canRead() ) {
        count.addAggregate('COUNT');
        count.query();
        var records = 0;
        if (count.next()) {
            records = count.getAggregate('COUNT');
        }
        g_processor.writeOutput('table ' + g_target + ' has ' + records + ' records');
    } else {
        g_processor.writeOutput('You do not have access to table ' + g_target);
    }
}
```

4. Click **Save.**

5. Test the new processor by entering the following URLs:
   https://<instancename>.service-now.com/incident.do?SIZE
https://<instancename>.service-now.com/sys_user.do?SIZE
Your instance reports the number of records in the table. For example, table incident has 82 records.

Create a custom processor
You can create a custom processor to execute a script from a URL query. This feature is deprecated.

About this task

Note: This feature is deprecated. While legacy, custom processors will continue to be supported, creating new custom processors has been deprecated. Instead, please use the Scripted REST APIs. The following information is left in the documentation for existing processors only.

When complete, you will be able to:

• Create a new processor
• Run a script from a URL query

The following example steps assume you have your own demonstration instance.

Procedure
1. Navigate to System Definition > Processors.
2. Click New.
3. For Name, enter Hello.
4. For Type, select script.
5. For Path, enter Hello.
6. For Script, enter the following code.

```javascript
var name = g_request.getParameter("name");
g_processor.writeOutput("text/plain", "Hello " + name);
```
7. Click Submit.
8. Logout of the instance and open a new browser window.
9. Enter a URL query to the instance with the following format: https://instance.service-now.com/processor_name.do?parameter=value. For example: https://<instance name>.service-now.com/Hello.do?name=world.
10. When prompted, enter credentials for valid user.
**Scripts - Background module**

Administrators can use the Scripts - Background module to run arbitrary JavaScript code from the server.

The Scripts - Background module consists of the following components.

- A text field to enter JavaScript
- A selector to specify the application scope
- A **Run script** button
- A list of available scripts
- A **Record for rollback?** check box. Selected by default, this creates a rollback context for the script execution. After the script is run, click the **Script execution and recovery available here** link to go to the **Script Execution History** form where you can rollback the script.

Administrators can run any valid JavaScript that uses the Glide API. The system displays results, information, and error messages at the top of the screen.

ℹ️ **Note:** Running free-form JavaScript can cause system disruption or data loss. Do not run free-form scripts from a production instance.

By default, administrators can access this module without elevating privileges. If you want to require elevated privileges to access this module, set the system property `glide.script_processor.admin` to `security_admin`.

**Installation settings**

Installation settings are global business rules with calculated names. Installation settings are calculated just before a record is displayed and facilitate dynamic determination of access and roles. Installation Settings permit the programmatic determination of a setting.

Installation settings controlling access to fields and records are:

- `CanRead()`
- `CanWrite()`
- `CanCreate()`
- `CanDelete()`

Functions can return true if access is permitted, false if not. No return value uses the permission calculated using roles. The function has access to the current record through the variable current code.
The name of the function checking the permission on a record is formed by prefixing the setting name with the record name:

```
record_nameCanRead()
```

Similarly, the permission on a field in a record is formed by prefixing the function name with the record name, underscore, and field name:

```
record_name_field_nameCanRead()
```

Naming examples:

```
function incidentCanWrite() {} // can user write to this record?
function incident_numberCanWrite() {} // can user write to the number field?
```

This sample business rule restricts the writing of the name field in the sys_dictionary file when the entry exists:

```
// the element name cannot be written unless this is a new record (not yet in database)
function sys_dictionary_nameCanWrite() {
    if (current.isNewRecord())
        return;

    return false;
}
```

**Using DurationCalculator to calculate a due date**

Using the DurationCalculator script include, you can calculate a due date, using either a simple duration or a relative duration based on schedules.

The following script demonstrates how to use the global API `DurationCalculator` to calculate a due date. The first part of the script illustrates how to set a start datetime using the `setStartDateTime()` method and then use the `calcDuration()` method to determine a due date that is "x" amount of continuous time (seconds) from the specified start datetime. The second half of the script illustrates how to use `DurationCalculator` to calculate a due date based on a schedule. Schedules enable you to apply a "filter" on future time, such as only including the days in a work week within the calculation. For example, if you apply a schedule "weekdays" (which only includes Monday through Friday) to your duration calculation, and the start datetime is Friday at 5:00 pm, when you add a duration of two days, your due date would be Tuesday at 5:00 pm. If you did not use a schedule, your due date would be Sunday at 5:00 pm. For additional information on schedules, see Creating and using schedules.

This script can be cut and pasted into the Scripts Background page and run as is. It can also serve as an example for authoring business rules, UI actions, or used any other place that server-side script can be authored.
/**
 * Demonstrate the use of DurationCalculator to compute a due date.
 *
 * You must have a start date and a duration. Then you can compute a
 * due date using the constraints of a schedule.
 */

gs.include('DurationCalculator');
executeSample();

/**
 * Function to house the sample script.
 */
function executeSample(){

    // First we need a DurationCalculator object.
    var dc = new DurationCalculator();

    // --------------- No schedule examples ------------------

    // Simple computation of a due date without using a schedule. Seconds
    // are added to the start date continuously to get to a due date.
    var gdt = new GlideDateTime("2012-05-01 00:00:00");
dc.setStartDateDateTime(gdt);
if(!dc.calcDuration(2*24*3600)){ // 2 days
    gs.log("*** Error calculating duration");
    return;
}
gs.log("calcDuration no schedule: " + dc.getEndDateDateTime()); // "2012-05-03 00:00:00"

    // Start in the middle of the night (2:00 am) and compute a due date 1 hour in the
    // future
    // Without a schedule this yields 3:00 am.
    var gdt = new GlideDateTime("2012-05-03 02:00:00");
dc.setStartDateDateTime(gdt);
if(!dc.calcDuration(3600)){
    gs.log("*** Error calculating duration");
    return;
}
gs.log("Middle of night + 1 hour (no schedule): " + dc.getEndDateDateTime()); // No
scheduled start date, just add 1 hour
// -------------- Add a schedule to the date calculator ---------------------
addSchedule(dc);

// Start in the middle of the night and compute a due date 1 hour in the future.
// Since we start at 2:00 am the computation adds the 1 hour from the start
// of the day, 8:00am to get to 9:00am
var gdt = new GlideDateTime("2012-05-03 02:00:00");
dc.setStartDateTime(gdt);
if(!dc.calcDuration(3600)){
    gs.log("*** Error calculating duration");
    return;
}

gs.log("Middle of night + 1 hour (with 8-5 schedule): " + dc.getEndDateTime()); // 9:00 am

// Start in the afternoon and add hours beyond quitting time. Our schedule says the work day
// ends at 5:00pm, if the duration extends beyond that, we roll over to the next work day.
// In this example we are adding 4 hours to 3:00pm which gives us 10:00 am the next day.
var gdt = new GlideDateTime("2012-05-03 15:00:00");
dc.setStartDateTime(gdt);
if(!dc.calcDuration(4*3600)){
    gs.log("*** Error calculating duration");
    return;
}

gs.log("Afternoon + 4 hour (with 8-5 schedule): " + dc.getEndDateTime()); // 10:00 am.

// This is a demo of adding 2 hours repeatedly and examine the result. This
// is a good way to visualize the result of a due date calculation.
var gdt = new GlideDateTime("2012-05-03 15:00:00");
dc.setStartDateTime(gdt);
for(var i=2; i<24; i+=1){
    if(!dc.calcDuration(i*3600)){
        gs.log("*** Error calculating duration");
        return;
    }
    gs.log("add " + i +" hours gives due date: " + dc.getEndDateTime());
}

// Setting the timezone causes the schedule to be interpreted in the specified timezone.

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Using DurationCalculator to compute a simple duration

A simple duration is the number of seconds between two date times. If no schedule is used then this is a simple time date subtraction. If a schedule is used, then the schedule is consulted to remove non-work hours from the computation. Suppose schedule “8-5 weekdays excluding holidays” is used. In this case, the number of work hours from noon Monday to noon Tuesday is nine hours. To compute a simple duration, initialize the global API DurationCalculator and call the calcScheduleDuration() method.

This script demonstrates how to use DurationCalculator to compute a simple duration.
/**
  * Sample script demonstrating use of DurationCalculator to compute simple durations
  *
  */

gs.include('DurationCalculator');
executeSample();

/**
  * Function to house the sample script.
  */
function executeSample()
{
  // First we need a DurationCalculator object.
  var dc = new DurationCalculator();

  // Compute a simple duration without any schedule. The arguments
  // can also be of type GlideDateTime, such as fields from a GlideRecord.
  var dur = dc.calcScheduleDuration("2012-05-01", "2012-05-02");
gs.log("calcScheduleDuration no schedule: " + dur); // 86400 seconds (24 hours)

  // The above sample is useful in limited cases. We almost always want to
  // use some schedule in a duration computation, let's load a schedule.
  addSchedule(dc);

  // Compute a duration using the schedule. The schedule
  // specifies a nine hour work day. The output of this is 32400 seconds, or
  // a nine hour span.
  dur = dc.calcScheduleDuration("2012-05-23 12:00:00","2012-05-24 12:00:00");
gs.log("calcScheduleDuration with schedule: " + dur); // 32400 seconds (9 hours)

  // Compute a duration that spans a weekend and holiday. Even though this
  // spans three days, it only spans 9 work hours based on the schedule.
  dur = dc.calcScheduleDuration("2012-05-25 12:00:00", "2012-05-29 12:00:00");
gs.log("calcScheduleDuration with schedule spanning holiday: " + dur); // 32400 seconds (9 hours)

  // Use the current date time in a calculation. The output of this is
  // dependent on when you run it.
  var now = new Date();
dur = dc.calcScheduleDuration("2012-05-15", new GlideDateTime());
gs.log("calcScheduleDuration with schedule to now: " + dur); // Different on every run.
}
Using relative duration

Relative duration is very similar to simple duration except a piece of script is used to determine what parts of a day to remove from the difference calculation.

This script is stored in table cmn_relative_duration and can be examined by navigating to System Scheduler > Schedules > Relative Durations. There are some example relative duration scripts in the out-of-the-box instance.

A relative duration sys_id is passed to the method calcRelativeDuration() of the global API DurationCalculator class after initialization. When this method is called, the DurationCalculator object is passed to the relative duration script (stored in table cmn_relative_duration) as the variable calculator. So, the relative duration script you write and store in cmn_relative_duration has access to the executing DurationCalculator through the variable calculator.

The following script demonstrates how to use DurationCalculator to calculate a relative duration.

```javascript
/**
 * Sample use of relative duration calculation.
 *
 */

gs.include('DurationCalculator');
executeSample();
```
function executeSample(){

    // First we need a DurationCalculator object. We will also use
    // the out-of-box relative duration "2 bus days by 4pm"
    var dc = new DurationCalculator();
    var relDur = "3bf802c20a0a0b52008e2859cd8abcf2"; // 2 bus days by 4pm if before 10am
    addSchedule(dc);

    // Since our start date is before 10:00am our result is two days from// now at 4:00pm.
    var gdt = new GlideDateTime("2012-05-01 09:00:00");
    dc.setStartDateTime(gdt);
    if(!dc.calcRelativeDuration(relDur)){
        gs.log("*** calcRelativeDuration failed");
        return;
    }
    gs.log("Two days later 4:00pm: "+ dc.getEndDateTime());

    // Since our start date is after 10:00am our result is three days from
    // now at 4:00pm.
    var gdt = new GlideDateTime("2012-05-01 11:00:00");
    dc.setStartDateTime(gdt);
    if(!dc.calcRelativeDuration(relDur)){
        gs.log("*** calcRelativeDuration failed");
        return;
    }
    gs.log("Three days later 4:00pm: "+ dc.getEndDateTime());}

function addSchedule(durationCalculator){
    // Load the "8-5 weekdays excluding holidays" schedule into our duration calculator.
    var scheduleName ="8-5 weekdays excluding holidays";
    var grSched =new GlideRecord('cmn_schedule');
    grSched.addQuery('name', scheduleName);
    grSched.query();
    if(!grSched.next()){  
        gs.log("*** Could not find schedule "+ scheduleName +"'");
    }
}
Querying tables in script
Using methods in the GlideRecord API, you can return all the records in a table, return records based on specific conditions or keywords, or return records from multiple tables with a single query.

⚠️ Note: Before querying a table or an index group, you must configure text indexing and search attributes for one or more tables you want to search. For more information, see Configure a single table for indexing and searching or Configure multiple tables for indexing and searching.

Query tables using the GlideRecord API. For API reference, see GlideRecord - Scoped.

Return records in a table
To query a table, first create a GlideRecord object. To create a GlideRecord, create the following in script:

```javascript
var target = new GlideRecord('incident');
```

Creating a GlideRecord creates a target variable which is a GlideRecord object for the incident table.

To process all records from the incident table, add the following script:

```javascript
target.query(); // Issue the query to the database to get all records
while (target.next()) {
    // add code here to process the incident record
}
```

This script issues the query() to the database. Each call to next() would load the next record for processing.

Return records in a table based on query conditions
Most of the time, you want to retrieve a specific record or a specific set of records, and you have some criteria (query conditions) that define the records you want to obtain. For example, say that you want to obtain all the incident records that have a priority value of 1. Here is the code that would accomplish that.

```javascript
var target = new GlideRecord('incident');
target.addQuery('priority',1);
target.query(); // Issue the query to the database to get relevant records
while (target.next()) {
    // add code here to process the incident record
}
```
Notice that the example script includes `target.addQuery('priority', 1);`. This line indicates that you only want the records where the `priority` field is equal to 1. In general, most queries that you want to perform are equality queries; queries where you want to find records with a field equal to a value. For this reason, you do not need to provide an equality operator. However, let's say you wanted to find all incidents where the `priority` field is greater than 1. In this case, you would provide the operator that you want to apply to the query.

```javascript
var target = new GlideRecord('incident')
target.addQuery('priority','>',1);
target.query(); // Issue the query to the database to get relevant records
while (target.next()) {
    // add code here to process the incident record
}
```

### Return records in a table based on a keyword

Use the '123TEXTQUERY321' reserved value to query a single table. For example, this script returns records from the Incident table that include the 'email' keyword.

```javascript
var now_GR = new GlideRecord('incident');
gr.addQuery('123TEXTQUERY321', 'email');
gr.query();
```

'123TEXTQUERY321' is a reserved value for the `name` parameter in the `addQuery()` method, or you can use it in an encoded query string. For example, instead of `addQuery()`, you can also use `gr.addEncodedQuery('123TEXTQUERY321=<search-term>')`.

### Return records from multiple tables

Use the '123TEXTINDEXGROUP321' reserved value to search for a string in a text index group. A text index group is a group of tables that the system indexes together and searches across. For more information about text index groups, see Configure multiple tables for indexing and searching.

ℹ️ **Note:** To search across multiple tables, they must all use the V4 indexing format.

For example, this script returns records that include the keyword 'email' in the 'portal' index group.

```javascript
var now_GR = new GlideRecord('kb_knowledge');
gr.addQuery('123TEXTQUERY321', 'email');
```
You would then create a query for each table in the index group and merge the results.

'123TEXTINDEXGROUP321' is a reserved value for the name parameter in the addQuery() method, or you can use it in encoded query string. For example, instead of addQuery(), you can use gr.addEncodedQuery('123TEXTINDEXGROUP321=<index-group>').

**Available JavaScript operators**

Describes the operators that can be used within an addQuery() request.

<table>
<thead>
<tr>
<th>Field</th>
<th>Definition</th>
<th>addQuery</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>Field must be equal to value supplied.</td>
<td>addQuery('priority', '=', 1);</td>
</tr>
<tr>
<td>&gt;</td>
<td>Field must be greater than value supplied.</td>
<td>addQuery('priority', '&gt;', 1);</td>
</tr>
<tr>
<td>&lt;</td>
<td>Field must be less than value supplied.</td>
<td>addQuery('priority', '&lt;', 3);</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Field must be equal or greater than value supplied.</td>
<td>addQuery('priority', '&gt;=', 1);</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Field must be equal or less than value supplied.</td>
<td>addQuery('priority', '&lt;=', 3);</td>
</tr>
<tr>
<td>!=</td>
<td>Field must not equal the value supplied.</td>
<td>addQuery('priority', '!=', 1);</td>
</tr>
<tr>
<td>STARTSWITH</td>
<td>Field must start with the value supplied. The example shown on the right returns all records where the short_description field starts with the text Error.</td>
<td>addQuery('short_description', 'STARTSWITH', 'Error');</td>
</tr>
<tr>
<td>CONTAINS</td>
<td>Field must contain the value supplied somewhere in the text. The example shown on the right returns all records where the short_description field contains the text Error anywhere in the field.</td>
<td>addQuery('short_description', 'CONTAINS', 'Error');</td>
</tr>
<tr>
<td>Field</td>
<td>Definition</td>
<td>addQuery</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>IN</td>
<td>Takes a map of values that allows commas, and gathers a collection of records that meet some other requirement. Behaves as: <code>Select * from &lt;table&gt; where short_description IN ('Error', 'Success', 'Failure')</code>, which is identical to <code>Select * from &lt;table&gt; where short_description='Error'</code>. For example, to query all variable values that belong to a specific Activity, use the <code>IN</code> clause, and store their sys_ids in a map, or comma-separated list. Then query the variable value table and supply this list of sys_ids.</td>
<td><code>addQuery('short_description', 'IN', 'Error,Success,Failure');</code></td>
</tr>
<tr>
<td>ENDSWITH</td>
<td>Field must terminate with the value supplied. The example shown on the right returns all records where the <code>short_description</code> field ends with text Error.</td>
<td><code>addQuery('short_description', 'ENDSWITH', 'Error');</code></td>
</tr>
<tr>
<td>DOES NOT CONTAIN</td>
<td>Selects records that do NOT match the pattern in the field. This operator does not retrieve empty fields. For empty values, use the operators &quot;is empty&quot; or &quot;is not empty.&quot; The example shown on the right returns all records where the <code>short_description</code> field does not have the word &quot;Error.&quot;</td>
<td><code>addQuery('short_description', 'DOES NOT CONTAIN', 'Error');</code></td>
</tr>
<tr>
<td>NOT IN</td>
<td>Takes a map of values that allows commas, and gathers a collection of records that meet some other requirement. Behaves as: <code>Select * from &lt;table&gt; where short_description NOT IN ('Error')</code>.</td>
<td><code>addQuery('short_description', 'NOT IN', 'Error,Success,Failure');</code></td>
</tr>
<tr>
<td>INSTANCEOF</td>
<td>Special operator that retrieves only records of a specified &quot;class&quot; for extended tables. The code example on the right, shows how to retrieve all <code>cmdb_ci_computer</code>.</td>
<td><code>addQuery('sys_class_name', 'INSTANCEOF', 'cmdb_ci_computer');</code></td>
</tr>
</tbody>
</table>
Available JavaScript Operators (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Definition</th>
<th>addQuery</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>configuration items that are classified as computers.</td>
<td></td>
</tr>
</tbody>
</table>

For additional information on the operators that are available for filters and queries, see Operators available for filters and queries.

There are also some special methods that you can use to search for data that is NULL or NOT NULL. To search for all incidents where the `short_description` field has not been supplied (is null), use the following query:

```javascript
var target = new GlideRecord('incident');
target.addNullQuery('short_description');
target.query(); // Issue the query to the database to get all records
while (target.next()) {
    // add code here to process the incident record
}
```

To find all incidents in which a `short_description` has been supplied, use the following query:

```javascript
var target = new GlideRecord('incident');
target.addNotNullQuery('short_description');
target.query(); // Issue the query to the database to get all records
while (target.next()) {
    // add code here to process the incident record
}
```

For more information on the GlideRecord API and its available methods, see GlideRecord.

GlideRecord query examples

These examples demonstrate how to perform various GlideRecord queries.

```javascript
query

var rec = new GlideRecord('incident');
rec.query();
while(rec.next()) {
    gs.print(rec.number + ' exists');
}
```
**update**

```javascript
var rec = new GlideRecord('incident');
rec.addQuery('active',true);
rec.query();
while(rec.next()) {
    rec.active = false;
    gs.print('Active incident ' + rec.number + ' closed');
    rec.update();
}
```

**insert**

```javascript
var rec = new GlideRecord('incident');
rec.initialize();
rec.short_description = 'Network problem';
rec.caller_id.setDisplayValue('Joe Employee');
rec.insert();
```

**delete**

```javascript
var rec = new GlideRecord('incident');
rec.addQuery('active',false);
rec.query();
while(rec.next()) {
    gs.print('Inactive incident ' + rec.number + ' deleted');
    rec.deleteRecord();
}
```

**Querying Service Catalog Tables**

You cannot directly query the variables of the Service Catalog Request Item table [sc_req_item]. Instead, query the Variable Ownership table, [sc_item_option_mtom], by adding two queries, one for the variable name and another for the value. The query returns the many-to-many relationship, which you can dot-walk to the requested item. The following example finds the request items that have the variable `item_name` with a value of `item_value` and displays the request item numbers:

```javascript
var now_GR = new GlideRecord('sc_item_option_mtom');
gr.addQuery('sc_item_option.item_option_new.name','item_name');
gr.addQuery('sc_item_option.value','item_value');
gr.query();
while(gr.next()) {
    gs.addInfoMessage(gr.request_item.number); }```

For additional information see GlideRecord.
Running order guides automatically

Service catalog order guides allow customers to make a single service catalog request that can generate several ordered items. Administrators can configure order guides to run automatically, from a workflow or a script to generate a set of ordered items without manually submitting a service catalog request. Administrators can also review and reprocess the order guide failures.

For example, an onboarding workflow for a new employee can run an order guide to automatically order items for that employee.

Running order guides from scripts

Running order guides with a server-side script is more complex than using workflows, but it allows more flexibility and can be used in non-workflow situations.

For example, you can use order guide scripts with UI actions or server-side business rules.

⚠️ Note: When order guides run automatically, order guide UI policies are not enforced. Also, options in the Choose Options screen cannot be selected, so make sure that order guide rules define sensible defaults for these options to avoid processing failures.

Use the `SNC.ScriptableOrderGuide` Java class to run order guides with a script.

Use the `SNC.ScriptableOrderGuide(String orderGuideId)` constructor to create a new `ScriptableOrderGuide` object.

Method summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>process(String json)</code></td>
<td>boolean</td>
<td>Runs the order guide using the JSON encoded string parameter as the input for the order guide. Returns <code>true</code> or <code>false</code> depending on whether processing was successful or not.</td>
</tr>
</tbody>
</table>

⚠️ Note: Both `opened_by` and `requested_for` parameters must be passed to the order guide, and both must have valid user record `sys_id` values.
<table>
<thead>
<tr>
<th>Method</th>
<th>Return Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reprocess(GlideRecord failure)</td>
<td>boolean</td>
<td>Runs the order guide again using the JSON encoded string parameter stored in the failure GlideRecord.</td>
</tr>
<tr>
<td>getMessage()</td>
<td>String</td>
<td>Retrieves the message populated after processing or reprocessing.</td>
</tr>
<tr>
<td>getRequest()</td>
<td>GlideRecord</td>
<td>Retrieves the request GlideRecord.</td>
</tr>
<tr>
<td>getFailure()</td>
<td>GlideRecord</td>
<td>Retrieves the failure GlideRecord from the Scriptable Order Guide Failures [sc_script_order_guide_failure] table.</td>
</tr>
</tbody>
</table>

**Script example**

This script processes an order guide called *IT Onboarding SOG*.

```java
// Creating the object to later be JSON encoded
var json = {
    "opened_by": "62826bf03710200044e0bfc8bcbe5df1", "requested_for": "06826bf03710200044e0bfc8bcbe5d8a", "department": "221f3db5c611228409f4becd3039cc9"
};

var now_GR = new GlideRecord("sc_cat_item_guide");
if (gr.get("name","IT Onboarding SOG")) {
    var sog = new SNC.ScriptableOrderGuide(gr.getValue("sys_id"));
    var result = sog.process(new JSON().encode(json));
    if (!result)
        gs.log("Processing the scriptable order guide failed with message: " + sog.getMessage());
    else {
        var request = sog.getRequest();
        gs.log("Request created - " + request.sys_id); 
    }
}
Running order guides from workflows

Running an order guide from a workflow is suitable if you include order guides as part of a broader workflow-based process.

For example, an activity within an onboarding workflow for a new employee can automatically run an order guide to order items for that employee.

Note: When order guides run automatically, order guide UI policies are not enforced. Also, options in the Choose Options screen cannot be selected, so make sure that order guide rules define sensible defaults for these options to avoid processing failures.

To run order guides from a workflow, use the **Scriptable Order Guide** workflow activity.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order Guide</td>
<td>The name of the order guide that this activity processes. For example, <strong>Example Employee Onboarding IT</strong>.</td>
</tr>
<tr>
<td>Script</td>
<td>A script passing information to the order guide. This information is sent as a JSON encoded string parameter assigned to the answer variable. The script must meet these requirements:</td>
</tr>
<tr>
<td></td>
<td>- The names of the variables in the script must match the names used within the order guide. For example, if the order guide uses a <em>department</em> variable in a rule condition, the script must also pass a <em>department</em> parameter.</td>
</tr>
<tr>
<td></td>
<td>- Both <em>opened_by</em> and <em>requested_for</em> parameters must be passed to the order guide, and both must have valid user record sys_id values.</td>
</tr>
</tbody>
</table>

**Results**

- **Success**: the activity successfully processed the order guide. This does not mean a request was created. If a request was created, the request sys_id is added to the workflow scratchpad under the *sc_request* variable.

- **Failure**: while processing the order guide a failure occurred, creating a failure record. If the processing fails, you can view and edit the failure record.
Workflow example

The Example Employee Onboarding IT Workflow workflow uses this example to generate IT catalog items for a new employee as part of an onboarding process.

The activity uses this script to:

1. Take a JSON string generated previously from the HR change record.
2. Append the mandatory opened_by and requested_for parameters to that string.
3. Submit the new string for processing by the order guide.

```javascript
var parameters = new JSON().decode(current.payload);

// Need to amend the json object to include additional values.
parameters.opened_by = current.opened_by + "";
parameters.requested_for = current.opened_for + "";

answer = new JSON().encode(parameters);
```

View order guide failures

Order guide processing can fail, for example if the order guide being run does not exist. When a failure occurs during the order guide processing, the Scriptable Order Guide Failures submodule allows you to review and reprocess the failures. A record is created for each failure and once you fix the errors that caused the initial failure, you can reprocess the failed order guides.

About this task

If a failure occurs, a failure record is created in the Scriptable Order Guide Failures [sc_script_order_guide_failure] table.

To view details of a failure, navigate to Service Catalog > Catalog Administration > Scriptable Order Guide Failures, then open a failure record.

Reprocessing Failures

If you have fixed the error that caused the initial failure, you can reprocess failed order guides.

Procedure

1. Navigate to Service Catalog > Catalog Administration > Scriptable Order Guide Failures.
2. Open the failure record.
3. Click the Reprocess related link.
To reprocess one or more failures:

a. Navigate to **Service Catalog > Catalog Administration > Scriptable Order Guide Failures**.
b. Select the check box beside one or more records to reprocess.
c. Select **Reprocess** from the **Actions** choice list.

**Scriptable assignment of execution plans**

Each catalog item has an associated execution plan, used whenever an item of that type is ordered; if no plan is specified, the default plan is used. This default is effective for most organizations, but your execution plan may need to vary based on additional criteria.

For example, in the base system service catalog, a request for a new PC always uses the PC Delivery Plan. However, this plan may need to vary for unusual circumstances - such as when a requester is working from home, at a remote location.

To provide this flexibility, you can use a script to override the default execution plan on a specific catalog item.

**Limitations during script execution**

Execution plan scripts have limitations that need to be considered during their implementation.

While the execution plan script runs:

- You cannot interact with any catalog tasks as catalog tasks are only created after the execution plan is selected.
- Some fields such as total delivery time and due date are not yet calculated, although the request itself is available within the script via `current.request()`.
- Approvals have not yet been generated.

**Writing the scripts**

Follow these guidelines when writing execution plan scripts.

Execution plan scripts can access the same global variables and other functions as in any other server side execution plan.

- `current` is the currently-requested catalog item, `sc_req_item`.
- `current.delivery_plan()` is the assigned execution plan for this catalog item.
The evaluated value from the script is used as the sys_id of the execution plan.

Simple example:

```javascript
current.delivery_plan.setDisplayValue('PC Delivery Plan')
```

If an invalid value is returned, such as undefined or not found, then the existing assigned value is used.

More complex example:

```javascript
getexecutionplan();
function getexecutionplan() {
  var location = current.request.requested_for.location.getDisplayValue();
  // if we're in Atlanta
  if (location == 'Atlanta') {
    // use the remote pc delivery plan instead of the normal one
    var remote_plan = new GlideRecord('sc_cat_item_delivery_plan');
    remote_plan.addQuery('name', 'Remote PC Delivery Plan');
    remote_plan.query();
    remote_plan.next();
    current.delivery_plan = remote_plan.sys_id;
    return remote_plan_sys_id;
  }
  return current_delivery_plan;
}
```

In this example, any time a request is for a user in Atlanta, ServiceNow uses the Remote PC Delivery Plan. Otherwise, the execution plan is not overridden and ServiceNow uses the catalog item’s normal execution plan, the PC Delivery Plan.

Add a script to a catalog item

You can add a script to a catalog item so that the script runs each time a user requests that item.

Procedure

1. Navigate to Service Catalog > Maintain Items.
2. Select the relevant catalog item to which you wish to add the script.
3. Configure the catalog item form to add the execution plan script field, often named Delivery Plan Script.
4. Fill in the script details.
5. Update the item form with your changes.

Results
The script runs each time that item is requested, selecting the execution plan to run with that item.

Use a script to approve an execution plan
You can use an approval rule script to approve an execution plan.
**Procedure**

1. Retrieve an approval execution plan task.
2. View the **Approval Script** field.
3. Fill in an approval script using the same syntax and rules you would use on an approval rule.

**Example:**
For example, in the script below, the requester’s manager is the approver.

![Approval Script Example](image)

**Scriptable service catalog variables**

You can use scripting to reference any request item variable from a table in scoped and non-scoped environment.

An example of a variable reference follows.

```javascript
current.variables.<variable_name>
```

Where `current` refers to the current record, and `<variable_name>` is the name of your variable.

**Note:** In order to reference a variable from JavaScript, it must have a name.
When a variable is part of a variable set, you can reference it as `current.variables.<variable_name>` or `current.variables.<variable_set_name>.<variable_name>`.

Variable set is also a first-class citizen in Service Catalog. Like variables, a variable set has read, write, and create roles. If roles are provided for a variable set, the roles are applicable for the variables within the set. Roles of an individual variable are overridden by the roles of the variable set.

**Print a variable**

```javascript
var original = current.variables.original_number;
gs.print(original);
```

**Set a variable**

```javascript
current.variables.name = "Auto-Generated:" + current.variables.asset_tag;
```

**Create an inventory item with fields set from variables**

```javascript
doCreation();

function doCreation ( ) {
  var create = current.variables.create_item;
  if (create == 'true') { // we want to create an asset
    var computer = new GlideRecord('cmdb_ci_computer');
    computer.initialize();
    computer.asset_tag = current.variables.asset_tag;
    computer.serial_number = current.variables.serial_number;
    computer.name = current.variables.name;
    computer.manufacturer = current.variables.company;
    computer.insert();
  }
}
```

**Get GlideElementVariable of variables and variable sets associated with a GlideRecord**

```javascript
now_GR.variables
```

**Get the name value pair of variables associated with a GlideRecord**

```javascript
now_GR.variables.getVariableValue();
```

**Get a list of GlideElementVariable for variables within a task record**

```javascript
now_GR.variables.getElements();
```
Get a list of GlideElementVariable for variables (including multi-row variable set) within a task record

```javascript
now_GR.variables.getElements(true);
```

### APIs for GlideElementVariable

- `now_GR.variables.<var_name>.isMultiRow()`: Get whether the GlideElementVariable is a multi-row variable set or a variable.
- `now_GR.variables.<var_name>.getQuestion()`: Get the Question object for a variable. Applicable only for a variable (isMultiRow() is false) and not for a multi-row variable set.
- `now_GR.variables.<var_name>.getLabel()`: Get the label of the GlideElementVariable. For a variable, the label of the variable is returned. For multi-row variable set, the title of the variable set is returned.
- `now_GR.variables.<var_name>.canRead()`: Get whether the user can view a variable or multi-row variable set.
- `now_GR.variables.<var_name>.canWrite()`: Get whether the user can edit a variable or multi-row variable set.
- `now_GR.variables.<var_name>.getDecryptedValue()`: Get the decrypted value for a masked variable. Applicable only for a masked variable.
- `now_GR.variables.<var_name>.getRows()`: Get the list of row objects for a multi-row variable set. Applicable only for a multi-row variable set (isMultiRow() is true).
- `now_GR.variables.<var_name>.getRowCount()`: Get the number of rows for multi-row variable set. Applicable only for a multi-row variable set (isMultiRow() is true).

### Example to access variables of GlideRecord for the Task table

```javascript
var now_GR = new GlideRecord('sc_req_item');
if (now_GR.get('635a1f5387320300e0ef0cf888cb0b73')) {
    var variables = now_GR.variables.getElements();
    for (var i=0;i<variables.length;i++) {
        var question = variables[i].getQuestion();
        gs.log(question.getLabel() + "":" + question.getValue())
    }
}
```
Example to access a multi-row variable set of GlideRecord for the Task table

```javascript
var now_GR = new GlideRecord('sc_req_item');
now_GR.get('02c38dcd87013300e0ef0cf888cb0bb2');

var vars = now_GR.variables.getElements(true);

for (var i=0; i<vars.length; i++) {
    var now_V = vars[i];
    if (now_V.isMultiRow()) {
        var rows = now_V.getRows();
        for (var j=0; j<now_V.getRowCount(); j++) {
            var row = rows[j];
            var cells = row.getCells();
            for (var k=0; k<cells.length; k++) {
                var cell = cells[k];
                gs.info(cell.getLabel() + ':' + cell.getCellDisplayValue());
            }
        }
    }
}
```

### Multi-row variable set

<table>
<thead>
<tr>
<th>Operation</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table operations</td>
<td></td>
</tr>
<tr>
<td>Return JSON array value as String</td>
<td><code>now_GR.variables.table_var</code></td>
</tr>
<tr>
<td>Set value of a multi-row variable set</td>
<td><code>now_GR.variables.table_var = &lt;val&gt;</code></td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: An array of ordered (key, value) pairs is also applicable as input.</td>
</tr>
<tr>
<td>Get value of column, var1, of a multi-row variable set</td>
<td><code>now_GR.variables.table_var.var1</code></td>
</tr>
<tr>
<td>Set value of a variable set, var1</td>
<td><code>now_GR.variables.table_var.var1 = &lt;val&gt;</code></td>
</tr>
</tbody>
</table>
## Multi-row variable set (continued)

<table>
<thead>
<tr>
<th>Operation</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operation</strong></td>
<td><strong>Usage</strong></td>
</tr>
<tr>
<td>&lt;note&gt;<strong>Note:</strong> An array of ordered (key, value) pairs is also applicable as input.&lt;/note&gt;</td>
<td></td>
</tr>
<tr>
<td>Row operations</td>
<td></td>
</tr>
<tr>
<td>Get the current row count</td>
<td><code>now_GR.variables.table_var.getRowCount()</code></td>
</tr>
<tr>
<td>Returns the row specified by the variable &quot;i&quot; - <code>getRow(&lt;int&gt; i)</code></td>
<td><code>var row = now_GR.variables.table_var.getRow(&lt;int&gt; i);</code></td>
</tr>
<tr>
<td>Get the cell value for a question column mapped to <code>&lt;var_name&gt;</code></td>
<td><code>row.&lt;var_name&gt;</code></td>
</tr>
<tr>
<td>Set the cell value for a question column mapped to <code>&lt;var_name&gt;</code></td>
<td><code>row.setCellValue('&lt;var_name&gt;', value)</code></td>
</tr>
<tr>
<td>Set the cell value for a question column mapped to <code>&lt;var_name&gt;</code></td>
<td><code>row.&lt;var_name&gt; = value</code></td>
</tr>
<tr>
<td>Add an empty row at the end of the table and return a scriptable object</td>
<td><code>var row = now_GR.variables.table_var.addRow()</code></td>
</tr>
<tr>
<td>Delete a row</td>
<td><code>row.deleteRow()</code></td>
</tr>
</tbody>
</table>

### Notes and limitations

1. You can only set a variable in a before business rule. Variables set in an after rule are not written to the database.

2. There is nothing in place to prevent namespace collision with variables. Creating two variables named computer_speed would result in only one of them showing up; the second one would overwrite the first one.

3. Date/time variables use the same time zone formatting and storage rules as all other dates in the system. They are stored internally in GMT, but translated into the user's local time zone and format for display.
Setting a GlideRecord variable to null

GlideRecord variables (including current) are initially null in the database. Setting these back to an empty string, a space, or the JavaScript null value will not result in a return to this initial state.

⚠️ Note: This functionality requires a knowledge of JavaScript.

⚠️ Note: Functionality described here requires the Admin role.

To set it back to the initial state, simply set the value to "NULL". Note that the update() function does not run on the current object but rather on the record. The object displays the initial value until it is called again from the record.

Example 1

```javascript
var gr1 = new GlideRecord('incident');
gr1.query();
while(gr1.next()) {
    gr1.priority = "NULL";
    gr1.update();
}
```

Example 2 (Business Rule)

```javascript
current.u_affected_value = 'NULL';
current.update();
```

Wizard scripts

You can create scripts to implement advanced custom functionality using wizards.

Use wizard scripts in record generator panels, transitions, and UI policies.

⚠️ Note: The System Wizards application is not active by default.

To reference a wizard variable in a script, use the format `wizard.<name>`, where <name> is the value in the Name field of the variable definition.

Wizard script examples

These examples demonstrate scripts that use wizard variables.

Example 1

```javascript
//Find a user in the sys_user table with the wizard variable 'EmpName'
function getUser(){
    var gu = new GlideRecord('sys_user');
```
gu.addQuery('sys_id', wizard.EmpName);
gu.query();
if(gu.next()){
    gs.addInfoMessage('Found user ' + gu.name)
}

Example 2

//Loop through items in a wizard list collector with variable name astList
function removeAssets() {
    var items = wizard.astList.toString();
    items = items.split(',');
    for (var i = 0; i < items.length; i++) {
        //got the asset id
        var sys_id = items[i];
    }
}

Example 3

//Set the url on wizard completion - user will be redirected to this location
var uri = ('sys_user_list.do?sysparm_query=sys_id=' + wizard.EmpName);
wizard.redirect = uri;

Wizard scripts on record generators

Use wizard scripts in record generator panels to create records in any table, including on more than one table.
Procedure

1. In the Table field, select Global.
2. Enter a script in the Script field. Configure the form to add the field, if necessary.

Wizard UI policy and client scripts

Create dynamic effects in wizards with UI policies and client scripts. For example, get or set variable values, hide or display variables, make variables mandatory, or validate values.

UI policies apply effects based on conditions constructed with a condition builder. Client scripts accomplish more advanced functionality, but require scripting knowledge. Since UI policies do not require scripting, they are less likely to need maintenance after system updates.

ℹ️ Note: The System Wizards application is not active by default.

Schedule Pages

A schedule page is a record that contains a collection of scripts that allow for custom generation of a calendar or timeline display.
Creation of timeline schedule pages requires understanding of the page/event flow and the ability to write client and server side JavaScript.

**Schedule pages form**

To access schedule pages, navigate to **System Scheduler > Schedules > Schedule Pages**.

The form provides the following fields, depending upon the View Type selected:

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>String</td>
<td>General name that is used to identify the current schedule page.</td>
</tr>
<tr>
<td>Schedule type</td>
<td>String</td>
<td>The schedule type is a string that is used to uniquely identify the schedule page via the &quot;sysparm_page_schedule_type&quot; URI parameter. For example, a schedule page could be accessed as follows: <code>/show_schedule_page.do?sysparm_page_schedule_type=gantt_chart&amp;sysparm_timeline_task_id=d530bf907f0000015ce594fd929cf6a4</code> Alternatively, the schedule page can also be accessed by setting the &quot;sysparm_page_sys_id&quot; URI parameter to that of the unique 32 character hexadecimal system identifier of the schedule page.</td>
</tr>
<tr>
<td>View Type</td>
<td>Choice</td>
<td>Each view type displays different field combinations. There are two options:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Calendars</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Timelines</td>
</tr>
<tr>
<td>Description</td>
<td>String</td>
<td>General description that provides additional information about the current schedule page as necessary.</td>
</tr>
<tr>
<td>Init function name</td>
<td>String</td>
<td>Note: This functionality is only used by Calendar type schedule pages. The init function name specifies the name of the JavaScript function to call inside the Client script function for calendar type schedule pages.</td>
</tr>
<tr>
<td>HTML</td>
<td>String</td>
<td>Note: This functionality is only used by Calendar type schedule pages. The HTML field is a scriptable section that is parsed by Jelly and injected into the display page prior to the rest of the calendar. It can be used to pass in variables from the server and define extra fields as necessary.</td>
</tr>
<tr>
<td>Client script</td>
<td>String</td>
<td>The client script is a scriptable section that allows for configuring options of the schedule page display. The API is different depending on the schedule page view type and is discussed below.</td>
</tr>
</tbody>
</table>
Schedule pages form (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server AJAX processor</td>
<td>String</td>
<td>Note: This functionality is only used by Calendar type schedule pages. The Server AJAX processor is specific to calendar type schedule pages. It is used to return a set of schedule items and spans to be displayed.</td>
</tr>
</tbody>
</table>

Timeline schedule pages

A Timeline Schedule Page is a specific record that contains configuration information for displaying time based points and spans in a "timeline" like fashion.

The timeline schedule page references a script include that extends from AbstractTimelineSchedulePage to perform dynamic modification to the timeline based on different events and conditions. Both the schedule page and the script include for timeline generation allow for extreme customization and their corresponding application programming interface (API) is documented below.

The following diagram shows the series of events that occur when a timeline schedule page is accessed. Once the timeline has been loaded, all subsequent events, such as events resulting from timeline interaction (e.g. moving a timeline span), follow the same logic flow shown in the gray event box.
Timeline Flow

Client browser accesses a schedule page. The request is sent to the server.

http://instance.service-now.com/show_schedule_page.do?sysparm_page_schedule_type=mntb_client4sysparm_timeline_task_id-0f030b6d971f3b6f00160e064d0f6046

The server interprets the schedule page HTTP request and obtains information about the specific schedule page from either of the following URI parameters:

sysparm_page_sys_id
or
sysparm_page_schedule_type

The server returns an HTTP response that contains the client script information from the specified schedule page.

The client browser immediately parses the Client script section of the schedule page upon loading of the page.
- Set configuration and display options
- Register event listeners

Event: getItems

The client timeline makes an AJAX request to the corresponding Script Include for the event registered with "getItems" to retrieve the set of items and spans to display on the timeline.

The server receives the request and executes the getItems code block inside the specified Script Include (which is an instance of the AbstractTimelineSchedulePage class). The server returns an XML document with TimelineItem objects.

The client receives the AJAX response with the specified TimelineItem objects and appropriately displays them on the screen.

Applications that use schedule pages to generate timelines

- Project Management
- Maintenance Schedules
- Group On-Call Rotation
- Field Service Management

Timeline schedule page example

The following example demonstrates how to create a timeline schedule page with corresponding script include utilizing a majority of the API described above.

For this example we are going to create an Incident Summary Timeline for a project support manager to visualize all of the new incidents. All new incidents
should be displayed as single points where the priority of the incident is distinguished by a different point icon. Additionally, all closed incidents should be displayed on the timeline in a separate group that shows the duration of the incident before it was closed. Since the Project Manager wants to be able to easily close new items that are resolved without using any form lists, we will handle the vertical move event allowing the new incidents to be dragged into the closed incident group or items within.

**Schedule Page**
Create a new schedule page with the following properties:

- **Name:** Hardware Incidents
- **Schedule type:** incident_timeline
- **View Type:** Timeline
- **Client Script:**

```javascript
// Set our page configuration
glideTimeline.setReadOnly(false);
glideTimeline.showLeftPane(true);
glideTimeline.showLeftPaneAsTree(true);
glideTimeline.showTimelineText(true);
glideTimeline.showDependencyLines(false);
glideTimeline.groupByParent(true);
glideTimeline.setDefaultPointIconClass('milestone');

// We will define what items to display and provide a custom event handler for moving new items to the closed state
glideTimeline.registerEvent('getItems', 'IncidentTimelineScriptInclude');
glideTimeline.registerEvent('elementMoveY', 'IncidentTimelineScriptInclude');
```

**Script Include**
Now that the schedule page has been created we need to generate a matching script include for the events that were registered. Create a new script include with the following properties:

- **Name:** IncidentTimelineScriptInclude
- **Active:** Checked
- **Client callable:** Checked
- **Script:**

```javascript
// Class Imports
```
var IncidentTimelineScriptInclude = Class.create();
IncidentTimelineScriptInclude.prototype =
Object.extendsObject(AbstractTimelineSchedulePage, {

getItems:function() {
  // Specify the page title
  this.setPageTitle('My Custom Incident Summary Timeline');

  var groupNew = new GlideTimelineItem('new');
  groupNew.setLeftLabelText('New Incidents');
  groupNew.setImage('../images/icons/all.gifx');
  groupNew.setTextBold(true);
  this.add(groupNew);

  var groupClosed = new GlideTimelineItem('closed');
  groupClosed.setLeftLabelText('Closed Incidents');
  groupClosed.setImage('../images/icons/all.gifx');
  groupClosed.setTextBold(true);
  groupClosed.setIsDroppable(true);
  this.add(groupClosed);

  // This allows us to drag an open incident onto the closed group row.
  this.add(groupClosed);

  // Get all the incidents and let's add only the new/closed ones appropriately
  var now_GR = new GlideRecord('incident');
  gr.query();
  while(gr.next()) {
    // Only loop through new/closed incidents
    if(gr.incident_state != '1' && gr.incident_state != '7') continue;

    // Ok, we have a new/closed incident. Create the item and the span first.
    var item = new GlideTimelineItem(gr.getTableName(), gr.sys_id);
    var span = item.createTimelineSpan(gr.getTableName(), gr.sys_id);

    // Specific properties for a new incident
    if(gr.incident_state == '1') { // New
      item.setParent(groupNew.getSysId());
      item.setImage('../images/icons/open.gifx');
      span.setTimeSpan(gr.getElement('opened_at').getGlideObject().getNumericValue(),
        gr.getElement('opened_at').getGlideObject().getNumericValue());

      // We'll show different colors based upon the priorities only for new incidents
      switch(gr.getElement('priority').toString()) {

      }
    }
  }

});
case '1': span.setPointIconClass('red_circle'); break;
case '2': span.setPointIconClass('red_square'); break;
case '3': span.setPointIconClass('blue_circle'); break;
case '4': span.setPointIconClass('blue_square'); break;
case '5': span.setPointIconClass('sepia_circle'); break;
default: // Otherwise, the default point icon class will be used (Milestone) 
}
}

// Specific properties for a closed incident
else if(gr.incident_state == '7') {
    item.setParent(groupClosed.getSysId());
    item.setImage('../images/icons/closed.gifx');
    span.setTimeSpan(gr.getElement('opened_at').getGlideObject().getNumericValue(),
    gr.getElement('closed_at').getGlideObject().getNumericValue()); 
}

    // Common item properties
    item.setLeftLabelText(gr.short_description);

    // Common span properties
    span.setSpanText(gr.short_description);
    span.setTooltip('<strong>' + GlideStringUtil.escapeHTML(gr.short_description) +
    '</strong><br>' + gr.number);
    span.setAllowXMove(false);
    span.setAllowYMove(gr.canWrite() ? true:false);
    span.setAllowYMovePredecessor(false);
    span.setAllowXDragLeft(false);
    span.setAllowXDragRight(false);

    // Now we add the actual item
    this.add(item);
}

ELEMENT_MOVE_Y

/**
 * This is one of the AbstractTimelineSchedulePage event handler methods that corresponds
 * to a vertical
 * move. The arguments for this function are defined in the API section of the event
 * handler methods.
 */
elementMoveY: function(spanId, itemId, newItemId) {
// Get information about the current incident
var now_GR = new GlideRecord('incident');
gr.addQuery('sys_id', spanId);
gr.query();
if(!gr.next())
    return this.setStatusError('Error', 'Unable to lookup the current incident.');

// Only allow the new incidents to have their state adjusted.
if(gr.incident_state != '1')
    return this.setStatusError('Error', 'Only new incidents can have their state adjusted.'TOTYPE

// Get information about the dropped GlideTimelineItem. If it was dropped in an item on
the "New Incidents"
// group let's do nothing. If it was dropped in the "Closed Incidents" then let's
adjust the state automatically.
var grDropped = new GlideRecord('incident');
grDropped.addQuery('sys_id', newItemId);
grDropped.query();
if(!grDropped.next() || grDropped.incident_state == '7') {
    // This means the dropped item was either the 'Closed Incidents' group (which has no
record or sys_id) or an
    // existing incident that is closed. The 'New Incidents' also has no sys_id;
however, the default behavior for
    // items without a sysId is to be non-droppable. This is why we explicitly denoted
the 'Closed Incidents' to
    // be marked as "droppable".

    // Return a dialog prompt
    this.setStatusPrompt('Confirm', 'Are you sure you want to close: ' +
    '<div style="margin:10px 0 10px 14px;padding:4px;background-color:#EBEBEB;">\<strong>\</strong><\div></div>',
    GlideStringUtil.escapeHTML(gr.short_description) +
    '<\strong><br/>\<div class="font_smaller">' + now_GR. number +
    '</div></div>',
    'this._elementMoveYHandler_DoClose', // This function is for when the OK
button is clicked.
    'this._elementMoveYHandler_DoNothing', // This function is for when the
Cancel button is clicked.
    'this._elementMoveYHandler_DoNothing'); // This function is for when the
Close button is clicked.
}
Screenshots / Results

1. Navigate to:

http://[YOURINSTANCE]:8080/show_schedule_page.do?
sysparm_page_schedule_type=incident_timeline

Notice the bold text is the value of the schedule page Schedule type field.

2. The page displays a timeline as specified by the schedule page and script include created. A link to this page can be created and placed as a module or UI action as necessary.
3. Attempting to move a closed incident anywhere displays the expected error message.

4. Moving the incident: I need more memory displays the following confirmation box.

5. Clicking the **Cancel** button closes the overlay. Clicking the **OK** button actually updates the incident_state of the record and then displays the following success box.
6. After clicking **OK**, it is clear the incident is now listed in the **Closed Incidents** group.

**XMLDocument script object**

A JavaScript object wrapper for parsing and extracting XML data from an XML document (String).

Use this Javascript class to instantiate an object from an XML string, usually a return value from a Web Service invocation, or the XML payload of ECC Queue. Using the XMLDocument object in a Javascript business rule lets you query values from the XML elements and attributes directly.

**Constructor**

The constructor of a script object creates a new instance of the object to be used.

```javascript
var xmlString = "<test>" +
    "  <one>" +
```
To perform XML parsing of an XML string that is name space qualified, specify "true" for the second argument for the XMLDocument constructor. The following is an example of parsing and XML string that contains name space qualification of its elements.

```javascript
var xmlString = "<bk:book xmlns:bk='urn:loc.gov:books'
         xmlns:isbn='urn:ISBN:0-395-36341-6'>" +
    "<bk:title>Cheaper by the Dozen</bk:title>" +
    "<isbn:number>1568491379</isbn:number>" +
    "</bk:book>";

var xmldoc = new XMLDocument(xmlString, true); // XML document is name space aware
```

### Locating nodes and elements

Now that we have the XMLDocument object, we can call the following APIs to locate nodes and elements of the XML document, as well as extract text from it directly. The query syntax for locating nodes and attributes is based on XPath.

???? Note: XMP parsing with this object is not namespace aware, this means that if you are locating a node name with namespace in it eg. "<names:myelement> ...", the XPath search string would be "///myelement"

The following are examples of locating a node by its XPath and getting the text value out of the resulting node.

```javascript
var str = xmldoc.getNodeText("/two"); // returns the first occurrence of the node
    // str == "abcd1234"

str = xmldoc.getNodeText("/three");
    // str == "1234abcd"

str = xmldoc.getNodeText("/test/one/*");
    // str == "abcd1234"

str = xmldoc.getNodeInt("/number");
    // str == 1234
```
The following examples locate the node by XPath and uses the API on node and element to get attributes and value.

```javascript
var node = xmldoc.getNode("/test/one/two");
// node.getNodeName() == "two"
// node.getNodeType() == "1" // 1 == ELEMENT_NODE
// node.getLastChild().getNodeType() == "3" // 3 == TEXT_NODE
// node.getLastChild().getNodeValue() == "abcd1234"
```

Or you can use the following convenience functions to get the node name and type

```javascript
str = xmldoc.getNodeName("//three");
// str == "three"

str = xmldoc.getNodeType("//three");
// str == "1"
```

You can also get a list of nodes that you can iterate or access directly by position

```javascript
var nodelist = xmldoc.getNodes("//one/*"); // two, three, and two
// nodelist.getLength() == "3"
// nodelist.item(0).getNodeName() == "two"
// nodelist.item(1).getNodeName() == "three"
```

The following is an example of parsing an XML string that is qualified with namespaces.

```javascript
    "<bk:title>Cheaper by the Dozen</bk:title>" +
    "<isbn:number>1568491379</isbn:number>" +
    "</bk:book>";

var xmldoc = new XMLDocument(xmlString, true);
var str = xmldoc.getNodeText("//bk:title"); // returns the first occurrence of the node
gs.log(str);

str = xmldoc.getNodeText("/bk:book/*");
gs.log(str);

str = xmldoc.getNodeInt("//isbn:number");
gs.log(str);
```

### Getting attribute values

An attribute is just an extension of a node and so it has all the same APIs.
The following example shows how to query for a node to get its attribute by position

```javascript
node = xmldoc.getNode("//two");
// node.getAttributes().item(0).getNodeValue() == "xxx"

str = xmldoc.getAttribute("//two", "att");
// str == "xxx"
```

XPath also has a query syntax for locating the attribute node directly, for example

```javascript
str = xmldoc.getNodeText("//*[@att="yyy"]");
// str == "1234abcd"

str = xmldoc.getNode("//@boo").getNodeValue();
// str == "yah"
```

## Creating new elements

An XML element can be created at any level of the XML document once it has been created. Use the `setCurrent` function to position where the new element will be created as a child element, and use the `createElement` function to create the element.

```javascript
var xmlString = "<test>" +
    "  <one>" +
    "    <two att="xxx">abcd1234</two>" +
    "    <three boo="yah" att="yyy">1234abcd</three>" +
    "    <two>another</two>" +
    "  </one>" +
    "  <number>1234</number>" +
"</test>";

var xmldoc = new XMLDocument(xmlString);
xmldoc.createElement("new", "test"); // creates the new element at the document element level if setCurrent is never called
xmldoc.createElement("new2"); // calling without a value will create a new element by itself

var el = xmldoc.createElement("new3");
xmldoc.setCurrent(el); // this is now the parent of any new elements created subsequently using createElement()
xmldoc.createElement("newChild", "test");
```

The resulting XML document looks like this
XMLHelper

The XML helper script include makes it easy to parse XML in scripts.

The script include converts your XML document into a JavaScript object.

The following changes were made to the XML helper script include:

• The `toObject()` method returns an object whose properties are all JavaScript objects. This method works properly whether the supplied parameter is an XML document or an XML string.

• The methods `toXMLDoc()` and `toXMLStr()` have been made available. These methods are the inverse of the existing `toObject()` method.

• The `toObject()` method has been extended to take an optional parameter of the XML input to convert as an alternative to the (still present) mechanism of specifying the XML input in the constructor.

⚠️ **Note:** You must escape ampersand characters (&amp) in your XML or the conversion silently fails.

Example

The following example takes an example XML document and then converts it into a JavaScript object. It then takes the outputted JavaScript object and uses a recursive function to output all members of the object. The recursive function is useful and reusable if you have any questions about how a particular XML document will be structured after being converted to a JavaScript object.

Script

```javascript
var xmlString = "<company><employee><id>10</id>
  <firstname>Tom</firstname><lastname>Cruise</lastname></employee></company>"
```
<employee>
  <id>20</id>
  <firstname>Paul</firstname>
  <lastname>Enderson</lastname>
  <test>test6</test>
</employee>

<employee>
  <id>30</id>
  <firstname>Paul</firstname>
  <lastname>Bush</lastname>
</employee>

var helper = new XMLHelper(xmlString);
var obj = helper.toObject();

logObj(obj,"");
function logObj(obj, sep){
  for(x in obj){
    if(typeof obj[x]!="function"{
      gs.log(sep + x +": ": + obj[x]);
      logObj(obj[x], sep +"*";});
  }
}

Output

*** Script: *employee:: [object Object],[object Object],[object Object]
*** Script: **2:: [object Object]
*** Script: ***id:: 30
*** Script: ***test:: test2,test4
*** Script: ****0:: test2
*** Script: ****1:: test4
*** Script: ***firstname:: Paul
*** Script: ***lastname:: Bush
*** Script: **0:: [object Object]
*** Script: ***id:: 10
*** Script: ***test:: test1,test3
*** Script: ****0:: test1
*** Script: ****1:: test3
*** Script: ***firstname:: Tom
*** Script: ***lastname:: Cruise
*** Script: **1:: [object Object]
*** Script: ***id:: 20
*** Script: ***test:: test6,test5
*** Script: ****0:: test6
*** Script: ****1:: test5
*** Script: ***firstname:: Paul
*** Script: ***lastname:: Enderson
JavaScript API Context-sensitive help

The syntax editor can display context-sensitive API information. JavaScript API Context-sensitive help includes the ability to:

- List script elements that are valid at the cursor's location. The system displays suggestions in a pop-up window.
- Add a selected script element at the cursor's location. If the cursor is within or adjacent to a partial entry, the system completes the entry with the selected script element.
- View API documentation for a selected suggestion.
- View the expected parameters and format of the current script element.

If the cursor is adjacent to a text string, the system searches for script elements that start with this text string. For example, while the cursor is within or adjacent to the string `GlideR`, the system displays script elements such as:

- GlideRecord
- GlideRecordSecure

Context-sensitive suggestions are based on script type. For example, when working on a business rule, only suggestions from the server API and for objects such as `current` and `previous` display. When working on a client script, the system only displays suggestions from the client API.

Client-side scripting

Run JavaScript on the client (web browser) when client-based events occur, such as when a form loads, after form submission, or when a field changes value.

The client-side Glide API provides classes and methods that you can use in client scripts.

Before you begin

Understand the limitations of your scripting environment. For example, client scripts run on forms in the Service Portal or Mobile environments can only include certain APIs. For more information, see Mobile client GlideForm (g form) scripting and migration.

Client-side scripting design and processing

Well-designed client scripts can reduce the amount of time it takes users to complete a form.
Proper client-side processing depends on the form loading first. Making record updates prior to form load can produce unexpected results that bypass client-side processing.

If you create client scripts to control field values on a form, you must use another method to control these field values in a list. You can:

- Disable list editing for the table.
- Create appropriate business rules or access controls for list editing.
- Create data policies.
- Create a separate onCellEdit client script.

**Restrict list editing**

If you create UI policies or client scripts for fields on a form, you must use another method to ensure that data in those fields is similarly controlled in a list.

With the exception of onCellEdit client scripts, UI policies and client scripts apply to forms only. Use the following methods to restrict list editing when using client scripts:

- Disable list editing for the table.
- Create appropriate business rules or access controls for list editing.
- Create data policies.
- Create a separate onCellEdit client script.

**Minimize server lookups**

Use client data as much as possible to eliminate the need for time-consuming server lookups.

Client scripting uses either data available on the client or data retrieved from the server. The top ways to get information from the server are g_scratchpad and asynchronous GlideAjax lookup.

The primary difference between these methods is that g_scratchpad is sent once when a form is loaded (information is pushed from the server to the client), whereas GlideAjax is dynamically triggered when the client requests information from the server.

**Note:** GlideRecord and g_form.getReference() are also available for retrieving server information. However, these methods are no longer recommended due to their performance impact. Both methods retrieve all fields in the requested GlideRecord when most cases only require one field.
Example: Retrieve server data using g_scratchpad

The g_scratchpad object passes information from the server to the client, such as when the client requires information not available on the form.

For example, if you have a client script that needs to access the field u_retrieve, and the field is not on the form, the data is not available to the client script. A typical solution to this situation is to place the field on the form and then always hide it with a client script or UI policy. While this solution may be faster to configure, it is slower to execute.

If you know what information the client needs from the server before the form is loaded, a display business rule can create g_scratchpad properties to hold this information. The g_scratchpad is sent to the client when the form is requested, making it available to all client-side scripting methods. This is a very efficient means of sending information from the server to the client. However, you can only load data this way when the form is loaded. The business rule cannot be triggered dynamically. In those cases, use an asynchronous GlideAjax call.

For example, assume you open an incident and need to pass this information to the client:

- The value of the system property css.base.color
- Whether or not the current record has attachments
- The name of the caller's manager

A display business rule sends this information to the client using the following script:

```javascript
const g_scratchpad = {);

// Save the information
g_scratchpad.css = gs.getProperty('css.base.color');
g_scratchpad.hasAttachments = current.hasAttachments();
g_scratchpad.managerName = current.caller_id.manager.getDisplayValue();
```

To access scratchpad data using a client script:

```javascript
// Check if the form has attachments
if (g_scratchpad.hasAttachments)
  // do something interesting here
else
  alert('You need to attach a form signed by ' + g_scratchpad.managerName);
```

Example: Retrieve server data using asynchronous GlideAjax

Asynchronous GlideAjax allows you to dynamically request information from the server.

This script compares the support group of the CI and the assignment group of the incident by name:
// Alert if the assignment groups name matches the support group
function onChange(control, oldValue, newValue, isLoading) {

    if (isLoading)
        return;

    var ga = new GlideAjax('ciCheck');
    ga.addParam('sysparm_name', 'getCiSupportGroup');
    ga.addParam('sysparm_ci', g_form.getValue('cmdb_ci'));
    ga.addParam('sysparm_ag', g_form.getValue('assignment_group'));
    ga.getXML(doAlert); // Always try to use asynchronous (getXML) calls rather than synchronous (getXMLWait)
}

// Callback function to process the response returned from the server
function doAlert(response) {

    var answer = response.responseXML.documentElement.getAttribute("answer");

    alert(answer);
}

This script relies on the accompanying script include:

var ciCheck = Class.create();

ciCheck.prototype = Object.extendsObject(AbstractAjaxProcessor, {

    getCiSupportGroup: function() {

        var retVal = ''; // Return value
        var ciID   = this.getParameter('sysparm_ci');
        var agID   = this.getParameter('sysparm_ag');
        var ciRec  = new GlideRecord('cmdb_ci');

        // If we can read the record, check if the sys_ids match
        if (ciRec.get(ciID)) {
            if (ciRec.getValue('support_group') == agID)
                retVal = 'CI support group and assignment group match';
            else
                retVal = 'CI support group and assignment group do not match';
        }

        // Can't read the CI, then they don't match
Use the setValue() displayValue parameter for reference fields

When using setValue() on a reference field, include the displayValue parameter to avoid additional server calls.

When using setValue() on a reference field, be sure to include the reference field display value as the 3rd parameter. If you set the value without the displayValue, the instance does a synchronous call to retrieve the display value for the record you specified. This extra round trip to the server can impact performance.

This example demonstrates the incorrect way to call setValue:

```javascript
var id = '5137153cc611227c000bbd1bd8cd2005';

g_form.setValue('assigned_to', id); // Client needs to go back to the server to
// fetch the name that goes with this ID
```

Instead, include the display value as an optional parameter in setValue():

```javascript
var id = '5137153cc611227c000bbd1bd8cd2005';
var name = 'Fred Luddy';

g_form.setValue('assigned_to', id, name); // No server call required
```

Use UI policy instead of a client script

When possible, consider using a UI policy instead of a client script.

UI policies provide these benefits over client scripts:

- UI policies have an Order field to allow full control over the order in which client-side operations take place.
- UI policies do not require scripting to make a field mandatory, read-only, or visible.

⚠️ Note: UI policies apply after client scripts.
Validate input using client scripts
An excellent use for client scripts is validating input from the user. This validation improves the user experience because the user finds out if there are data issues before submitting the information.

An example of validation is to verify that the **Impact** field value is valid with the **Priority** field value. In this example, **Low** impact is not allowed with **High** priority.

```javascript
if (g_form.getValue('impact') == '3' && g_form.getValue('priority') == '1')
    g_form.showFieldMsg('impact', getMessage('Low impact now allowed with High priority'), 'error');
```

Set client script order
Control the order of execution for your client scripts using the Order field. To avoid having two or more client scripts run concurrently and then conflict, you can add an order for the scripts to run in.

Before you begin
Role required: admin

About this task
Adding an order to the client script creates a processing sequence, ordered from lowest to highest number. If two scripts conflict, the client script with the lower number executes first.

Procedure
1. Navigate to **System Definition > Client Script** and open an existing client script or click **New**.
2. Configure the form to include the **Order** field.
3. Add a number to the order field based on what order you want it to run in relation to other client scripts. Choose a lower number for the script you want to execute first.

Avoid DOM manipulation
Avoid Document Object Model (DOM) manipulation if possible. It can cause a maintainability issue when browsers are updated.

Instead, use the GlideForm API or consider a different approach for the solution. In general, when using DOM manipulation methods, you have to reference an element in the DOM by ID or using a CSS selector. When referencing out-of-box DOM elements, there is a risk that the element ID or placement within the DOM
could change, causing the code to stop working and/or generate errors. Use forethought, caution, and have a full understanding of the risk you are incurring. Review these objects and reduce the use of DOM manipulation methods as much as possible.

Avoid global client scripts

A global client script is any client script where the selected Table is Global. Global client scripts have no table restrictions, therefore they will load on every page in the system introducing browser load delay in the process.

There is no benefit to loading this kind of scripts on every page.

As an alternative, and for a more modular and scalable approach, consider moving client scripts to a base table (such as Task[task] or Configuration Item[cmdb_ci]) that can be derived for all the child/extension tables. This eliminates the system loading the scripts on every form in the UI - such as home pages or Service Catalog where they are rarely (if ever) needed.

Enclose code in functions

Enclose the code in a client script inside a function.

Client scripts without a function cause issues with variable scope. When code is not enclosed in a function, variables and other objects are available and shared to all other client-side scripts. If you are using the same variable names, it is possible that they could collide. This can lead to unexpected consequences that are difficult to troubleshoot.

Consider this example:

```javascript
var state = "6";

function onSubmit() {
    if(g_form.getValue('incident_state') == state) {
        alert("This incident is Resolved");
    }
}
```

Because the `state` variable is not enclosed in a function, all client-side scripts have access to it. Other scripts may also use the common variable name `state`. The duplicate names can conflict and lead to unexpected results. These issues are difficult to isolate and resolve. To avoid this issue, ensure that all code is wrapped in a function:
function onSubmit() {

    var state = "6";

    if(g_form.getValue('incident_state') == state) {
        alert("This incident is Resolved");
    }
}

This solution is much safer because the scope of the variable state is limited to the onSubmit() function. Therefore, the state variable does not conflict with state variables in other client-side scripts.

**Run only necessary scripts**

To avoid running time-consuming scripts unnecessarily, make sure that client scripts perform only necessary tasks.

The following examples demonstrate improvements to the initial code sample. Each example demonstrates a particular enhancement to the script to improve performance and avoid unnecessary calls.

Remember that client scripts have no **Condition** field. This means that onLoad() and onChange() scripts run in their entirety every time the appropriate form is loaded. This example is an inefficient onChange() client script set to run when the Configuration item field changes.

```javascript
//Set Assignment Group to CI's support group if assignment group is empty
function onChange(control, oldValue, newValue, isLoading) {

    var ciSupportGroup = g_form.getReference('cmdb_ci').support_group;

    if (ciSupportGroup != '' && g_form.getValue('assignment_group) != '')
        g_form.setValue('assignment_group', ciRec.support_group.sys_id);
}
```

This example improves upon the first by replacing the getReference() or GlideRecord lookup with an asynchronous GlideAjax call.

```javascript
//Set Assignment Group to support group if assignment group is empty
function onChange(control, oldValue, newValue, isLoading) {

    var ga = new GlideAjax('ciCheck');

    ga.addParam('sysparm_name', 'getSupportGroup');
    ga.addParam('sysparm_ci', g_form.getValue('cmdb_ci'));
```
The `isLoading` flag is the simplest way to prevent unnecessary code from taking up browser time in `onChange` scripts. The `isLoading` flag should be used at the beginning of any script that is not required to run when the form is loading. There is no need to run this script on a form load because the logic would have already run when the field was last changed. Adding the `isLoading` check to the script prevents it from doing a `cmdb_ci` lookup on every form load.

The `isTemplate` flag indicates that a template is loading.

```javascript
// Set Assignment Group to CI's support group if assignment group is empty
function onChange(control, oldValue, newValue, isLoading, isTemplate) {
  if (isLoading)
    return;

  var ga = new GlideAjax('ciCheck');

  ga.addParam('sysparm_name', 'getSupportGroup');
  ga.addParam('sysparm_ci', g_form.getValue('cmdb_ci'));
  ga.getXML(setAssignmentGroup);
}

function setAssignmentGroup(response) {
  var answer = response.responseXML.documentElement.getAttribute("answer");

  g_form.setValue('assignment_group', answer);
}
```

If the `onChange` script should run during loading, use the following convention:

```javascript
function onChange(control, oldValue, newValue, isLoading, isTemplate) {
  if (isLoading) {} // run during loading

  // rest of script here
}
The `newValue` check tells this script to continue only if there is a valid value in the relevant field. This prevents the script from running when the field value is removed or blanked out. This also ensures that there will always be a valid value available when the rest of the script runs.

```javascript
//Set Assignment Group to CI's support group if assignment group is empty
function onChange(control, oldValue, newValue, isLoading, isTemplate) {
    if (isLoading)
        return;

    if (newValue) {
        var ga = new GlideAjax('ciCheck');
        ga.addParam('sysparm_name', 'getSupportGroup');
        ga.addParam('sysparm_ci', g_form.getValue('cmdb_ci'));
        ga.getXML(setAssignmentGroup);
    }
}

function setAssignmentGroup(response) {
    var answer = response.responseXML.documentElement.getAttribute("answer");
    g_form.setValue('assignment_group', answer);
}
```

To have the script react to a value that changes after the form loads, use the `newValue != oldValue` check.

**Note:** This example does not catch users changing a value and then changing it back to its original value.
In this example, the GlideAjax call is buried one level deeper by rearranging the script to check as many things available to the client as possible before running the server calls. The script checks the assignment before executing the GlideAjax call. This prevents the server lookup when the assignment_group field is already set.

```javascript
// Set Assignment Group to CI's support group if assignment group is empty
function onChange(control, oldValue, newValue, isLoading, isTemplate) {
  if (isLoading)
    return;

  if (newValue) {
    if (newValue != oldValue) {
      if (g_form.getValue('assignment_group') == '') {
        var ga = new GlideAjax('ciCheck');
        ga.addParam('sysparm_name', 'getSupportGroup');
        ga.addParam('sysparm_ci', g_form.getValue('cmdb_ci'));
        ga.getXML(setAssignmentGroup);
      }
    }
  }
}

function setAssignmentGroup(response) {
  var answer = response.responseXML.documentElement.getAttribute("answer");
  g_form.setValue('assignment_group', answer);
}
```
Client scripts

Client scripts allow the system to run JavaScript on the client (web browser) when client-based events occur, such as when a form loads, after form submission, or when a field changes value.

Use client scripts to configure forms, form fields, and field values while the user is using the form. Client scripts can:

- make fields hidden or visible
- make fields read only or writable
- make fields optional or mandatory based on the user's role
- set the value in one field based on the value in other fields
- modify the options in a choice list based on a user's role
- display messages based on a value in a field

Introduction to client scripts, script types, APIs, and good practices

Where client scripts run

With the exception of `onCellEdit()` client scripts, client scripts only apply to forms and search pages. If you create a client script to control field values on a form, you must use one of these other methods to control field values when on a list.

- Create an access control to restrict who can edit field values.
- Create a business rule to validate content.
- Create a data policy to validate content.
- Create an `onCellEdit()` client script to validate content.
- Disable list editing for the table.

⚠️ Note: Client scripts are not supported on ServiceNow mobile applications.

Client script form

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the client script.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Table</td>
<td>Table to which the client script applies.</td>
</tr>
<tr>
<td>UI Type</td>
<td>Target user interface to which the client script applies.</td>
</tr>
<tr>
<td>Type</td>
<td><strong>onLoad()</strong> — runs when the system first renders the form and before users can enter data. Typically, <strong>onLoad()</strong> client scripts perform client-side-manipulation of the current form or set default record values.</td>
</tr>
<tr>
<td></td>
<td><strong>onSubmit()</strong> — runs when a form is submitted. Typically, <strong>onSubmit()</strong> scripts validate things on the form and ensure that the submission makes sense. An <strong>onSubmit()</strong> client script can cancel form submission by returning a value of false.</td>
</tr>
<tr>
<td></td>
<td><strong>onChange()</strong> — runs when a particular field value changes on the form. The <strong>onChange()</strong> client script must specify these parameters.</td>
</tr>
<tr>
<td></td>
<td>• <strong>control</strong>: the DHTML widget whose value changed.</td>
</tr>
<tr>
<td></td>
<td>• <strong>oldValue</strong>: the value the widget had when the record was loaded.</td>
</tr>
<tr>
<td></td>
<td>• <strong>newValue</strong>: the value the widget has after the change.</td>
</tr>
<tr>
<td></td>
<td>• <strong>isLoading</strong>: identifies whether the change occurs as part of a form load.</td>
</tr>
<tr>
<td></td>
<td>• <strong>isTemplate</strong>: identifies whether the change occurs as part of a template load.</td>
</tr>
<tr>
<td></td>
<td><strong>onCellEdit()</strong> — runs when the list editor changes a cell value. The <strong>onCellEdit()</strong> client script must specify these parameters.</td>
</tr>
<tr>
<td></td>
<td>• <strong>sysIDs</strong>: an array of the sys_ids for all items being edited.</td>
</tr>
<tr>
<td></td>
<td>• <strong>table</strong>: the table of the items being edited.</td>
</tr>
<tr>
<td></td>
<td>• <strong>oldValues</strong>: the old values of the cells being edited.</td>
</tr>
<tr>
<td></td>
<td>• <strong>newValue</strong>: the new value for the cells being edited.</td>
</tr>
<tr>
<td></td>
<td>• <strong>callback</strong>: a callback that continues the execution of any other related cell edit scripts. If <strong>true</strong> is passed as a parameter, the other scripts are executed or the</td>
</tr>
</tbody>
</table>
change is committed if there are no more scripts. If false is passed as a parameter, any further scripts are not executed and the change is not committed.

Note: onCellEdit() client scripts do not apply to dashboard list widgets.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>Application where this client script resides.</td>
</tr>
<tr>
<td>Active</td>
<td>Enables the client script when selected. Unselect this field to disable the client script.</td>
</tr>
<tr>
<td>Inherited</td>
<td>Indicates whether the client script applies to extended tables.</td>
</tr>
<tr>
<td>Global</td>
<td>If true, the client script runs on all views of the table.</td>
</tr>
<tr>
<td>View</td>
<td>Only visible when <strong>Global</strong> is unselected. Views on which the client script will run.</td>
</tr>
<tr>
<td>Description</td>
<td>Content describing the functionality and purpose of the client script.</td>
</tr>
<tr>
<td>Messages</td>
<td>Text string (one per line) available to the client script as localized messages using <code>getmessage('[[message]]')</code>. For additional information, see Translate a client script message.</td>
</tr>
<tr>
<td>Script</td>
<td>Contains the client script.</td>
</tr>
</tbody>
</table>

**UI scripts**

UI scripts provide a way to package client-side JavaScript into a reusable form, similar to how script includes store server-side JavaScript. Administrators can create UI scripts and run them from client scripts and other client-side script objects and from HTML code.

UI scripts are not supported for mobile.

**Global UI scripts**

You can create a UI script and designate it as global, which makes the script available on any form in the system. You cannot create a global UI script in a scoped application.
You can mark a UI script as Global to make it available on any form in the system. For example, you can create a UI script that has a function `helloWorld()`, and has the Global field checked:

```javascript
function helloWorld() {
    alert('Hi');
}
```

After you create this global UI script, you can call the `helloWorld()` function from any client script or UI policy you write.

**Create a UI script**

Create a UI script to define reusable client-side JavaScript code.

**Procedure**

To create UI scripts, navigate to **System UI > UI Scripts** and create or edit a record (see table for field descriptions).

**Run UI scripts**

Follow these guidelines when running UI scripts.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Script Name</td>
<td>Name of the UI script. Ensure the name is unique on your system.</td>
</tr>
<tr>
<td>API Name</td>
<td>The API name of the UI script, including the scope and script name (for example, x_custom_app.HelloWorld).</td>
</tr>
<tr>
<td>Application</td>
<td>Application that contains the UI script.</td>
</tr>
<tr>
<td>Active</td>
<td>Indicator of whether the UI script is active. Only active UI scripts can run.</td>
</tr>
<tr>
<td>Global</td>
<td>Indicator of whether the script loads on every page in the system.</td>
</tr>
</tbody>
</table>

**Note:** Use caution when creating global UI scripts because they can impact performance. You cannot create a global UI script in a scoped application.
Run a UI script from a form

To run a UI script on a form, create a formatter. In the associated UI macro, include a \texttt{g:requires} tag and specify the \texttt{name=} parameter as the name of the UI script followed by the .jsdbx extension. Add the formatter on the form view. This code ensures that the definitions and results of the UI script are immediately available in the browser.

```xml
<?xml version="1.0" encoding="utf-8" ?>
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null" xmlns:g2="null">
  <g2:evaluate var="jvar_stamp">
    var now_GR = new GlideRecord('sys_ui_script');
    gr.orderByDesc('sys_updated_on');
    gr.query();
    gr.next();
    gr.getValue('sys_updated_on');
  </g2:evaluate>
  <g:requires name="<UI SCRIPT NAME>.jsdbx" params="cache=$[jvar_stamp]" />
</j:jelly>
```

Call a UI script in HTML

To run a UI script from HTML code, use the \texttt{<script>} tag and specify the \texttt{src=} argument as the API name of the UI script followed by the .jsdbx extension. For example, include the UI script named CoolClock with this code:

```html
<script language="javascript" src="CoolClock.jsdbx" />
```

Call a UI script from client-side code

Access UI scripts from within client-side code using the \texttt{g\_ui\_scripts} global object. For more information, see GlideUIScripts - Client.

\textbf{Note:} This class does not support UI scripts with the \texttt{Global} field set to true.

Catalog client scripts

Client-side scripts can add dynamic effects and validation to forms. Scripts can apply to service catalog items or variable sets, allowing administrators to use the same functionality that is available on other forms.

You can use client side scripts to:

- Get or set variable values.
- Hide or display variables.
• Make variables mandatory or not.
• Validate form submission.
• Add something to the cart.
• Order something immediately.

**Catalog client script considerations**
When you create catalog client scripts, be aware of the following considerations.

• Catalog client scripts run when a user orders an item from the service catalog. Catalog client scripts can also run when variables or variable sets for a catalog item are displayed when a user requests that item.
• For a variable to be accessible using a catalog client script, it must have a variable name. Variables without names do not appear in the list of available variables.
• When using standard client scripts on a Requested Item or Catalog Task form, make a note of fields with the same name as variables. If a table field and a variable of the same name are both present on a form, the table field is matched when it is accessed using a script. If this happens, specifically address the variable by naming it `variables.variable name`. For example:
  ```javascript
g_form.setValue('variables.replacement', 'false');
  ```
• If you are using record producers to pass variables from the service catalog to other types of records, these variables are made visible in those records with a variable editor, such as the Change Variable Editor UI formatter on Change request forms. You can manipulate these variables using standard client script methods, such as `setDisplay`, `setMandatory`, `setValue`, and `getValue`.
• Catalog client scripts can be used for catalog items included in a wizard.
• You can use the `g_form.refreshSlushbucket(fieldName)` API to update a list collector variable.

**Catalog client script differences**
Catalog client scripts are very similar to standard client scripts, with a few important differences.
Instead of selecting a table such as Incident for the script, select a catalog item or variable set. As your system may have a large number of catalog items, you should select a catalog item or variable set using a reference field instead of the choice list that the standard Client Script form uses.

When using an `onChange()` catalog client script, it is linked to a particular variable instead of a field. The system automatically populates the variable name selection list with any named variables from the catalog item or variable set selected.

Create a catalog client script

Follow this procedure to create a catalog client script.

**Procedure**

1. Navigate to Service Catalog > Catalog Administration > Catalog Client Scripts.
   A list of current custom catalog client scripts appears.
2. Click New.
3. Fill in the fields, as appropriate (see table).

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique name for the catalog client script.</td>
</tr>
<tr>
<td>Applies to</td>
<td>Select the item type this client script applies to:</td>
</tr>
<tr>
<td></td>
<td>• A Catalog Item: enables the Catalog item field.</td>
</tr>
<tr>
<td></td>
<td>• A Variable Set: enables the Variable set field.</td>
</tr>
<tr>
<td>Active</td>
<td>Select the check box to enable the client script. Clear the check box to</td>
</tr>
<tr>
<td></td>
<td>disable the script.</td>
</tr>
<tr>
<td>UI Type</td>
<td>Whether to apply this to desktop, mobile, or both.</td>
</tr>
<tr>
<td>Script</td>
<td>Enter the client script that should run on the service catalog item.</td>
</tr>
<tr>
<td>Type</td>
<td>Select when the script should run, such as <code>onLoad</code> or <code>onSubmit</code>.</td>
</tr>
<tr>
<td>Catalog item or</td>
<td>Select a catalog item or variable set from the list. The field name and</td>
</tr>
<tr>
<td>Variable set</td>
<td>options available depend on the selection in the Applies to field.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Applies on a Catalog Item view</td>
<td>Select the check box to apply the catalog client script to catalog items displayed within the order screen on the service catalog. Available in the requester view.</td>
</tr>
<tr>
<td>Applies on Requested Items</td>
<td>Select the check box to apply the catalog client script on a Requested Item form, after the item is requested. Available in the fulfiller view. See VEditor.</td>
</tr>
<tr>
<td>Applies on Catalog Tasks</td>
<td>Select the check box to apply the catalog client script when a Catalog Task form for the item is being displayed. Available in the fulfiller view. See VEditor.</td>
</tr>
<tr>
<td>Applies on the Target Record</td>
<td>Select the check box to support the catalog UI policy on a record created for task-extended tables via record producers. See Default variable editor.</td>
</tr>
</tbody>
</table>

4. Click Submit.

**Catalog client script examples**

Examples of client scripts to perform common actions.

**Example: Get the value of a variable**

Use the following syntax to obtain the value of a catalog variable. Note that the variable must have a name. Replace `variable_name` with the name of the variable.

```javascript
g_form.getValue('variable_name');
```

**Example: Restrict the number of characters a user can type in a variable**

This is an example of a script that runs when the variable is displayed, rather than when the item is ordered.

```javascript
function onLoad(){
   var sd = g_form.getControl('short_description');
   sd.maxLength=80;
}
```

**Mobile client GlideForm (g form) scripting and migration**

Client scripting for mobile is identical to scripting for the web, with some exceptions. All new scripts must conform to certain guidelines. The following
items are affected on the mobile platform: client scripts, UI policies, navigator modules, and UI actions.

**Client scripts**
For new or existing scripts to be valid for mobile, they must conform to the following requirements:

- Use the new mobile methods in place of `g_form.getControl()`.
- Do not use deprecated methods.
- Do not reference unsupported browser objects.
- Do not make synchronous JavaScript, GlideAjax, and GlideRecord calls.
- Do not call methods that are not available for mobile.
- Enable scripts to run on the mobile UI.

---

**Requirements**

| Use the new mobile methods | Several new methods are available for modifying form fields instead of directly manipulating the HTML. These methods replace previous usages of `g_form.getControl()`, which is deprecated for the mobile platform. In your existing scripts, ensure that the new methods are used in place of methods that are not valid on the mobile platform. For information on these new methods, refer to Mobile GlideForm() API. |
| Do not use deprecated methods | The following methods have been deprecated for the mobile platform because direct access to HTML elements is not allowed: |
| Do not reference unsupported browser objects | The following browser objects are not supported in mobile scripts: |

- `g_form.getControl()`
- `g_form.getFormElement()`
- `g_form.getElement()`

To ensure that existing scripts are compatible, remove all calls to deprecated methods from your code. For new scripts, do not use deprecated methods if you want the script to be valid for mobile.

For `g_form.getControl()`, some of the functionality previously included with this method has been extracted to individual methods. Instead of `g_form.getControl()` use the new methods described on the developer site.

- Window
- jQuery or Prototype ($, $j, or $$)
- Document
Requirements (continued)

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Make sure that new scripts do not use these objects, and remove any usage of these objects from your existing scripts. Use GlideForm (g_form) instead, which provides methods such as setLabel(), addDecoration(), and hasField() for accomplishing the same tasks.</td>
<td></td>
</tr>
<tr>
<td>Do not make synchronous JavaScript calls</td>
<td>The mobile platform does not allow synchronous JavaScript calls. The <code>g_form.getReference()</code> method must now have the callback parameter defined. For example: <code>g_form.getReference(fieldName, callback)</code></td>
</tr>
<tr>
<td></td>
<td>Be sure that all <code>g_form.getReference()</code> calls include the callback parameter. For example, the following script does not include a callback and is incompatible with the mobile platform:</td>
</tr>
</tbody>
</table>
|                                                   | ```javascript
var userName = g_form.getReference('assigned_to').user_name;
g_form.setValue('u_assigned_user_name', userName);
```
|                                                   | The following script has been updated to include the callback and is compatible with the mobile platform:                                                                                                                     |
|                                                   | ```javascript
function(now_GR) {
    g_form.setValue('u_assigned_user_name', gr.user_name);
};
g_form.getReference('assigned_to', function(now_GR) {
    g_form.setValue('u_assigned_user_name', now_GR.user_name);
});
```
| Do not make synchronous GlideAjax calls           | The mobile platform does not allow synchronous GlideAjax calls. Any use of `getXMLWait()` in a GlideAjax call will not work on the mobile platform. Be sure that all GlideAjax calls are asynchronous. For more on synchronous versus asynchronous GlideAjax calls and `getXMLWait()`, see AJAX. For information on the available GlideAjax methods, refer to the GlideAjax API. |
| Do not make synchronous GlideRecord calls         | The mobile platform does not allow synchronous GlideRecord calls. Make sure that any existing GlideRecord calls include a callback. For example, the following script does not include a callback and is incompatible with the mobile platform: |
|                                                   | ```javascript
var now_GR = new GlideRecord('incident');
gr.addQuery('number', g_form.getValue('related_incident'));
gr.query();
gr.next();
g_form.setValue('u_related_incident_description', gr.short_description);
```
|                                                   | The following script has been updated to include the callback, and is compatible with the mobile platform:                                                                                                                     |
|                                                   | ```javascript
var now_GR = new GlideRecord('incident');
gr.addQuery('number', g_form.getValue('related_incident'));
```
Do not use methods unavailable on the mobile platform

<table>
<thead>
<tr>
<th>Do not use methods unavailable on the mobile platform</th>
<th>Due to the limitations and reduced functionality that is imposed by the mobile platform, the following methods are not deprecated but are not available on the mobile platform. If these run on the mobile platform, no action occurs:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• showRelatedList ()</td>
<td>• showRelatedList ()</td>
</tr>
<tr>
<td>• hideRelatedList ()</td>
<td>• hideRelatedList ()</td>
</tr>
<tr>
<td>• showRelatedLists ()</td>
<td>• flash()</td>
</tr>
<tr>
<td>• hideRelatedLists()</td>
<td>• getSections()</td>
</tr>
<tr>
<td>• flash()</td>
<td>• enableAttachments()</td>
</tr>
<tr>
<td>• getSections()</td>
<td>• disableAttachments()</td>
</tr>
<tr>
<td>• enableAttachments()</td>
<td>• setReadonly() (Note that setReadOnly() is available)</td>
</tr>
<tr>
<td>• disableAttachments()</td>
<td>• getParameter()</td>
</tr>
</tbody>
</table>

Enable scripts for mobile

Scripts must be enabled for the mobile platform. See Enable client scripts for the mobile browser for the ServiceNow Classic mobile app.

Note: Focusing an element on a mobile form is not supported.

UI policies

Use the Run scripts in UI type field to determine whether scripts run on the mobile platform, the desktop, or both. Update existing policies so that they apply to either the mobile platform or both. For new scripts, also ensure that the mobile option or both is selected. For more on UI policies for mobile, see Enable UI policies for the mobile browser.

Navigator modules

For existing code, modules must be transferred to either the sys_ui_application or sys_ui_module tables to be available on the mobile platform. When developing new code, be sure that all modules are created in the sys_ui_application or...
sys_ui_module tables. For more information, see Enable an application menu for the ServiceNow Classic mobile app.

**UI actions**

UI actions must be transferred to the sys_ui_ng_action table to appear on the mobile platform. UI action scripts that do not use deprecated methods do not require changes to the script itself. For new UI actions, be sure that they are created in the sys_ui_ng_action table. For more information, see Mobile UI actions for the ServiceNow Classic mobile app.

**AJAX**

AJAX (asynchronous JavaScript and XML) is a group of interrelated, client-side development techniques used to create asynchronous Web applications. AJAX enables web applications to send and retrieve information to and from a server in the background, without impacting the user experience with the displayed web page.

**GlideAjax**

The GlideAjax class allows the execution of server-side code from the client. GlideAjax calls pass parameters to the script includes, and, using naming conventions, allows the use of these parameters.

!? Note: This functionality requires a knowledge of JavaScript.

Using GlideAjax:

- Initialize GlideAjax with the name of the script include that you want to use.
- When creating the script include, you must set the name field to be exactly the same as the class name.
- When creating the script include, you must select the Client callable check box.
- Specify the parameter sysparm_name. GlideAjax uses sysparm_name to find which function to use.
- Any extra parameters may be passed in, all of which must begin with sysparm_. Avoid using predefined parameter names:
  - sysparm_name
  - sysparm_function
  - sysparm_value
  - sysparm_type
- Code is then executed with the getXML() or getXMLWait() functions.
For additional information on GlideAjax, refer to GlideAjax in the development portal.

**Examples of asynchronous GlideAjax**

There are two parts to the asynchronous GlideAjax script: client-side and server-side code.

**Hello World: Returning a value from the server**

**Example: Client side**

This code runs on the client (the web browser). Create a client script as normal. This sends the parameters to server, which then does the processing. So that the client does not wait for the result, a callback function is used to return the result, passed to the `getXML()` function. (In this case it is called `HelloWorldParse`.)

The `getXMLWait()` function does not need a separate callback function, but this will block the client. If the client-server communication takes a long time (for example on slow networks), the application will seem unresponsive and slow. An example of `getXMLWait()` is in the following section.

```javascript
var ga = new GlideAjax('HelloWorld');
ga.addParam('sysparm_name', 'helloWorld');
ga.addParam('sysparm_user_name', "Bob");
ga.getXML(HelloWorldParse);

function HelloWorldParse(response) {
    var answer = response.responseXML.documentElement.getAttribute("answer");
    alert(answer);
}
```

**Example: Server side**

The server-side code for the above function. Do not create a business rule, but instead navigate to **System Definition > Script Include** and create a new script. Paste in the code below.

ℹ️ **Note:** You must set the name of the script include to `HelloWorld`.

- The `sys_script_include` code must extend the `AbstractAjaxProcessor` class and be client-callable.
- Function names starting with "_" are considered private and are not callable from the client.
- Avoid overriding methods of `AbstractAjaxProcessor`, including `initialize`. While it is possible to invoke methods of your superclass object which you have overridden, it is complicated and best avoided altogether.
var HelloWorld = Class.create();
HelloWorld.prototype = Object.extendsObject(AbstractAjaxProcessor, {
    helloWorld:function() { return "Hello " + this.getParameter('sysparm_user_name') + "!"; },
    _privateFunction: function() { // this function is not client callable
    }
});

This results in an alert box that says 'Hello Bob!' when you visit the form.

**Returning multiple values**

Since the response is an XML document we are not limited to returning a single answer value. Here is a more complex example returning multiple XML nodes and attributes.

**Example: AJAX processor script include**

```javascript
/*
 * MyFavoritesAjax script include Description - sample AJAX processor returning multiple value pairs
 */
var MyFavoritesAjax = Class.create();
MyFavoritesAjax.prototype = Object.extendsObject(AbstractAjaxProcessor, {

    /*
     * method available to client scripts call using:
     * var gajax = new GlideAjax("MyFavoritesAjax");
     * gajax.addParam("sysparm_name","getFavorites");
     */
    getFavorites: function() { // build new response xml element for result
        var result = this.newItem("result");
        result.setAttribute("message","returning all favorites");

        //add some favorite nodes with name and value attributes
        this._addFavorite("color","blue");
        this._addFavorite("beer","lager");
        this._addFavorite("pet","dog");
    },
    // all items are returned to the client through the inherited methods of AbstractAjaxProcessor
    _addFavorite: function(name, value) {
        var favs = this.newItem("favorite");
        favs.setAttribute("name",name);
        favs.setAttribute("value",value);
    };
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
/* MyFavoritesAjax */

```javascript
Example: Client script

// new GlideAjax object referencing name of AJAX script include
var ga = new GlideAjax("MyFavoritesAjax");
// add name parameter to define which function we want to call
// method name in script include will be getFavorites
ga.addParam("sysparm_name","getFavorites");

// submit request to server, call ajaxResponse function with server response

function ajaxResponse(serverResponse) {
  // get result element and attributes
  var result = serverResponse.responseXML.getElementsByTagName("result");
  var message = result[0].getAttribute("message");

  // check for message attribute and alert user
  if(message) alert(message);

  // build output to display on client for testing
  var output = ";

  // get favorite elements
  var favorites = serverResponse.responseXML.getElementsByTagName("favorite");
  for(var i = 0; i < favorites.length; i++) {
    var name = favorites[i].getAttribute("name");
    var value = favorites[i].getAttribute("value");
    output += name + " = " + value + "\n";
  }

  alert(output); }
```

Example: XML response

```xml
<xml sysparm_max= "15" sysparm_name= "getFavorites" sysparm_processor= "MyFavoritesAjax">  
  <result message= "returning all favorites"></result>  
  <favorite name= "color" value= "blue"></favorite>  
  <favorite name= "beer" value= "lager"></favorite>
</xml>
```
Examples of synchronous GlideAjax

Use synchronous when your script cannot continue without the GlideAjax response. This stops the session until the response is received.

If your use case demands that no further processing can occur until the GlideAjax response has been received, you can use `getXMLWait()`. However, because this will slow down your code and lock the user session until the response is received, it is generally recommended that you use `getXML()` with a callback function.

⚠️ Note: Do not use `AJAXEvaluateSynchronously`.

⚠️ Note: The `getXMLWait()` method is not available in scoped applications.

This code results in a client-side alert that displays The Server Says Hello Bob!.

The client code.

```javascript
var ga = new GlideAjax('HelloWorld') ;
ga.addParam('sysparm_name','helloWorld');
ga.addParam('sysparm_user_name','Bob');
ga.getXMLWait();
alert(ga.getAnswer());
```

The server-side script include code.

```javascript
var HelloWorld = Class.create();
HelloWorld.prototype = Object.extendsObject(AbstractAjaxProcessor, { helloWorld: function() { return "The Server Says Hello " + this.getParameter('sysparm_user_name') + "!"; } });
```

AJAXClientHelper

Provides helper functions for Ajax clients to generate Choice Lists and retrieve displayed values from a choice list.

Where to use

Use this script include wherever your need to generate choice lists or retrieve a value from an Ajax client.
**Method summary**

<table>
<thead>
<tr>
<th>Method summary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getValues()</code></td>
<td>Gets the values from the choice list.</td>
</tr>
<tr>
<td><code>generateChoice()</code></td>
<td>Generates the choices for a choice list.</td>
</tr>
<tr>
<td><code>generateChoiceTable()</code></td>
<td>Generates the choice table.</td>
</tr>
<tr>
<td><code>getDisplay()</code></td>
<td>Gets the display value from the choice list.</td>
</tr>
</tbody>
</table>

**Method detail**

<table>
<thead>
<tr>
<th>API method</th>
<th>Description</th>
<th>Input parameters</th>
<th>Output returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getValues()</code></td>
<td>Gets the values from the choice list.</td>
<td>none</td>
<td>The choice list values.</td>
</tr>
<tr>
<td><code>generateChoice()</code></td>
<td>Generates the choices for a choice list.</td>
<td>none</td>
<td>The choice list.</td>
</tr>
<tr>
<td><code>generateChoiceTable()</code></td>
<td>Generates the choice table.</td>
<td>none</td>
<td>The choice table.</td>
</tr>
<tr>
<td><code>getDisplay()</code></td>
<td>Gets the display value from the choice list.</td>
<td>none</td>
<td>The display value.</td>
</tr>
</tbody>
</table>

**AJAXClientTiming**

Saves client timing values for a transaction.

**Method summary**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>process()</code></td>
<td>Method called by the Prototype JavaScript Framework during object processing.</td>
</tr>
</tbody>
</table>
Method detail

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Input Fields</th>
<th>Output Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>process()</td>
<td>Called by the Prototype JavaScript Framework during object processing. Do not call this method directly.</td>
<td>none</td>
<td>Returns: void.</td>
</tr>
</tbody>
</table>

Jelly tags

Use Jelly to turn XML into HTML.

Watch these introductory to learn about using Jelly in the Now Platform.

- Introducing Jelly Scripting - Part 1 (Video)
- Introducing Jelly Scripting - Part 2 (Video)
- Introducing Jelly Scripting - Part 3 (Video)

Jelly Tags

if

- **Description**: The `if` tag is just what it looks like, an if tag. This is like an if statement in any programming language, but keep in mind that there is no elseif tag and no else tag. If you want to create that kind of structure, try the choose/when/otherwise syntax.

- **Parameters**: `test` - The condition to evaluate in order to determine if the block will execute.

- **Example**:

```xml
<evaluate var="jvar_gr" object="true">
  var now_GR = new GlideRecord("incident");
  gr.addQuery("active", true);
  gr.query();
  gr;
</evaluate>

<if test="!jvar_gr.hasNext()">
  We did not find any active incidents.
</if>
<if test="jvar_gr.next()">
  We found ${jvar_gr.getRowCount()} active incidents.
</if>
```

© 2021 ServiceNow, Inc. All rights reserved.
while

- **Description**: The *while* tag does a while loop.
- **Parameters**:
  - **test** - The condition to evaluate in order to determine if the statement will loop through. This should be an expression enclosed in ${} or $[] that evaluates to true or false.
- **Example**:

  ```
  <g:evaluate var="jvar_gr" object="true">
    var now_GR = new GlideRecord("incident");
    gr.addQuery("active", true);
    gr.query();
    gr;
  </g:evaluate>

  <j:while test="${jvar_gr.next()}">
    <a href="incident.do?sys_id=${jvar_gr.getValue('sys_id')}">${jvar_gr.getValue('number')}</a>
  </j:while>
  ```

set

- **Description**: The *set* tag sets a variable.
- **Parameters**:
  - **var** - The variable to set. Often the system prefixes these variables with jvar_ for consistency.
  - **value** - The value to set var to. This is often an expression enclosed in ${} or $[].
  - **defaultValue** - If the value results to null or empty, this value is put into the var.
- **Example**:

  ```
  <j:set var="jvar_incident_number" value="${jvar_gr.getValue('number')}"/>
  ```

set_if
• **Description:** The `set_if` tag sets a variable based on a test. This tag is similar to the ternary operator in other programming languages (`var = <test> ? <if_true> : <if_false>`).

• **Parameters:**
  - `var` - The variable to set. Often the system prefixes these variables with `jvar_` for consistency.
  - `test` - The condition to evaluate in order to determine if the statement will evaluate the true value or the false value. This should be an expression enclosed in `${}` or `[]` that evaluates to true or false.
  - `true` - The value to set the variable to if `test` evaluates to `true`. This parameter is optional, so if the field is blank, and if `test` evaluates to `true`, the variable will be left blank.
  - `false` - The value to set the variable to if `test` evaluates to `false`. This parameter is optional, so if the field is blank, and if `test` evaluates to false, the variable will be left blank.

choose

• **Description:** The `choose` tag starts a choose block of code. This is similar to the if-elseif-else kind of syntax in most programming languages. With a `choose` tag, you can use `when` and `otherwise` tags to specify other blocks of code.

• **Parameters:** None.

• **Example:**

```<j:choose>
  <j:when test="${jvar_gr.getRowCount() AMP&lt; 1}">We found multiple records!</j:when>
  <j:when test="${jvar_gr.next()}">We found record ${jvar_gr.getValue('number')} </j:when>
  <j:otherwise>Sorry, we could not find the record you specified.</j:otherwise>
</j:choose>```

when

• **Description:** The `when` tag is used within a choose block to indicate a condition. This tag is similar to an if or an elseif in that it specifies a condition, executes the inner content, and then implies a break at the end to leave the if-elseif construct.
• **Parameters**: test - The condition to evaluate in order to determine if the statement will loop through. This should be an expression enclosed in ${[] or $[]} that evaluates to true or false.

• **Example**:

```xml
<j:choose>
  <j:when test="${jvar_gr.getRowCount() lt; 1}"">We found multiple records!</j:when>
  <j:when test="${jvar_gr.next()}"">We found record ${jvar_gr.getValue('number')}"</j:when>
  <j:otherwise>Sorry, we could not find the record you specified.</j:otherwise>
</j:choose>
```

`otherwise`

• **Description**: The `otherwise` tag is used within a `choose/when/otherwise` block, and is like the "else" or "default" case.

• **Parameters**: None.

• **Example**:

```xml
<j:choose>
  <j:when test="${jvar_gr.getRowCount() lt; 1}"">We found multiple records!</j:when>
  <j:when test="${jvar_gr.next()}"">We found record ${jvar_gr.getValue('number')}"</j:when>
  <j:otherwise>Sorry, we could not find the record you specified.</j:otherwise>
</j:choose>
```

**Glide Tags**

**evaluate**

• **Description**: The `evaluate` tag evaluates JavaScript code (server side), and makes variables visible to future code. Unlike other tags, the `evaluate` tag evaluates the content that is inside the tag as server side JavaScript.

The context is the same as that of script includes in the system. Other script includes, global business rules, GlideRecord, GlideSystem, and Jelly variables (prefixed with jelly. if the parameter jelly="true" is set) are available.

• **Parameters**: 

© 2021 ServiceNow, Inc. All rights reserved.
• **var** - The name of the variable that will be set to the result of the script.

• **object** - If set to **true**, the result of the expression will be treated as an object instead of a primitive variable (string or integer variable values).

• **jelly** - If set to **true**, allows Jelly context variables to be referenced in the script.

• **expression** - This is an expression to be executed for the value to put in **var**. The expression can be either of two places. First, it can be an attribute on the evaluate tag itself. Otherwise, the content between the beginning tag and ending tag is the expression. The last line of the expression is the actual value passed into **var**.

**Example:**

```xml
<g:evaluate var="jvar_gr" object="true">
    var now_GR = new GlideRecord("incident");
    gr.addQuery("active", "true");
    gr.query();
    gr; // this is the variable put into the variable jvar_gr
</g:evaluate>
```

```xml
<g:evaluate var="jvar_gr" object="true" expression=""
    var now_GR = new GlideRecord('incident');
    gr.addQuery('active', 'true');
    gr.query();
    gr; // this is the variable put into the variable jvar_gr" />
```

**messages**

• **Description**: The **messages** tag helps with translation. When `gs.getMessage()` is called anywhere on a page, there are two possible places where the translation is found. First, the page checks a local cache of translations. Second, the page makes an AJAX call to the server to find the translation. What `g:messages` does is allow pages to cache certain messages.

• **Parameters**: None.

• **Example**:

```xml
<g:messages>
    Yes
    No
</g:messages>
```
breakpoint

• **Description:** When the breakpoint tag is called, it prints a list of all the variables in Jelly at the current moment, with their respective values. If a variable is specified, it prints the requested variable and its value. The output is placed in the System Log.

• **Parameters:** `var` - (Optional) The variable to log the value for. If `var` is not specified, then all variables will be dumped into the log.

• **Example:**

  ```xml
  <g:breakpoint />
  <g:breakpoint var="sysparm_view"/>
  ```

no_escape

• **Description:** The system, by default, uses escaped output as a security measure. Output placed inside of no_escape tags is not escaped before output. Be careful when using these tags, because if user input is displayed here it can open a security vulnerability on the page.

• **Parameters:** None.

• **Example:**

  ```xml
  <g:no_escape>
  ${jvar_raw_html_data}
  </g:no_escape>
  ```

macro_invoke

• **Description:** The macro_invoke tag calls a UI macro that you have specified in the database. You may also call a UI macro by specifying it in the tag name. For example, if you had a UI macro named my_macro, you could call that macro with the tag `<g:my_macro/>`.

• **Parameters:**
- **macro** - The name of the UI macro to execute. If your tag name is `g:macro_invoke`, then the macro attribute specifies the name of the macro. If the tag name includes the name of the macro, then there is no need to include a macro attribute.

- **Other attributes** - For each attribute you specify, a variable with that name will be available in the context of the UI macro, prefixed with "jvar_".

**Example:**

```xml
<!-- Will invoke the contents of the UI macro named "sample_macro", which will have the variable jvar_message available within it-->
<g:macro_invoke macro="sample_macro" message="This is a sample macro variable." />

<!-- Will invoke the contents of the UI macro named "sample_macro", which will have the variable jvar_message available within it-->
<g:sample_macro message="This is a sample macro variable." />
```

---

### Jelly escaping types

You use different methods when escaping characters in JavaScript and HTML. JavaScript uses the backslash character, and HTML uses the ampersand character.

**Note:** This functionality requires a knowledge of JavaScript, HTML, and Apache Jelly (a Java and XML based scripting and processing engine for turning XML into executable code).

There are two different types of escaping that is required when generating output from Jelly:

- **JavaScript**
- **HTML**

The escaping for each of these consists of:

<table>
<thead>
<tr>
<th>Type</th>
<th>From</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>JavaScript</strong></td>
<td>' (single quote)</td>
<td>'</td>
</tr>
<tr>
<td></td>
<td>&quot; (double quote)</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>CR (carriage return)</td>
<td>(blank)</td>
</tr>
<tr>
<td></td>
<td>NL (newline)</td>
<td>\n ('' followed by 'n')</td>
</tr>
<tr>
<td>Type</td>
<td>From</td>
<td>To</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------</td>
<td>----------</td>
</tr>
<tr>
<td>HTML</td>
<td>&amp; (ampersand)</td>
<td>&amp;</td>
</tr>
<tr>
<td></td>
<td>&lt; (less than)</td>
<td>&lt;</td>
</tr>
<tr>
<td></td>
<td>&gt; (greater than)</td>
<td>&gt;</td>
</tr>
</tbody>
</table>

You can also escape HTML using the `getHTMLValue()` function which will enforce all line breaks and escape the characters mentioned above. It can be used as follows:

```java
$(test.getHTMLValue())
```

### Add escaping to a Jelly replacement

You can handle character escaping in Jelly files. XML escaping behavior can be modified only by users with the security_admin role.

### About this task

**Note:** This functionality requires a knowledge of JavaScript, HTML, and Apache Jelly (a Java and XML based scripting and processing engine for turning XML into executable code).

### Procedure

Add a prefix to the `${expression}` or `[$expression]` indicating the escaping to be performed.

```java
${JS:expression}
${HTML:expression}
```

The prefix tells the system to take the result of the expression and escape it before outputting. The escaping may be combined by specifying a comma-separated list of prefixes:

```java
${JS,HTML:expression}
```

### Extensions to Jelly syntax

Apache's Jelly syntax is used to render forms, lists, UI pages, and many other things rendered in ServiceNow.

With Jelly, logic can be embedded within static content and computed values may be inserted into the static content.

**Attention:** This functionality requires a knowledge of Apache Jelly (a Java and XML based scripting and processing engine for turning XML into executable code).
This page from Apache has a summary of the standard Jelly tags: http://commons.apache.org/jelly/tags.html

**Namespaces**

Jelly often includes multiple namespaces when invoking tags.

The "j" namespaces are standard Jelly whereas the "g" namespaces are unique to ServiceNow scripts. For example, the <g:evaluate> tag is supplied by ServiceNow to allow you to compute a value using JavaScript. The standard Jelly tag <j:test> is used to evaluate a condition.

**Phases**

Usually, there are two phases indicated by namespaces <j> versus <j2> and <g> versus <g2>.

The namespaces without the "2" happen in the first phase of processing and these are cached except when used in a UI page. Those with the "2" are never cached. Care must be taken when selecting whether to use phase 1 or phase 2 for efficiency and correct results.

In addition to the namespaces, the syntax used to insert values into static content differs depending on which phase is to supply the value. A dollar with braces surrounding a value inserts the value in phase 1. For example, ${jvar_ref} inserts the value jvar_ref during phase 1 of the jelly process. A dollar with brackets surrounding a value inserts the value in phase 2. For example, ${[jvar_ref]} inserts the value jvar_ref during phase 2. A value surrounded by quotes is treated as a string. For example, ['jvar_ref'] inserts the value jvar_ref as a string during phase 2.

```html
<script>
if (confirm("${gs.getMessage('home.delete.confirm')}"))
...
</script>
<input type="hidden" id="${jvar_name}" name="${jvar_name}" value="${jvar_value}" class="${jvar_class}" />
```

**If tests**

You can use if statements in Jelly scripts.

Testing whether something is true or not can be done as follows:

```xml
<j:if test="${jvar_something}">...do something...</j:if>
<j:if test="${!jvar_something}">...do something...</j:if>
```
The reason this statement works, is that, in Jelly, a term like jvar_something is "truthful" in an if tag if:

1. it is Boolean and true
2. it is a String and = "true", "yes", "on", or "1"

Testing whether something exists can be done as follows:

```xml
<j:if test="${empty(jvar_something)}">...do something...</j:if>
```

The reason this statement works is that the JEXL empty function returns true if its argument is:

1. null
2. an empty string
3. a zero length collection
4. a map with no keys
5. an empty array

⚠️ Note: You cannot mix javascript and jvar variables in a JEXL expression. They must be broken into separate expressions.

**Set If**

Sets a variable to one of two different values depending on whether a test is true or false.

```xml
<g2:set_if var="jvar_style" test="${gs.getPreference('table.compact') != 'false'}" true="margin-top:0px; margin-bottom:0px;" false="margin-top:2px; margin-bottom:2px;" />
```

**<g:insert> versus <g:inline> versus <g:call>**

This page provides a comparative explanation of three tags: `<g:insert>`, `<g:inline>`, and `<g:call>`.

**<g:insert>**

The `<g:insert>` tag inserts a Jelly file into your Jelly in a new context. This means you cannot access the variables previously established in your Jelly.

```xml
<g:insert template="get_target_form_function.xml" />
```
The `<g:inline>` tag inserts a Jelly file into your Jelly in the same context. This means that the inserted Jelly can access the variables you previously established and it can change the values of those variables.

```
<g:inline template="element_default.xml" />
```

The `<g:call>` tag may be used. Your function will only have access to the values passed to it. The Jelly context will look the same after a call as before the call. This means you cannot set a global variable here and read it later. This also means you can't mistakenly set a global variable called "jvar_temp" and overwrite a variable that somebody else was relying on.

Passing values, if needed, is done explicitly by including the name of the parameter on the `<g:call>` line followed by the equal sign followed by the value in quotes:

```
<g:call function="collapsing_image.xml" id="${jvar_section_id}" image="${jvar_cimg}"
    first_section_id="${jvar_first_section_id}" image_alt="${jvar_cimg_alt}" />
```

If values are passed, and you want to have defaults or required parameters, your Jelly referenced in the function must then include a line to declare whether the parameters are required or have a default value:

```
<g:function id="REQUIRED" image="REQUIRED" image_prefix="" image_alt="REQUIRED"/>
```

The example above indicates that 3 of the parameter are required and one parameter is option with a blank default value. Note that if you are not passing values or if you do want to have default or required values, you do not need to include the `<g:function>` line at all. In general, however, you will want to include a `<g:function>` line.

The value can then be referenced in your template by prepending the "jvar_" prefix to the parameter's name:

```
<img id="img.${jvar_id}" src="images/${jvar_image}" alt="${jvar_image_alt}"
    onclick="toggleSectionDisplay('${jvar_id}', '${jvar_image_prefix}', '${jvar_first_section_id}');"/>
```

For `<g:call>`, parameters may also be pass implicitly as a list of named variables in an "arguments" parameter:

```
<g:call function="item_link_default.xml"
    arguments="sysparm_view,ref_parent,jvar_target_text"/>
```

As an alternative to passing variables into the function via separate tag arguments, it is possible to pass a list of variables in a single 'arguments'
argument. All variables identified by name (comma separated) in the argument parameter are re-introduced within the function under the exact same name (e.g. inside the function template, we'd have variables sysparm_view, ref_parent, and jvar_target_text available to us).

The function template may return a value to the calling template using the return attribute. Within the function the jvar_answer variable sets the return value.

```xml
g:call function="item_body_cell_calc_style.xml" arguments="jvar_type" return="jvar_style"/>
```

**<g:evaluate>**

The <g:evaluate> tag is used to evaluate an expression written in Rhino JavaScript and sometimes to set a variable to the value of the expression.

The last statement in the expression is the value the variable will contain.

```xml
g2:evaluate var="jvar_page" jelly="true">
  var page = "";
  var pageTitle = "";
  var pageGR = new GlideRecord("cmn_schedule_page");
  pageGR.addQuery("type", jelly.jvar_type");
  pageGR.query();
  if (pageGR.next()) {
    page = pageGR.getValue("sys_id");
    pageTitle = pageGR.getDisplayValue();
  } 
  page;
</g2:evaluate>
```

```xml
<g2:evaluate var="not_important" expression="sc_req_item.popCurrent()" />
```

**object="true"**

If you would like to have the evaluate return an object (for example an array), use the argument object="true".

```xml
<g2:evaluate object="true" var="jvar_items" expression="SncRelationships.getCMDBViews()" />
```

**jelly="true"**

If you would like to access Jelly variables inside an evaluate, include jelly="true" in the evaluate and add "jelly." before the Jelly variable's name. For example, to access the GlideJellyContext:
Another example of accessing a jvar using the jelly="true" parameter. The value of jvar_h was set previously and we can access it inside the evaluate:

```
${NLBR:jvar_h.getHTMLValue('newvalue')}
```

```g2:evaluate var="jvar_fix_escaping" jelly="true">
  var auditValue = jelly.jvar_h.getHTMLValue('newvalue');
  gs.log("************ " + auditValue);
</g2:evaluate>
```

**copyToPhase2="true"**

If you have a need to take the results of an evaluation that occurs in phase 1 and propagate it to phase 2, use copyToPhase2="true". There is some protection for escaping in this use. For example:

```g:evaluate var="jvar_has_special_inc" copyToPhase2="true">
  var specialInc = gs.tableExists("special_incident");
  specialInc;
</g:evaluate>
```

```
${jvar_has_special_inc}
```

If you do not need to evaluate something, you can do this more directly. Beware of escaping issues here (double quotes in jvar_rows would cause a problem in the example):

```j2:set var="jvar_rows" value="${jvar_rows}"/>
```

```
<g:breakpoint/>
```

This tag can be used to display the current Jelly variables and their values in the log.

Be sure to remove this tag before going to production.

```
<g:ui_form/>
```

This tag defines a form on the UI page.

For example, if your form contained the application_sys_id field:

```
<g:ui_form>
  <p>Click OK to run the processing script.</p>
</g:ui_form>
```
The g:ui_form may benefit greatly from a processing script.

**<g:ui_input_field />**
This tag adds a reference to a UI macro that creates an input field on a page that allows users to input information. The ui_input_field passes a label, name, value, and size into the UI macro.

Here is an example from a UI page:

```html
<g:ui_input_field label="sys_id" name="syid" value="9d385017c611228701d22104cc95c371" size="50"/>
```

**<g:ui_checkbox/>**
This tag puts a user-editable check mark on a page. The name and value are passed into the UI macro.

Here is an example from a table on a UI page:

```html
<table>
  <tr>
    <td nowrap="true">
      <label>Time Card Active:</label>
    </td>
    <td>
      <g:ui_checkbox name="timecard_active" value="${sysparm_timecard_active}"/>
    </td>
  </tr>
</table>
```

**<g:dialog_buttons_ok_cancel/>**
This tag puts buttons on the UI page that run a specified processing script if the tag returns true.

If your UI page contains a form (uses the <g:form> tag), you can submit the form and have the Processing Script run. The Processing Script can naturally access fields on the form. For example, if your form contained the application_sys_id field:

```html
<g:ui_form>
  <p>Click OK to run the processing script.</p>
</g:ui_form>
```
This tag adds a reference to a page that can be referenced by a Processing Script.

The following example creates a reference defined by name, id, and table parameters in the tag:

```xml
<g:ui_reference name="QUERY:active=true^roles=itil" id="assigned_to" table="sys_user" />
```

Then in the Processing Script, reference the name field like this:

```javascript
newTask.assigned_to = request.getParameter("QUERY:active=true^roles=itil");
```

You can specify a reference qualifier, so that the "name" attribute can be unique. The following example creates a reference defined by name, id, and table parameters in the tag. Note: the "columns" attribute only applies to the auto-completer.

```xml
<g:ui_reference name="parent_id" id="parent_id" table="pm_project" query="active=true" completer="AJAXTableCompleter" columns="project_manager;short_description"/>
```

### Ampersand

Ampersands in Jelly can cause you grief because Jelly is XML.

Use `${AMP}` to insert an ampersand in Jelly. If you are writing JavaScript that appears in the HTML part of say a UI page or UI macro that is actually going to run on the browser you are better off putting this code in the "client script" field and that way you can avoid escaping issues. However, if you really must put it in the "HTML" field, you will need to do something like this:

```javascript
ta = ta[1].split('$[AMP]');
```

### And

Use `${AND}` to insert a JavaScript and in Jelly.

For example:

```javascript
if (d ${AND} e)
    var color = d.value;
```

Alternately, in a Jelly test you would use `&amp;`. For example:
Less than

Similar to ampersands, less than (<) signs can also cause problems due to Jelly being XML. This can be resolved by reversing your test such that it is not necessary or by using ${AMP}lt; in place of the less than sign.

```xml
<g2:evaluate var="jvar_text">
    var days = ";
    var selectedDays = '${ref}';
    for (var i = 1; i <= 7; i++) {
        if (selectedDays.indexOf(i.toString()) >= 0) {
            days += gs.getMessage("dow" + i);
            days += " ";
        }
    }
    days;
</g2:evaluate>
```

Many times you can avoid the "less than" operator all together by just using "not equals" which doesn't have escaping issues. For example:

```xml
for (var i=0; i != ta.length; i++) {
}
```

Whitespace

Normally, white space is removed by Jelly. To keep it, you must specify that it not be trimmed.

For example, the following keeps the space after the colon.

```xml
<j2:whitespace trim="false">${gs.getMessage('Did you mean')}: </j2:whitespace>
```

Spaces

To encode a non-breaking space (&nbsp;), you can use $[SP]$.

For example:

```xml
<span id="gsft_domain" style="display: inline">
    ${gs.getMessage('Domain')}:$[SP]
    <span id="domainDD" class="drop_down_element" style="text-decoration: none; color: white">
        $[gs.getMessage("Loading...")]
    </span>
</span>
```
Tracing Jelly

ServiceNow has a feature that allows the evaluation of Jelly to be traced.

The trace is sent to the log. This should only be turned on during debugging as this produces a lot of logging. To turn on the trace, set the property glide.ui.template.trace to true. For example, the following script can be executed to do this:

GlideProperties.set('glide.ui.template.trace', true);

If you want to see your log entries on your web browser at the bottom of each page, navigate to System Diagnostics > Debug Log.

Useful scripts

Scripts that provide useful functionality not included in the core system.

Get a user object

In a business rule or other server script, the gs.getUser() method returns a user object. The user object is an internal representation of the currently logged in user and provides information about the user and various utility functions.

About this task

For a list and description of the available scoped methods for the user object, see GlideUser.

Procedure

1. Retrieve the current user.

   var myUserObject = gs.getUser()

2. Use the getUserByID method to fetch a different user using the user_name field or sys_id on the target record.

   For example:

   var ourUser = gs.getUser();
   gs.print(ourUser.getFirstName()); //print the first name of the user you are currently logged in as
   newUser = ourUser.getUserByID('abel.tuter'); //fetch a different user, using the user_name field or sys_id on the target user record.
   gs.print(newUser.getFirstName()); //first name of the user you fetched above
   gs.print(newUser.isMemberOf('Capacity Mgmt'));
Accessing the workflow scratchpad from business rules

A catalog item has been requested, the attached workflow contains a run script activity that populates a value in the scratchpad. From a business rule running on the requested item, we want to retrieve or set scratchpad values.

Prerequisites
Role required: admin

Name: Access Workflow Scratchpad from Business Rules
Type: Business Rule
Table: sc_req_item (Requested Item)

Description: A catalog item has been requested, the attached workflow contains a run script activity that populates a value in the scratchpad. From a business rule running on the requested item, we want to retrieve or set scratchpad values.

Parameters: n/a

Script:

```javascript
//the run script activity sets a value in the scratchpad
workflow.scratchpad.important_msg = "scratch me";

//get the workflow script include helper
var workflow = new Workflow();

//get the requested items workflow context
//this will get all contexts so you'll need to get the proper one if you have multiple workflows for a record
var context = workflow.getContexts(current);
//make sure we have a valid context
if (context.next()) {
    //get a value from the scratchpad
    var msg = context.scratchpad.important_msg;
    //msg now equals "scratch me", that was set in the run script activity

    //add or modify a scratchpad value
    context.scratchpad.status = "completed";
    //we need to save the context record to save the scratchpad
    context.update();
}
```
Add a field to the service catalog checkout

This is an example of adding a Company field to the checkout below the Requested for field using non-cart layout macros, that is, glide.sc.use_cart_layouts is false.

About this task

This field will then passing the value of that field to the Company field of the Service Catalog Request.

This example makes the following assumptions.

• This example is for an instance using two-step checkout. If two-step checkout is not enabled, enable it before beginning.

• This example populates the Company field on the Service Catalog Request form. If the field does not appear on the form, configure the form before beginning.

Procedure

1. Go to System UI > UI Macros and select servicecatalog_cart_template.
2. Find where there are hidden variables declared and add the following line:
   
   ```html
   <input type="HIDDEN" name="cart_id" id="cart_id" value="${sc_cart.sys_id}" />
   ```

3. Find the following code, which generates the Requested For code:

   ```html
   <tr class="header">
   <td width = "30%">
   ${gs.getMessage('Requested for')}: 
   </td>
   <td width="70%">
   <label for="requestor_location">${gs.getMessage('Deliver to')}:</label>
   </td>
   </tr>
   ```
4. Add the following code afterwards:

```html
<tr class="header">
    <td colspan="2">Company</td>
</tr>
<tr>
    <td>$[SP]</td>
</tr>
<tr>
    <td colspan="2">
        <g2:ui_reference name="core_company" table="core_company" onchange="setCartValue()" />
    </td>
</tr>
<tr>
    <td>$[SP]</td>
</tr>
```

- Note that 'ui_reference' is another macro that defines a reference field. There are several macros for different field types. You can see examples of these field types under System UI -> UI Macros. These macros start with 'ui_'. For this example, the reference field created is named core_company.

5. Now navigate to System UI > UI Pages and select the servicecatalog_checkout_one UI Page. Add this script to the Client script field:
function setCartValue() {
    var newField = gel('core_company');
    var myCart = gel('cart_id');
    var cart_item = new GlideRecord('sc_cart_item');
    cart_item.addQuery('cart', myCart.value);
    cart_item.query();
    if(cart_item.next()) {
        cart_item.hints = '<hints><entry key="sysparm_processing_hint" value="setfield:request.company=" + newField.value + "/"></hints>";
        cart_item.update();
    }
}

For this example, the reference field was called core_company, and the field being populated on the request is company. If different fields are used:

• Find this line: var company = gel('core_company'); and replace core_company with the name of the field in the checkout.

• In the line that starts with 'cart_item.hints' replace 'request.company' with the name of the field to be populated on the request ticket where 'request' is the request being generated and 'company' is the name of the field.

Results
Now, when an item is ordered, the company field appears on the Catalog form:
Add role to every user

Adds a role to every user.

Prerequisites

**Note:** This functionality requires a knowledge of JavaScript.

Role required: admin

In this sample, the role being added is Self Service. To add a different role, simply substitute the desired role for `self_service`.

**Name:** Add Role to Every User

**Type:** Client Script, Background Script

**Table:** sys_user

**Description:** Adds a role to every user. In this sample, the role being added is Self Service. To add a different role, simply substitute the desired role for `self_service`.

**Parameters:**

**Script:**
Assign a catalog item to a group based on a delivery plan task
This assignment rule assigns a service catalog item to the database group if it uses a delivery plan that has a catalog task assigned to the desktop group.

Prerequisites
Role required: admin

⚠️ CAUTION: The customization described here was developed for use in specific instances, and is not supported by ServiceNow Technical Support. This method is provided as-is and should be tested thoroughly before implementation. Post all questions and comments regarding this customization to our community forum.

Name: Assign Catalog Item to Group Based on Delivery Plan Task
Type: Assignment Rule
Table:
Description: This assignment rule assigns a service catalog item to the database group if it uses a delivery plan that has a catalog task assigned to the desktop group.

Parameters:

Script:
```javascript
//Return catalog items that have no group but do have a delivery plan assigned var ri = new GlideRecord("sc_cat_item"); ri.addQuery("group", "=", null); ri.addQuery("delivery_plan", "!", null); ri.query(); while(ri.next()) { gs.log("Found an item"); //Return tasks that point to the same delivery plan as the above item var dptask = new GlideRecord("sc_cat_item_delivery_task"); dptask.addQuery("delivery_plan", ",", ri.delivery_plan);
```
dptask.query();
while(dptask.next()) {
    gs.log("Found a task");
    var gp = dptask.group.getDisplayValue();
    gs.log(gp);
    //If the task is assigned to desktop, assign the item's group to desktop
    if (dptask.group.getDisplayValue() == "Desktop") {
        ri.group.setDisplayValue("Desktop");
        gs.log("updating " + ri.getDisplayValue());
        ri.update();
        break; } } } }

Change form color on state change
Changes color of a form field of the form on state change. The script can easily be changed to adjust any property of any object on the page accessible via the HTML DOM.

⚠️ CAUTION: The customization described here was developed for use in specific instances, and is not supported by ServiceNow Technical Support. This method is provided as-is and should be tested thoroughly before implementation. Post all questions and comments regarding this customization to our community forum.

Name: Change Form Color on State Change
Type: Client Script
Table:
Description: Changes color of a form field of the form on state change. The script can easily be changed to adjust any property of any object on the page accessible via the HTML DOM.
Parameters:
Script:

```javascript
function onChange(control, oldValue, newValue, isLoading) {
    var elementID = gel("incident.priority");
    switch(newValue) {
    case "1": elementID.style.backgroundColor = "red"; break;
    case "2": elementID.style.backgroundColor = "tomato"; break;
    case "3": elementID.style.backgroundColor = "orange"; break;
    case "4": elementID.style.backgroundColor = "yellow"; break;
    case "5": elementID.style.backgroundColor = "green"; break;
    default: elementID.style.backgroundColor = "white"; break; } }
```
Create a UI routing action

This solution enables you to create a record with the service desk without knowing whether it is an incident or request item; the service desk can then route the record to the appropriate table.

About this task

Note: Functionality described here requires the Admin role.

To create a UI routing action:

Procedure

1. Create a new table that extends the task table (for example, New Call).
2. Create a module to create a new New Call record.
3. Create any fields that you want on the New Call table.

   The only fields you need are those fields necessary to determine whether the new call should route to an Incident or a Request Item. Ensure that the form contains any fields that you want to pass to the Incident or Request Item. In this example, the following are created on the form:

   - **Requested for** (reference)
   - **Location** (reference)
   - **Call type** (choice with two values--Incident and Request)
   - **Request Item** (reference to the sc_cat_item Item table)

4. Add some UI policies to set a couple of fields to mandatory and hide the Request Item field based on the Call type selection.
5. Remove unnecessary buttons and functionality from the form.
6. Create a new UI Action button. This button redirects the user to either an incident or a request. It also creates the incident record and copies values to the incident and the Request Item form.

   ```javascript
   var reqItem = current.u_item;
   var requestedFor = current.u_requested_for;
   var location = current.location;

   if(current.u_incident_request == 'Incident') {
       //Create a new incident record and redirect to the new incident
       var rec = new GlideRecord('incident');
       rec.initialize();
       rec.caller_id = requestedFor;
   }
   ```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.

Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
rec.location = location;
rec.insert();
action.setRedirectURL(rec);
}

if(current.u_incident_request == 'Request'){
    //Build the url and route the user to the request item
    var url = ''; 
    if(current.u_item.sys_class_name == 'sc_cat_item_guide'){
        url = 'com.glideapp.servicecatalog_cat_item_guide_view.do?sysparm_initial=true&sysparm_guide='
            + reqItem + '&sysparm_user=' + requestedFor + '&sysparm_location=' + location;
    }else{
        url = 'com.glideapp.servicecatalog_cat_item_view.do?sysparm_id=' + reqItem +
            '&sysparm_user=' +
            requestedFor + '&sysparm_location=' + location;
    }
    action.setRedirectURL(url);
}

7. The Route button in the preceding example passes the Requested for and Location values in the URL to the Request Item form. Ensure that you have variables called requested_for and location on your item, record producer, or order guide that map these values using the following client script. There is a limit as to how much information you can pass, as the URL has a restricted length. Avoid sending information from long text fields using this method.

function onLoad() {
    var url = document.location.toString();
    var userKey = 'sysparm_user=';
    var locKey = 'sysparm_location=';
    var userPosition = url.indexOf(userKey);
    var locPosition = url.indexOf(locKey)
    if (userPosition != -1) {
        var user = url.substr(userPosition+userKey.length, 32);
        g_form.setValue('requested_for',user);
    }
    if (locPosition != -1) {
        var loc = url.substr(locPosition+locKey.length, 32);
        g_form.setValue('location',loc);
    }
}
Custom approval UI macro

This section describes how to create a custom approval UI macro.

⚠️ CAUTION: The customization described here was developed for use in specific instances, and is not supported by ServiceNow Technical Support. This method is provided as-is and should be tested thoroughly before implementation. Post all questions and comments regarding this customization to our community forum.

Script Name: Custom Approval UI Macro
Type: UI Macro
Description: Here is an option to get more detail out of the My Approvals view of an Execution Plan. This can be done by creating a new UI macro. Navigate to System UI and click UI Macros. First, you will need to rename the existing approval_summarizer_sc_task to something like approval_summarizer_sc_task_old and deactivate it. Then you will need to create a new one using the same name (approval_summarizer_sc_task). The name should basically tell you what the macro does and what it applies to. In this case, we’re replacing an existing one so we decided to re-use the existing name.

Then you should copy the xml script at the bottom of this article into the xml code window in the new UI macro. This is great way to give some detail to an approver when you are doing line item approvals via approval tasks within the Service Catalog Execution Plans.
The old way

This is the view you see in My Approvals when using an approval task via the old method.

My Approvals (old way)

Notice there is not much detail telling the approver what they are actually approving. You can see the short description of the task but not much around what the item is.

The new way

This is the view you will see if you use the xml script below in place of the OOB (out-of-box) UI macro.

My Approvals

Using this method you can see details much like the request approval. You have a link into the item ordered, a short description (which contains the ability to expand the variables from the item), price, quantity and the total price. This helps the approver in that it shows more detail. They can now see what they are actually approving.

Script:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<j:jelly trim="true" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null" xmlns:g2="null">
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.

4551
<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
<th>Price</th>
<th>Quantity</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<j:set var="jvar_line_num" value="0" />
<tr width="100%">
    <td width="100%">
        <table width="100%">
            <tr class="header">
                <td colspan="2">${sc_req_labels.number.sys_meta.label}</td>
                <td>${sc_req_labels.description.sys_meta.label}</td>
                <td>${sc_req_labels.price.sys_meta.label}</td>
                <td>${sc_req_labels.quantity.sys_meta.label}</td>
                <td>${gs.getMessage('Total')}</td>
            </tr>
            <j:set var="jvar_item_price" value="${sc_task.request_item.price * sc_task.request_item.quantity}"/>
            <j:set var="jvar_overall_total" value="${jvar_overall_total + jvar_item_price}"/>
            <j:set var="jvar_line_color" value="odd"/>
            <j:set var="jvar_line_num" value="${jvar_line_num + 1}"/>
            <j:if test="${jvar_line_num % 2 == 0}">
                <j:set var="jvar_line_color" value="even"/>
            </j:if>
            <g:evaluate var="ni" expression="'">
                var smart_description =
                sc_task.request_item.cat_item.short_description;
                if (smart_description == null || smart_description == '')
                    smart_description = 'undefined';
                smart_description = sc_task.request_item.cat_item.name;
            '</ni>
    </td>
</tr>
Display field messages

Rather than use JavaScript `alert()`, for a cleaner look, you can display an error on the form itself. The methods `showFieldMsg()` and `hideFieldMsg()` can be used to display a message just below the field itself.

`showFieldMsg` and `hideFieldMsg` are methods that can be used with the `g_form` object.
These methods are used to change the form view of records (Incident, Problem, and Change forms). These methods may also be available in other client scripts, but must be tested to determine whether they work as expected.

When a field message is displayed on a form on load, the form scrolls to ensure that the field message is visible. Ensuring that users do not miss a field message because it was off the screen.

The global property `glide.ui.scroll_to_message_field` controls automatic message scrolling when the form field is offscreen (scrolls the form to the control or field).

### Method Detail

<table>
<thead>
<tr>
<th>Method Detail</th>
<th>Parameters</th>
<th>Example</th>
</tr>
</thead>
</table>
| `showFieldMsg(input, message, type, [scrollform])` | - input — name of the field or control  
- message — message you would like to appear  
- type — 'info', 'error', or 'warning'; defaults to info if not supplied  
- scroll form — (optional) Set `scrollForm` to false to prevent scrolling to the field message offscreen | **Error Message**  
```
g_form.showFieldMsg('impact','Low impact not allowed with High priority','error');
```

<table>
<thead>
<tr>
<th>Impact:</th>
<th>3 - Low</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="https://example.com/warning.png" alt="Warning" /></td>
<td>Low impact not allowed with High priority</td>
</tr>
</tbody>
</table>

**Informational Message**  
```javascript
//or this defaults to info type  
//g_form.showFieldMsg('impact','Low impact response time can be one week');
```

<table>
<thead>
<tr>
<th>Impact:</th>
<th>3 - Low</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>Low impact response time can be one week</td>
</tr>
</tbody>
</table>
## Method Detail (continued)

<table>
<thead>
<tr>
<th>Method Detail</th>
<th>Parameters</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>hideFieldMsg(input)</td>
<td>• input — name of the field or control</td>
<td>Removing a Message</td>
</tr>
<tr>
<td></td>
<td>• clearAll — (optional) boolean parameter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>indicating whether to clear all messages.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If true, all messages for the field are cleared.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If false or empty, only the last message is</td>
<td></td>
</tr>
<tr>
<td></td>
<td>removed</td>
<td></td>
</tr>
</tbody>
</table>

### Removing a Message

```javascript
//this will clear the last message printed to the field
g_form.hideFieldMsg('impact');
```

## Legacy support

The `showErrorBox()` and `hideErrorBox()` are still available but simply call the new methods with type of error. You should use the new methods.

## GSLog

GSLog is a script include that simplifies script logging and debugging by implementing levels of log output, selectable by per-caller identified `sys_properties` values.

### Log level

Logs can be at the level of Debug, Info, Notice, Warning, Err, or Crit (after BSD syslog.h and followers). The default logging level is Notice, so levels should be chosen accordingly.

### Where to use

Use for any server-side script where you want to implement event logging.
Available Methods

**initialize(traceProperty, caller)**
Called by the Prototype JavaScript Framework during object creation to initialize a new instance of this class. Provide the input parameters, but do not call this method directly.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>traceProperty</td>
<td>String</td>
<td>System property that contains a value indicating the level at or above which messages will be written to the log.</td>
</tr>
<tr>
<td>caller</td>
<td>String</td>
<td>Name of the script calling the logger.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Example**

```javascript
var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");
```

**logDebug(msg)**
Logs debug events.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to write to the log</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Example**
var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");
gl.logDebug("This is a debug message");

Output:

*** Script [TaskSLA]: This is a debug message

logInfo(msg)
Logs information events.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to write to the log</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");
gl.logInfo("This is an info message");

logNotice(msg)
Logs notice events.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to write to the log</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");
gl.logNotice("This is a notice message");
var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");

// Output:

*** Script [TaskSLA]: This is a notice

### logWarning(msg)

Logs warning events.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to write to the log</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

#### Example

```javascript
var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");

// Output:

*** Script [TaskSLA]: This is a warning message
```

### logErr(msg)

Logs error events.

#### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to write to the log</td>
</tr>
</tbody>
</table>

#### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>
Example

```javascript
var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");
gl.logErr("This is an error message");
```

Output:

```text
*** Script [TaskSLA]: This is an error message
```

**logCrit(msg)**

Logs critical events.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to write to the log</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

```javascript
var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");
gl.logCrit("This is a critical message");
```

Output:

```text
*** Script [TaskSLA]: This is a critical message
```

**logAlert(msg)**

Logs alert events.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to write to the log</td>
</tr>
</tbody>
</table>
### logAlert(msg)
Logs an alert event.

**Example**

```javascript
var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");
gl.logAlert("This is an alert");
```

**Output:**
```
*** Script [TaskSLA]: This is an alert
```

### logEmerg(msg)
Logs emergency events.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to write to the log</td>
</tr>
</tbody>
</table>

**Returns**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

**Example**

```javascript
var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");
gl.logEmerg("This is an emergency message");
```

**Output:**
```
*** Script [TaskSLA]: This is an emergency message
```

### log(level, msg)
Logs a message at the specified level.

**Example**

```javascript
var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");
gl.log("This is a message");
```

**Output:**
```
*** Script [TaskSLA]: This is a message
```
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>level</td>
<td>String</td>
<td>Log level</td>
</tr>
<tr>
<td>msg</td>
<td>String</td>
<td>Message to write to the log</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

### Example

```javascript
var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");
gl.log("debug", "debug message");
```

#### getLevel(level)

Returns the log level.

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>level</td>
<td>String</td>
<td>Optional. Log level</td>
</tr>
</tbody>
</table>

### Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Log level</td>
</tr>
</tbody>
</table>

### Example

```javascript
var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");
gl.setLevel("debug");
gs.print(gl.getLevel());
```

**Output:**

```plaintext
*** Script: debug
```

#### setLevel(level)

Sets the log level.
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>level</td>
<td>String</td>
<td>Log level to set</td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void</td>
<td></td>
</tr>
</tbody>
</table>

Example

```
var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");
gl.setLevel("debug");
```

debugOn()

Determined if debug is turned on.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>If true, debug is on; if false, debug is off.</td>
</tr>
</tbody>
</table>

Example

```
var gl = new GSLog("com.snc.sla.tasksla.log", "TaskSLA");
gl.setLevel("debug");
gs.print(gl.debugOn());
```

Output:

```
*** Script : true
```

Modify a GlideDateTime field value

This example demonstrates how to modify a GlideDateTime field value using a script.
Name: Modify a GlideDateTime Field Value
Type: A server side script that accesses a GlideDateTime field.
Table: N/A
Description: Given a GlideDateTime field or script object, show a variety of ways to easily modify value. The same concept also applies to the GlideDate object.
Parameters: N/A

Script:
```javascript
//You first need a GlideDateTime object
//this can be from instantiating a new object "var gdt = new GlideDateTime()"
//or getting the object from a GlideDateTime field
//getting the field value (for example: var gdt = current.start_date) only returns the string value, not the object
//to get the object use var gdt = current.start_date.getGlideObject();
//now gdt is a GlideDateTime object
var gdt = current.start_date.getGlideObject();

//All methods can use negative values to subtract intervals

//add 1 hour (60 mins * 60 secs)
gdt.addSeconds(3600);

//add 1 day
gdt.addDaysLocalTime(1);

//subtract 1 day
gdt.addDaysLocalTime(-1);

//add 3 weeks
gdt.addWeeksLocalTime(3);

//subtract 6 months
gdt.addMonthsLocalTime(-6);

//add 1 year, representing the date and time using the UTC timezone instead of the local user's timezone.
gdt.addYearsUTC(1);

//set the value of the GlideDateTime object to the current session timezone/format
GlideSession.get().setTimeZoneName('US/Eastern');
gdt.setDisplayValue('2018-2-28 00:00:00');
gs.info('In ' + GlideSession.get().getTimeZoneName() + " : " + gdt.getDisplayValue());
```
Sample ASP.NET with C Sharp redirect with cookies

This sample ASP .NET code creates a simple authentication portal and passes an unencrypted HTTP header as a cookie.

⚠️ Note: Functionality described here requires the Admin role.

⚠️ Note: Cookies are domain specific and cannot be used across different network domains. The only domain that can read a cookie is the domain that sets it. It does not matter what domain name you set. If you do not have an option of your SSO portal being in the same network domain as your ServiceNow instance (for example, in an on-premisis deployment, an alternative is to pass the SSO token using URL GET or POST parameters.

This sample assumes:

- The web server supports ASP .NET and C#
- The target ServiceNow instance is https://<instance name>.service-now.com/
- SiteMinder or another single sign-on application has pre-authenticated the user
- The target ServiceNowinstance expects an HTTP header of SM_USER

Change the ASP code to redirect users to the proper ServiceNow instance.

```html
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Portal Page Login</title>
    <%- --
        <meta http-equiv="REFRESH" content="0;url=https://<instance name>.service-now.com/"--%>
    </%- --

</head>
<body>
    <form id="form1" runat="server">
        <h2><b> Portal Page Login </b></h2>
        <hr style="position: static" />
        <br />
        <asp:Label ID="Label2" runat="server" Font-Size="Larger" Height="21px" Style="position: static"
            Text="Instance URL:" Width="113px"></asp:Label>
        <asp:TextBox ID="urlBox" runat="server" Font-Size="Large" Style="position:
            static"></asp:TextBox><br />
        <br />
```
The following C# code handles the `OnClick` button event for the form. The code:

- Creates the cookie “SM_USER”
- Performs a redirect to the URL specified on the ASP form.

Change the C# code to create the proper cookie name.

```csharp
using System;
using System.Data;
using System.Configuration;
```
using System.Web;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    
    protected void Button1_click(object sender, EventArgs e)
    {
        try
        {
            HttpCookie myCookie = new HttpCookie("SM_USER");
            myCookie.Value = userNameBox.Text;
            Response.Cookies.Add(myCookie);
            Response.Redirect(urlBox.Text);
        }
        catch
        {}
    }
}

Useful approval assignment scripts

This is a searchable version of the useful approval and assignment scripts.

⚠️ CAUTION: The customization described here was developed for use in specific instances, and is not supported by ServiceNow Technical Support. This method is provided as-is and should be tested thoroughly before implementation. Post all questions and comments regarding this customization to our community forum.

For an easy-to-navigate version, visit the Useful Scripts portal.

Assign a group for ESS requests

Type: Assignment Rule

Description: This script automatically assigns a group for all ESS Requests.

Script example:

```csharp
if(current.opened_by.roles=""){
    current.assignment_group.setDisplayValue('Network');
    current.update();
}
```

Assign Catalog Item to Group Based on Delivery Plan Task
Type: Assignment Rule

Description: This assignment rule assigns a service catalog item to the database group if it uses a delivery plan that has a catalog task assigned to the desktop group.

```javascript
//Return catalog items that have no group but do have a delivery plan assigned
var ri = new GlideRecord("sc_cat_item");
ri.addQuery("group","=" ,null);
ri.addQuery("delivery_plan","!=" ,null);
ri.query();
while(ri.next()){
    gs.log("Found an item");
    //Return tasks that point to the same delivery plan as the above item
    var dptask = new GlideRecord("sc_cat_item_delivery_task");
dptask.addQuery("delivery_plan","=" ,ri.delivery_plan);
dptask.query();
    while(dptask.next()){
        gs.log("Found a task");
        var gp = dptask.group.getDisplayValue();
        gs.log(gp);
        //If the task is assigned to desktop, assign the item's group to desktop
        if(dptask.group.getDisplayValue()=="Desktop"){
            ri.group.setDisplayValue("Desktop");
            gs.log(" updating "+ ri.getDisplayValue());
            ri.update();
            break;}}
}
```

Assign items with one task

Type: Assignment Rule

Description: Automatically assigns any catalog items with only one task associated to a particular group.

```javascript
//Get the catalog item for the current requested item
var scCatItem = new GlideRecord("sc_cat_item");
if(scCatItem.get('sys_id', current.cat_item)){
    // If the catalog item already has an assignment group or if using workflow we don't need to make an assignment
    if(!scCatItem.delivery_plan.nil() && scCatItem.group.nil()){)
        var dpTask = new GlideRecord("sc_cat_item_delivery_task");
dpTask.addQuery("delivery_plan","=" ,scCatItem.delivery_plan);
dpTask.query();
        if(dpTask.getRowCount()==1 && dpTask.next()){
            // Check that there is only 1 record in the GlideRecord
            dpTask.group;}}}
```

Assignment based on workload

Type: Business Rule
Description: Populate the assigned to based on the assignment group member who has the least amount of active incidents.

Parameters:

- **order**: >1000 if you want to execute after assignment rules
- **condition**: current.assigned_to == " & current.assignment_group != "
- **when**: before, insert/update

```javascript
var assignTo = getLowestUser();
gs.addInfoMessage("assigning to is " + assignTo);
current.assigned_to= assignTo;

function getLowestUser(){
    var userList =new Array();
    var cg =new GlideRecord('sys_user_grmmember');
    cg.addQuery('group', current.assignment_group);
    cg.query();
    while(cg.next()){  
        var tech = cg.user.toString();
        var cnt = countTickets(tech);
        gs.addInfoMessage("Tech counts " + cg.user.name + ' ' + cnt + ' ' + tech);
        userList.push({ sys_id: tech, name: cg.user.name, count: cnt });
    }
    for(var i=0; i < userList.length; i++){  
        gs.addInfoMessage(userList[i].sys_id + " " + userList[i].name + " " + userList[i].count);
    }
    userList.sort(function(a, b){
        gs.addInfoMessage("Sorting: " + a.sys_id + "(" + a.count + ");" + b.sys_id + "(" + b.count + ");
        return a.count - b.count;
    });
    if(userList.length<=0)return"";
    return userList[0].sys_id;
}

function countTickets(tech){
    var ct =new GlideRecord('incident');
    ct.addQuery('assigned_to',tech);
    ct.addQuery('active',true);
    ct.query();
    return ct.getRowCount();
}
```

Run assignment rules when category is changed

Type: Client script
Table: Incident

Description: This example is an onChange client script on the category field within Incident. Note: this script used to use synchronous AJAX (asynchronous behavior is specified by the third parameter of the ajaxRequest call). The implementation below uses asynchronous AJAX. The drawback of using the synchronous version is that a network response problem could cause the browser to hang.

```javascript
// Make an AJAX request to the server to get who this incident would be
// assigned to given the current values in the record. This runs the assignment
// rules that have been defined in System Policy and returns the assigned_to and
// the assignment_group

function onChange(control, oldValue, newValue, isLoading){
    if(isLoading){return;
    // No change, do not do anything
    }

    // Construct the URL to ask the server for the assignment
    var url = "xmlhttp.do?sysparm.processor=AJAXAssignment&sys_target=incident";
    var uv = gel('sys_uniqueValue');
    if(uv){
        url += '&sys_uniqueValue=' + uv.value;
    // Make the AJAX request to the server and get the response
    var serial = g_form.serialize();
    // get all values currently assigned to the incident
    var response = ajaxRequest(url, serial, true, responseFunc);
}

// This callback function handles the AJAX response.
function responseFunc(response){
    var item = response.responseXML.getElementsByTagName("item")[0];
    // Process the item returned by the server
    if(item){
        // Get the assigned_to ID and its display value and put them on the form
        var name = item.getAttribute("name");
        var name_label = item.getAttribute("name_label");
        if(name_label && name){
            g_form.setValue('assigned_to', name, name_label);
        } else{
            g_form.setValue('assigned_to', '', '');
        // Get the assignment_group ID and its display value and put them on the form
        var group = item.getAttribute("group");
        var group_label = item.getAttribute("group_label");
        if(group_label && group){
```
Custom approval UI macro

Type: UI macro

The following option illustrates how to obtain more detail from the My Approvals view of an Execution Plan by creating a new UI Macro.

- **Navigate to System UI and click UI Macros.**
- **Rename the existing “approval_summarizer_sc_task” to something like “approval_summarizer_sc_task_old” and deactivate it.**
- **Create a new one using the same name (“approval_summarizer_sc_task”). The name should basically tell you what the macro does and to what it applies. In this case, we’re replacing an existing one so we decided to re-use the existing name.**

Then copy the xml script at the bottom of this article into the xml code window in the new UI Macro. This is great way to give some detail to an approver when you are doing line item approvals using approval tasks within the Service Catalog Execution Plans.

**Different ways**

**Old way**

This is the view you see in My Approvals when using an approval task using the old method.
Notice there is not much detail telling the approver what they are actually approving. You can see the short description of the task but not much information about the item.

**New way**

This is the view you will see if you use the xml script below in place of the OOB (out-of-box) UI Macro.

Using this method you can see details much like the request approval. You have a link into the item ordered, a short description (which contains the ability to expand the variables from the item), price, quantity and the total price. This helps the approver in that it shows more detail. They can now see what they are actually approving.

**Useful field scripts**

This is a searchable version of the useful field customization scripts.

⚠️ **CAUTION:** The customization described here was developed for use in specific instances, and is not supported by ServiceNow Technical Support. This method is provided as-is and should be tested thoroughly before implementation. Post all questions and comments regarding this customization to our community forum.

For an easy-to-navigate version, visit the Useful Scripts portal.
AKA Incident Template, Auto Assignments, Quick Calls, Call Script, Auto Populate

Let's say you want to auto-fill your Short Description based on the selected Subcategory. First, create a lookup table, then populate the key field, in this case the Subcategory and the auto-filled field, Short Description. So let's say your table had a record with Subcategory = Password and Short Description = Password Reset. When the user selects the Subcategory of Password on the Incident form, a client script looks up the matching record and sets Short Description equal to Password Reset.

Client script settings... Type = onChange, Table name = incident, Field name = Subcategory

```javascript
function onChange(control, oldValue, newValue, isLoading){
  if(isLoading){return;}
  var newrec = gel('sys_row');
  //Check if new record
  if (newrec.value == -1) {
    var lookup = new GlideRecord('u_short_desc_lookup');
    lookup.addQuery('u_subcategory', g_form.getValue('subcategory'));
    lookup.query();
    var temp; //temp var - reusable
    if(lookup.next()){
      temp = lookup.u_short_description;
      if(null != temp) {
        //Set the form value from lookup if there is a lookup value
        g_form.setValue('short_description', temp);
      } else {
        g_form.setValue('short_description','');
      }
    } else {
      //If a lookup record does not exist based on lookup.addQuery
      //Then set to UNDEFINED or NULL depending on type
      g_form.setValue('short_description','');
    }
  }
}
```

You could populate multiple fields or even pull Call Script questions into the Comments field so call center personnel gather good information to pass on to a technician. There are already Assignment Rules, Templates, and Wizards built in that perform similar functions.

Disable HTML tags in descriptions
Description: This code disables HTML tags in descriptions and short descriptions
by substituting the tags with harmless versions that won't execute.
doit();

function doit(){
var desc = current.description.toString();
var shdesc = current.short_description.toString();
if(desc.indexOf('script>')>-1|| shdesc.indexOf('script>')>-1){
desc = desc.replace(/<script>/g,"(script)");
current.description = desc.replace(/<\/script>/g,"(\/script)");
shdesc = shdesc.replace(/<script>/g,"(script)");
current.short_description = shdesc.replace(/<\/script>/g,"(\/script)");}
}

Eliminate leading and trailing spaces in fields
Table: sys_user
Description: This example of the script trims trailing and leading spaces in the
FirstName and LastName fields of sys_user.
doit();

function doit(){
var now_GR =new GlideRecord('sys_user');
gr.query();
while(gr.next()){
if((gr.first_name.toString().length!=
gr.first_name.toString().trim().length)||(gr.last_name.toString().length!=
gr.last_name.toString().trim().length)){
gr.first_name= gr.first_name.toString().trim();
gr.last_name= gr.last_name.toString().trim();
gr.autoSysFields(false);
gr.update();}}
}

Make a field label flash
Type: Client script
Description: The following example is for the number field on incident. The label
will flash for two seconds:
g_form.flash("incident.number","#FFFACD",0);

The arguments for the flash method are as follows:

4574

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


1. `tablename.fieldname`

2. RGB color or acceptable CSS color like "blue" or "tomato"

3. integer that determines how long the label will flash:

   - Use 2 for a 1-second flash
   - Use 0 for a 2-second flash
   - Use -2 for a 3-second flash
   - Use -4 for a 4-second flash

Do not specify this argument if you want the field label simply colored the specified color.

**Make field label bold**

Type: Client script

Description: This script makes the label of a particular field (in this case, **Short Description** on the **Incident Table**) bold.

```javascript
function onLoad(){
    var l = g_form.getLabel('incident.short_description');
    l.style.fontWeight = 'bold';}
```

**Make fields read-only**

Type: Client script

Table: Incident

Description: This onLoad client script makes fields read-only. For this example, the script makes the following fields on the Incident table read-only: **Incident state**, **Impact**, **Urgency**, **Priority**, **Configuration item**, and **Assigned to**. It also removes the magnifying glass for the read-only Reference Fields (**Configuration item** and **Assigned to**).

```javascript
function onLoad(){
    var incidentState = g_form.getValue('incident_state');
    if( incidentState == '6'|| incidentState == '7'){
        g_form.setReadonly('incident_state',true);
        g_form.setReadonly('impact',true);
        g_form.setReadonly('urgency',true);
        g_form.setReadonly('priority',true);
        g_form.setReadonly('cmdb_ci',true);
        g_form.setReadonly('assigned_to',true);}}
```

**Set current date/time in field**
Type: Client script

Description: You can use the following two lines to set the current date and
time in a date/time field. This bypasses the problem of getting the value into the
proper format and proper timezone.

```javascript
var ajax = new GlideAjax('MyDateTimeAjax');
ajax.addParam('sysparm_name','nowDateTime');
ajax.getXML(function(){
    g_form.setValue('put your field name here', ajax.getAnswer());});
```

For more information on running server side scripts with the client, refer to
GlideAjax.

System script include

```javascript
// Be sure the "Client callable" checkbox is checked

var MyDateTimeAjax = Class.create();
MyDateTimeAjax.prototype = Object.extendsObject(AbstractAjaxProcessor, {
    nowDateTime:function(){
        return gs.nowDateTime();}});`
Modify GlideDateTime field value

Type: A server side script that accesses a GlideDateTime field.

Description: Given a GlideDateTime field or script object, show a variety of ways to easily modify value. The same concept also applies to the GlideDate object.

//You first need a GlideDateTime object
//this can be from instantiating a new object "var gdt = new GlideDateTime()"
//for getting the object from a GlideDateTime field
//getting the field value (for example: var gdt = current.start_date)
//only returns the string value, not the object
//to get the object use var gdt = current.start_date.getGlideObject();
//now gdt is a GlideDateTime object
var gdt = current.start_date.getGlideObject();

//All methods can use negative values to subtract intervals

//add 1 hour (60 mins * 60 secs)
gdt.addSecondsLocalTime(3600);

//add 1 day
gdt.addDaysLocalTime(1);

//subtract 1 day
gdt.addDaysLocalTime(-1);
Useful scheduling scripts

A business rule script specifies the actions that the business rule takes. Scripts commonly include predefined global variables to reference items in your system, such as the current record. Global variables are available to all business rules.

⚠️ **CAUTION:** The customization described here was developed for use in specific instances, and is not supported by ServiceNow Technical Support. This method is provided as-is and should be tested thoroughly before implementation. Post all questions and comments regarding this customization to our community forum.

Calculate duration given a schedule

Type: Before update/insert business rule

Table: Incident

Description: A Business Duration calculates the Open to Close duration on an incident based on the particular Creating and using schedules. If there is no schedule specified, the script will simply use the first schedule returned by the query.

Script example:

The example below sets the resolved duration when the incident state moves to resolved.

```javascript
if(current.incident_state==6){
  var dur = calcDurationSchedule(current.opened_at, current.sys_updated_on);
  current.u_resolved_duration= dur;
}

function calcDurationSchedule(start, end){
  // Get the user
  var usr = new GlideRecord('sys_user');
  // Other code...
```

© 2021 ServiceNow, Inc. All rights reserved.
usrgef(gs.getUserID());
// Create schedule - pass in the sys_id of your standard work day schedule and pass in
the users timezone
var sched = new GlideSchedule('08fcd0830a0a0b2600079f56b1adb9ae',usr.time_zone);
// Get duration based on schedule/timezone
return(sched.duratiom(start.getGlideObject(), end.getGlideObject()));}

Check upcoming termination dates

Type: Scheduled script

Description: This script checks nightly for termination dates on contracts coming
up in 90, 50, or 10 days (depending on the contract duration field).

Script example:

function contractNoticeDue(){
 var now_GR = new GlideRecord("contract");
 gr.addQuery("u_contract_status","Active");
 gr.query();
 while(gr.next()){  
  if((gr.u_termination_date<= gs.daysAgo(-90))&&(gr.u_contract_duration=="Long")){
   gr.u_contract_status="In review";
  }elseif((gr.u_termination_date<= gs.daysAgo(-50))&&(gr.u_contract_duration=="Medium")){
   gr.u_contract_status="In review";
  }elseif((gr.u_termination_date<= gs.daysAgo(-10))&&(gr.u_contract_duration=="Short")){
   gr.u_contract_status="In review";
  }
  gr.update();
}
}

Use scripts in business rules to accomplish common tasks such as:

- Comparing two date fields.
- Parsing XML payloads.
- Aborting a database action in a business rule.

With scripts, you can also:

- Specify the operation that triggers the business rule.
- Use the scratchpad with display business rules to change form values just
  before a user loads the form.
- Use the OR condition like you would in a condition builder.

You can also utilize the system’s scripting functionality available for server-side
scripts.
You can use options on the Business Rules form to build conditions, set field values, and display alert messages without needing to write a script.

**Abort a database action in a business rule**

During a before business rule script, you can cancel or abort the current database action using the current.setAbortAction(true) method.

For example, if the before business rule is executed during an insert action, and you have a condition in the script that calls current.setAbortAction(true), the new record stored in current is not created in the database.

**Add autofill functionality**

Add autofill functionality is also called incident template, auto assignments, quick calls, call script, or auto populate.

Let's say you want to auto-fill your **Short Description** based on the **Subcategory** selected. First, create a lookup table, then populate the key field, in this case **Subcategory** and the auto-filled field, **Short Description**. So let's say your table had a record with **Subcategory** = Password and **Short Description** = Password Reset. When the user selects the subcategory of Password on the Incident form a client script looks up the matching record and sets short description equal to Password Reset. Client script settings... **Type** = onChange, **Table name** = incident, **Field name** = Subcategory.

```javascript
function onChange(control, oldValue, newValue, isLoading) {
    if (isLoading) { return; }
    var newrec = gel('sys_row');
    //Check if new record
    if (newrec.value == -1) {
        var lookup = new GlideRecord('u_short_desc_lookup');
        lookup.addQuery('u_subcategory', g_form.getValue('subcategory'));
        lookup.query();
        var temp; //temp var - reusable
        if (lookup.next()) {
            temp = lookup.u_short_description;
            if (null != temp) { //Set the form value from lookup if there is a lookup value
                g_form.setValue('short_description', temp);
            } else {
                g_form.setValue('short_description', "" );
            }
        } else {
            //If a lookup record does not exist based on lookup.addQuery
            //Then set to UNDEFINED or NULL depending on type
            g_form.setValue('short_description', "");
        }
    }
}
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
You could populate many fields or even pull in call script questions into the Comments field so call center personnel gather good information to pass on to a technician. There are already Assignment Rule, Templates and Wizards built in that perform similar functions.

**Example script: A default before-query business rule**

You can use a query business rule that executes before the database query is made to prevent users from accessing certain records.

⚠️ **CAUTION:** The customization described here was developed for use in specific instances, and is not supported by ServiceNow Technical Support. This method is provided as-is and should be tested thoroughly before implementation. Post all questions and comments regarding this customization to our community forum.

Consider the following example from a default business rule that limits access to incident records.

### Default business rule limits access to incident records

<table>
<thead>
<tr>
<th>Name</th>
<th>Table</th>
<th>When</th>
</tr>
</thead>
<tbody>
<tr>
<td>incident query</td>
<td>Incident</td>
<td>before, query</td>
</tr>
</tbody>
</table>

**Example script**

This example prevents users from accessing incident records unless they have the itil role are listed in the Caller or Opened by field. So, for example, when self-service users open a list of incidents, they can only see the incidents they submitted.

```javascript
if(!gs.hasRole("itil")&& gs.isInteractive()){  
  var u = gs.getUserID();  
  var qc = current.addQuery("caller_id", u).addOrCondition("opened_by", u).addOrCondition("watch_list","CONTAINS", u);  
  gs.print("query restricted to user: "+ u);  
}
```

⚠️ **Note:** You can also use access controls to restrict the records that users can see.

**Schedule script for weekdays**

Type: Business Rules/Client Scripts
This script schedules the script for weekdays. Insert any script where it says "Your Script Here."

```javascript
var go = 'false';
var now = new Date();

// Correct time zone, which is by default GMT -7
now.setHours(now.getHours() + 8);
var day = now.getDay();

// No go on Saturday or Sunday
if (day != 0 && day != 6) {
    // (your script here)
}
```

### Set date field according to current date

This script sets a date field depending on the current day of the week. In this example, if the day is Monday through Wednesday, it sets the date to this coming Monday; otherwise it sets the date field to next Monday.

```javascript
function setCabDate() {
    var today = new Date();
    var thisDay = today.getDay();
    // returns 0 for Sunday, 1 for Monday, etc. thru 6 for Saturday.
    var thisMon = new GlideDateTime();
    thisMon.setDisplayValue(gs.beginningOfThisWeek());
    var nextMon = thisMon.getNumericValue();
    nextMon += (1000 * 60 * 60 * 24 * 7);

    if ((thisDay < 4) && (thisDay > 0)) {
        // if today is Mon thru Wed (thisDay = 1, 2, or 3), set cab to this coming Monday.
        current.u_req_cab_rev_date.setDateNumericValue(thisMon.getNumericValue());
    } else if ((thisDay >= 4) || (thisDay == 0)) {
        // if today is Thurs thru Sun (thisDay = 4, 5, 6, or 0), set cab to next Monday.
        current.u_req_cab_rev_date.setDateNumericValue(nextMon);
    }
}
```

To validate the input of all date/time fields, you can use the following in a validation script ([System Definition > Validation Scripts](#)). Because the date/time format is hard coded in this script, it must match your instance's date/time format. If your instance's date/time format changes, you must update your validation script.
Set the validation script’s type to *Date/Time*. Then, with this validation script, if a user enters an incorrect format in a date/time field, they will receive an error message.

```javascript
function validate(value){
  // empty fields are still valid dates
  if(!value)returntrue;

  // We "should" have the global date format defined always defined. but there's always that
  // edge case...
  if(typeof g_user_date_time_format !=='undefined')return isDate(value,
    g_user_date_time_format);

  // if we don't have that defined, we can always try guessing
  return parseDate(value)!==null;}
```

### Date/time validation

![Validation Script](image)

**Sample ASP Script for unencrypted single sign-on**

This sample ASP .NET code creates a simple authentication portal and passes an unencrypted HTTP header as a URL parameter.

This sample assumes:

- The web server supports ASP .NET
- The target instance is `https://<instance name>.service-now.com/`
• SiteMinder or another single sign-on application has pre-authenticated the user
• The target instance expects an HTTP header of SM_USER

Change the ASP code to redirect users to the proper instance and create the proper HTTP header.

```html
<html xmlns = "http://www.w3.org/1999/xhtml" > <head runat = "server" > <title >Portal Page Login </title > </head > <body > <form id = "form1"  runat = "server" > <h2 >Portal Page Login </h2 > <hr style = "position: static" / > <br / > <asp:Label ID = "Label2"  runat = "server"  Font-Size = "Larger" Height = "21px" Style = "position: static"; Text = "Instance URL:" Width = "113px" ><br /> <asp:TextBox ID = "urlBox"  runat = "server"  Font-Size = "Large" Style = "position: static;" ><br /></asp:TextBox> <br / > <br / > <asp:Label ID = "Label1"  runat = "server"  Font-Size = "Larger" Height = "17px" Style = "position: static;" Text = "User Id:" Width = "113px" ><br /> <asp:TextBox ID = "userNameBox" runat = "server" Font-Size = "Large" Style = "position: static;" ><br /></asp:TextBox> <br / > <br / > <asp:Button ID = "Button1"  runat = "server" Height = "39px" Style = "position: static;" Text = "Ok" Width = "88px" OnClick = "go_to" / > </form > </body > </html >
```
To validate the input of all date/time fields, you can use the following in a validation script (System Definition > Validation Scripts).

Because the date/time format is hardcoded in this script, it must match your instance's date/time format. If your instance's date/time format changes, you must update your validation script.

Set the validation script's type to "glide_date_time". Then, with this validation script, if a user enters an incorrect format in a date/time field, they will receive an error message.

```
function validate (value ) { if ( !value ) { return true ; } return (getDateFromFormat (value , 'yyyy-MM-dd HH:mm:ss' ) != 0 ) ; }
```
Calculating durations

Often you may need to provide users with a way to specify when a task or process is due. Using the DurationCalculator script include, you can calculate the due date, using either a simple duration or relative duration.

For information on schedules, which you can use as inputs to DurationCalculator methods, see Creating and using schedules.

This script demonstrates how to use DurationCalculator to compute a due date.

```javascript
/**
 * Demonstrate the use of DurationCalculator to compute a due date.
 *
 * You must have a start date and a duration. Then you can compute a
 * due date using the constraints of a schedule.
 */

gs.include('DurationCalculator');
executeSample();

/**
 * Function to house the sample script.
 */function executeSample(){

    // First we need a DurationCalculator object.
    var dc = new DurationCalculator();

    // No schedule examples
```
// Simple computation of a due date without using a schedule. Seconds are added to the start date continuously to get to a due date.
dc.setStartDateTime("5/1/2012");
if(!dc.calcDuration(2*24*3600)){
gs.log("**** Error calculating duration");return;
gs.log("calcDuration no schedule: "+ dc.getEndDateTime());
// "2012-05-03 00:00:00" two days later
}

// Start in the middle of the night (2:00 am) and compute a due date 1 hour in the future// Without a schedule this yields 3:00 am.
dc.setStartDateTime("5/3/2012 02:00:00");
if(!dc.calcDuration(3600)){
gs.log("*** Error calculating duration");return;
gs.log("Middle of night + 1 hour (no schedule): "+ dc.getEndDateTime());
// No scheduled start date, just add 1 hour
}

// -------------- Add a schedule to the date calculator ---------------------
addSchedule(dc);

// Start in the middle of the night and compute a due date 1 hour in the future./ Since we start at 2:00 am the computation adds the 1 hour from the start// of the day, 8:00am to get to 9:00am
dc.setStartDateTime("5/3/2012 02:00:00");
if(!dc.calcDuration(3600)){
gs.log("*** Error calculating duration");return;
gs.log("Middle of night + 1 hour (with 8-5 schedule): "+ dc.getEndDateTime());
// 9:00 am
}

// Start in the afternoon and add hours beyond quiting time. Our schedule says the work day// ends at 5:00pm, if the duration extends beyond that, we roll over to the next work day.// In this example we are adding 4 hours to 3:00pm which gives us 10:00 am the next day.
dc.setStartDateTime("5/3/2012 15:00:00");
if(!dc.calcDuration(4*3600)){
gs.log("*** Error calculating duration");return;
gs.log("Afternoon + 4 hour (with 8-5 schedule): "+ dc.getEndDateTime());
// 10:00 am.
}

// This is a demo of adding 2 hours repeatedly and examine the result. This is a good way to visualize the result of a due date calculation.
dc.setStartDateTime("5/3/2012 15:00:00");
for(var i=2; i<24; i+=1)
if(!dc.calcDuration(i*3600)){
    gs.log("*** Error calculating duration");return;
    gs.log("add "+ i +" hours gives due date: "+ dc.getEndDateTime());
}
Simple duration vs relative duration

How much work is required to complete a task can be expressed as a "relative duration".

Relative duration determines the expected due date and time relative to the starting time. Examples of relative durations include "Next business day by 4pm," or "2 business days by 10:30am."

To calculate a relative duration, the calendar and time zone must be considered to determine what "next business day" means since it is the calendar that defines which days are valid work days and the time zone will affect the result as well. As an example, consider "Next business day by 4pm":

- If it is Monday at 12pm: Next business day by 4pm => Tuesday at 4pm
- If it is Friday at 2pm: Next business day by 4pm => the following Monday at 4pm

⚠️ Note: Next business day is often defined by a starting day and time. For example, "next business day at 4pm if before 2pm" indicates that if the current time is after 2pm on a business day, then "Next business day" really means 2 business days since today does not count.
Calculating a simple duration

This business rule and script example demonstrate how to calculate a simple duration.

```javascript
var dur = new DurationCalculator();
dur.setSchedule(current.schedule);
dur.setStartDateTime("");

if(current.duration_type=="""){
    dur.calcDuration(current.duration.getGlideObject().getNumericValue()/1000);}
else{
    dur.calcRelativeDuration(current.duration_type);
}

current.end_date_time= dur.getEndDateTime();
current.work_seconds= dur.getSeconds();
```

This script demonstrates how to use DurationCalculator to calculate a simple duration.

```javascript
/**
 * Sample script demonstrating use of DurationCalculator to compute simple durations
 *
 */

gs.include('DurationCalculator');
executeSample();

/**
 * Function to house the sample script.
 */
function executeSample(){

    // First we need a DurationCalculator object.
    var dc = new DurationCalculator();

    // Compute a simple duration without any schedule. The arguments
    // can also be of type GlideDateTime, such as fields from a GlideRecord.
    var dur = dc.calcScheduleDuration("5/1/2012","5/2/2012");
    gs.log("calcScheduleDuration no schedule: "+ dur);
    // 86400 seconds (24 hours)

    // The above sample is useful in limited cases. We almost always want to
    // use some schedule in a duration computation, let's load a schedule.
    addSchedule(dc);

    // Compute a duration using the schedule. The schedule
```
Calculating a relative duration

An example of a relative duration calculation script.

This script calculates the relative duration for "Next day at 4pm if after 10am":

```javascript
// Next day at 4pm if before 10am
var days = 1;
if (calculator.isAfter(calculator.startDateTime, "10:00:00"))
    days++;
```
This script demonstrates how to use DurationCalculator to calculate a relative duration.

```java
/**
 * Sample use of relative duration calculation.
 *
 */

gs.include('DurationCalculator');
executeSample();

/**
 * Function to house the sample script.
 */
function executeSample(){

    // First we need a DurationCalculator object. We will also use
    // the out-of-box relative duration "2 bus days by 4pm"
    var dc = new DurationCalculator();
    var relDur = "3bf802c20a0a0b52008e2859cd8abcf2";
    // 2 bus days by 4pm if before 10am
    addSchedule(dc);

    // Since our start date is before 10:00am our result is two days from
    // now at 4:00pm.
    dc.setStartDateTime("5/1/2012 09:00:00");
    if(!dc.calcRelativeDuration(relDur)){
        gs.log("*** calcRelativeDuration failed");
            return;
    }
    gs.log("Two days later 4:00pm: " + dc.getEndDateTime());

    // Since our start date is after 10:00am our result is three days from
    // now at 4:00pm.
    dc.setStartDateTime("5/1/2012 11:00:00");
    if(!dc.calcRelativeDuration(relDur)){
        gs.log("*** calcRelativeDuration failed");
            return;
    }
    gs.log("Three days later 4:00pm: " + dc.getEndDateTime());
}

/**
 * Add a specific schedule to the DurationCalculator object.
 */
```
Elapsed time vs work time

Typically, setting a due date requires that you calculate the actual work time required for completion rather than the total time that elapses until the due date.

SLAs, Workflow tasks, and approvals are examples of situations that require the actual work rather than the total time that has elapsed. In this case, only the part of the day when work is performed is considered when determining when the work is to be complete. For example, if a task is due in 10 hours, but the actual time to perform the task is restricted to a business day schedule, the 10 hours of work can only be done at most 8 hours each day. If this work starts at 10am on Monday, it is expected to complete, or be due, on Tuesday at 12pm:

10am-5pm on Monday (6 hours) + 8am-12pm on Tuesday (4 hours)

How to implement a relative duration

You can implement a relative duration by creating the cmn_relative_duration table and the DurationCalculator script include.

Before you begin
Role required: admin

Procedure
1. Create the cmn_relative_duration table.
2. Create the DurationCalculator script include.
3. Create a sample relative duration entry (for example, "Next business day by 4pm").
4. Add the needed fields to SLA tables to support relative durations.
5. Modify duration calculation for SLAs.
6. Modify SLA Percentage timer calculation for SLAs (this must use work_seconds).
7. Add schedule fields to the Workflow: Schedule and Timezone (selected based on the field from workflow table).
8. Add duration support fields to the Workflow Task activity.
9. Implement duration calculation script for the task activity.

The relative duration table and the DurationCalculator methods

The cmn_relative_duration table supports the definition of a due date as either a duration of time or a relative duration.

This table consists of two fields: "name" and "script." The "script" field contains the relative duration calculation script. This script includes the "calculator" variable, which is used to calculate the due date.

The DurationCalculator script include can be used to perform the duration calculations. The following are methods that are available in this script include.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>setSchedule(String schedID, [String timezone])</td>
<td>Sets the schedule and time zone to be used for calculating the due date.</td>
</tr>
<tr>
<td>setStartDateTime(GlideDateTime start)</td>
<td>Sets the start time for the duration calculations. If 'start' is blank, uses current date/time.</td>
</tr>
<tr>
<td>calcDuration(int seconds)</td>
<td>Calculates the end date and time. Upon completion the this.endDateTime and this.seconds properties will be set to indicate the results of the calculation.</td>
</tr>
<tr>
<td>calcRelativeDuration(String relativeDurationID)</td>
<td>Calculates the duration using the specified relative duration script. Upon completion the this.endDateTime and this.seconds properties will be set to indicate the results of the calculation.</td>
</tr>
</tbody>
</table>
## DurationCalculator script include table (continued)

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getEndDateTime()</td>
<td>Gets the this.endDateTime property that was set by calcDuration/calcRelativeDuration indicating the end date and time for the duration.</td>
</tr>
<tr>
<td>getSeconds()</td>
<td>Gets the this.seconds property that was set by calcDuration/calcRelativeDuration indicating the total number of seconds of work to be performed for the duration.</td>
</tr>
<tr>
<td>getTotalSeconds()</td>
<td>Gets the this.totalSeconds property that was set by calcDuration/calcRelativeDuration indicating the total number of seconds between the start and end times of the duration.</td>
</tr>
</tbody>
</table>

**Note:** This is the total work time, not the total time between start and end times and may be used to determine percentages of the work time.

## Relative duration script functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean isAfter(GlideDateTime dt, String time)</td>
<td>Is 'time' of day after the time of day specified by 'dt'? dt, if blank, uses current date/time. time is in &quot;hh:mm:ss&quot; in 24-hour format.</td>
</tr>
<tr>
<td>calcRelativeDueDate(GlideDateTime start, int days, String endTime)</td>
<td>Calculates the due date starting at 'start' and adding 'days' using the schedule and time zone. When we find the day that the work is due on, set the time to 'endTime' of that day. Upon completion, this.endDateTime and this.seconds properties will be set to indicate the results of the calculation.</td>
</tr>
</tbody>
</table>
Relative duration script functions (continued)

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>endTime is blank, use end of the ending work day.</td>
<td></td>
</tr>
</tbody>
</table>

JavaScript engine on the platform

The JavaScript engine that evaluates scripts supports the ECMAScript5 standard and is based on Rhino version 1.7 R5.

There are no plugins or properties needed to install the new JavaScript engine. Starting with the Helsinki release, all instances use the JavaScript engine that supports ECMAScript5.

The benefits include the following.

- You can use modern library code, such as lodash.js and moment.js.
- Your script will follow standard ECMAScript5 behavior.

What you need to know

The JavaScript engine provides an improved environment for developing scripts.

- Legacy code continues to work.
- Compatibility mode supports the legacy modifications to the old JavaScript engine.
- All scripts created prior to the Helsinki release and global scripts run in compatibility mode.
- New applications created in Helsinki and later releases default to run in ES5 standards mode.

For additional information on this topic, see this ServiceNow Community discussion on support for ECMAScript 2015 (ES6) or newer.

JavaScript modes

To support existing scripts and new scripts developed to the ECMAScript5 standard, the JavaScript engine has two modes. The modes are Compatibility Mode and ES5 Standards Mode. The JavaScript engine dynamically determines which mode to use on a script-by-script basis.
**Compatibility mode**

Compatibility mode is used for all scripts developed prior to the Helsinki release, and all global scripts. Compatibility mode has some differences from the old JavaScript engine.

JSON support changes.

- `JSON.stringify()` and `JSON.parse()` are now implemented using the ES5 Native JSON object.
- The new `JSON().encode()` and new `JSON().decode()` are still supported, but should only be used when the legacy behavior is required.

The use of 3rd party JavaScript libraries is not supported in Compatibility mode.

**ES5 Standards mode**

ES5 Standards mode is the default when you create new scoped scripts. This mode does not preserve the legacy behaviors in the pre-Helsinki JavaScript engine.

ES5 standards mode supports ECMAScript5 syntax and features, including the following.

- The `use strict` declaration
- Control over extensibility of objects
- Get and set properties on objects (accessors)
- Control over write-ability, configurability, and enumer-ability of object properties
- New Array and Date methods
- Native JSON support
- Support for modern third-party libraries such as lodash.js and moment.js

**Porting code to ES5 standards mode scripts**

ES5 standards mode catches errors that compatibility mode allows.

Things to watch for when porting code from existing scripts to new scoped scripts using ES5 standards mode.

ECMAScript5 evaluates the term `new Boolean(false)` to true. In compatibility mode, it evaluated to false.

ECMAScript5 throws an EcmaError when a non-existent property is referenced. In compatibility mode no error was thrown.
ECMAScript5 throws an EcmaError when a non-existent function is called. In compatibility mode, no error was thrown.

ECMAScript5 correctly handles new lines. In the past, a newline character after a comment was recognized, which is wrong. In this example, in compatibility mode, all three functions are called. In ECMAScript5, only the first function is called.

```javascript
var expr = doFoo(); // do foo
doBar(); // do bar
finish(); // all done
eval(expr);
```

ECMAScript5 correctly handles postfix increment and decrement. In this example, in compatibility mode, the variable `x` gets the incremented value, which is wrong.

```javascript
var x = gr.limit++;
```

**Debugging scripts**

Debug scripts using session logs and Now Platform debugging tools such as a walk-through script debugger and error messages that display in the UI.

**Debugging server-side scripts**

Use the Script Debugger and session logs to debug server-side code. For more information, see [Script Debugger and Session Log](#).

You can also use session debug to display error messages related to a server-side script that runs as a result of a client-side change. For more information, see [Session debug](#).

**Debugging client-side scripts**

Use session debug to display debugging messages in the user interface. For more information, see [Session debug](#). Use the session log to view logging information for script includes and custom UIs, such as Agent Workspace.

You can also debug client-side scripts using browser-based developers tools.

**Debugging applications and scopes**

Use the application debugging options to understand how a script's application scope might affect your application, table, or record. You may need to update cross-scope privileges to troubleshoot scope access issues. See [Debugging applications](#).
Script Debugger and Session Log

The Script Debugger enables users with the script_debugger role to debug server-side JavaScript, while the Session Log enables you to view and download required logs.

Users with the script_debugger role can perform these actions using Script Debugger:

- Have a dedicated debug transaction, which applies only to the current session.
- Set and remove breakpoints.
- Pause the current session at a breakpoint.
- Evaluate expressions during runtime.
- Step through code line-by-line.
- Step into and out of function and method calls.
- View the value of local and global variables.
- View the value of private variables from function closures.
- View the call stack.
- View the transaction that the system is processing.
- Turn off the script debugger to resume running paused scripts.

Use the Session Log tab to retrieve the session log for business rules, script includes, and a custom UI such as ServiceNow® Agent Workspace that has a GraphQL component. Users with the script_debugger role can:

- View session logs in a separate tab.
- Download a log.
- View logs for Agent Workspace.
- Specify debug options to view or download only the required logs.

By default, 100 transactions and 10,000 messages appear on the Session Log tab. If the transaction or message count exceeds the default value, the session log is cleared and the next transactions or messages appear. You can configure this transaction and message count using the `glide.debugger.log.transaction.count` and `glide.debugger.log_messages_limit` user preferences respectively. For more information about the `glide.debugger.log.transaction.count` and `glide.debugger.log_messages_limit` user preferences, see User preference settings.
Note: Enable Session Log as a separate tab with Script Debugger using the `glide.debugger.log.ui` system property.

- The Page option displays logs under forms and lists and on the Session Log tab.
- The Session option displays logs only on the Session Log tab.

For more information about the `glide.debugger.log.ui` system property, see Available system properties.

When you execute a statement in the Console, the executed statement is stored in the browser cache. You can use the up arrow key to get the previous statement and down arrow key to get the next statement from the browser cache. The user preference setting, `glide.debugger.console.cached_stmt_limit`, defines the number of statements cached in a browser session. The default statement cache value is 20 and the maximum value is 100. You can configure the statement cache value from user preferences.

Note: The cached statements are not available when the browser cache is cleared or when you log in from a different browser or a different computer.

The Script Debugger can pause any server-side script that runs in an interactive transaction such as business rules, script includes, script actions, or UI actions that require a response to proceed. If the GlideSystem method `isInteractive()` returns True when running the script in context, then the Script Debugger can pause it.

Note: Some script objects, such as script includes, can be called from multiple contexts. For example:

- when a business rule runs a script include on a form submit that is an interactive transaction waiting on the form data to change before continuing.
- when a scheduled job runs the same script include that is a non-interactive background transaction that can also run other scripts simultaneously.

To debug client-side scripts, you can use browser-based developers tools. A debugger transaction remains open as long as the user session is valid. If a user logs out or their session times out, the system closes the debugger transaction.

To view debug logs, see Display debugging logs.
Note: When the Script Debugger is enabled, code is executed in interpreted mode. If parts of the script are set to run in strict mode, the debugger is not able to find the correct objects and the debugger fails. The Script Debugger must run on scripts outside of strict mode.

Script Tracer and debugging scripts
The Script Tracer can help you filter your debugging search to quickly narrow down script problems. You can identify lines of scripts in the Glide record that have undergone change during execution. Finding those specific lines of scripts rather than doing a wide search helps save time and improves productivity.

Overview
Use the Script Tracer to narrow your search so you can debug scripts and business rules more efficiently. You can find the Script Tracer by searching in the left navigation pane.

Note: To use the Script Tracer, your role must be admin.

Once you enable Script Tracer and execute a UI transaction, the Tracer searches through all the scripts being executed. The following filters are available:

- **File type**: Search for a specific file type
- **Table**: Look in the specific table for the script being executed

The Script Tracer searches for changes in the script during execution and presents them in a list for you to examine. When you click **Start Tracer**, the Tracer begins searching for changes in the Glide record. You can click the **Debug Script** button at any time to see the script itself.
Use the tabs to see specific information from the Tracer:

- **State**: Displays the difference in the old and new scripts
  - By default, the **Show only changed values** check box is enabled, so you can avoid fields that have not changed.
  - To view all the fields (changed or not), you can clear that check box.

  **Note**: If the file is not reflected in the trace statement, it means the changes in the Glide record is not recognized by the system.
  If there are any errors, they display at the top of the State tab, with their line numbers and error message displayed in order of occurrence.

  **Script**: Displays the line of changed scripts that the Glide record has undergone during execution. You can view the entire line of script by clicking the **Show Script** button.

  **Transaction**: Shows all transaction records of the trace

  **Debug Script**: Opens the script in Debugger to debug the script

  **View File**: Opens the script in the ServiceNow platform for editing

  **Clear trace**: Clears the trace when you are finished.
Limiting the tracer

You may want to set a limit for your trace so that you don't generate too many returns. By default there will be up to 1,000 lines of script traced. Once this number is reached you must clear the trace and start tracing again. If you want to change the maximum number of lines for tracing you can configure your limit using the property `glide.debug.trace.trace_line_limit`.

Since each trace you run is new, make sure you're finished reading the results of one trace before clearing it and beginning another one.

To learn more, see Debugging scripts.

Parts of the user interface

The Script Debugger interface displays information about breakpoints set, the call stack and line number of the currently executing script line, details about variables and transactions, and status of console.
# Parts of the Script Debugger

<table>
<thead>
<tr>
<th>User Interface Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Breakpoints</strong></td>
<td>Displays a list of the breakpoints set by script type, script name, and line number. The debugger updates this list as you add and remove breakpoints.</td>
</tr>
<tr>
<td><strong>Call stack</strong></td>
<td>Displays a list of script calls that preceded or invoked the current line number. This information is only visible when the debugger pauses on a breakpoint.</td>
</tr>
<tr>
<td><strong>Transaction details</strong></td>
<td>Displays information about the current transaction. This information is only visible when the debugger pauses on a breakpoint.</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>Displays if the debugger is waiting for a breakpoint, paused on a breakpoint, or has encountered an exception.</td>
</tr>
<tr>
<td>User interface element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>User</td>
<td>Displays the name of the user who is running the current debugger session.</td>
</tr>
<tr>
<td>Coding pane header</td>
<td>Displays the script type and name of the script in the coding pane.</td>
</tr>
<tr>
<td>Breakpoint icon</td>
<td>Indicates the line number where the debugger pauses when evaluating the current script.</td>
</tr>
<tr>
<td>Pause debugging button</td>
<td>Stops any current debugging session, and disables the Script Debugger for the current user. The Script Debugger doesn't pause on breakpoints for the current user until it's restarted.</td>
</tr>
<tr>
<td>Console</td>
<td>Displays a command line interface used for evaluating expressions during runtime. The console is available only when the script execution is paused.</td>
</tr>
<tr>
<td>Resume script execution button</td>
<td>Advances from the current breakpoint to the next breakpoint. If there are no other breakpoints, the script runs to completion.</td>
</tr>
<tr>
<td>Step over next function call button</td>
<td>Advances past the method that's about to be called, executing the method as a single step.</td>
</tr>
<tr>
<td>Step into next function call</td>
<td>Advances to the first line of executed code within a method call. Stepping into a method updates the current position within the call stack. If the user doesn't have read access to the method call, then this control acts like step over instead.</td>
</tr>
<tr>
<td>Step out of current function</td>
<td>Exits from current method call and returns to the calling script from the call stack. If the user isn’t within a method call, then this control acts like step over instead.</td>
</tr>
</tbody>
</table>
### Parts of the Script Debugger (continued)

<table>
<thead>
<tr>
<th>User Interface Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Local</strong></td>
<td>Displays a list of local scope JavaScript variable names and their values. This information is only visible when the debugger pauses on a breakpoint.</td>
</tr>
<tr>
<td><strong>Closures</strong></td>
<td>Displays a list of global scope JavaScript variable names and their values set by function closure. This information is only visible when the debugger pauses on a breakpoint.</td>
</tr>
<tr>
<td><strong>Global</strong></td>
<td>Displays a list of global scope JavaScript variable names and their values. This information is only visible when the debugger pauses on a breakpoint.</td>
</tr>
</tbody>
</table>

### Session Log

![Session Log](image)
Session Log user interface elements

<table>
<thead>
<tr>
<th>User Interface element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transactions</td>
<td>Transaction ID. Displays information about the current transaction.</td>
</tr>
<tr>
<td>Filter for log text</td>
<td>Field to enter text to filter the logs that contain a specific text.</td>
</tr>
<tr>
<td>Debug Output</td>
<td>Option to filter logs based on the dynamically loaded debug output types. For example, Security Rule.</td>
</tr>
<tr>
<td>Apps</td>
<td>Option to filter logs based on the dynamically loaded apps. For example, Service Management Integrations.</td>
</tr>
<tr>
<td>Message Type</td>
<td>Option to filter logs based on the dynamically log levels. For example, Info.</td>
</tr>
<tr>
<td>Clear log</td>
<td>Clears all logs.</td>
</tr>
<tr>
<td>Download log</td>
<td>Download the logs in HTML file format.</td>
</tr>
<tr>
<td>Settings</td>
<td>Session debug options. For information about the debug options, see Session debug.</td>
</tr>
</tbody>
</table>

Script Debugger step-through and console controls

After the Script Debugger pauses a script, use the step-through controls to move between script lines and move between scripts in the call stack. Use the Console controls to expand console, collapse console, clear console, and rerun expressions.

Step-through controls

<table>
<thead>
<tr>
<th>Control</th>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stop debugging</td>
<td></td>
<td>Stops any current debugging session, and disables the Script Debugger for the current user. The Script Debugger does not pause on breakpoints for the current user until it is restarted.</td>
</tr>
<tr>
<td>SHIFT+F2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Step-through controls (continued)

<table>
<thead>
<tr>
<th>Control</th>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start debugger - F2</td>
<td>✪</td>
<td>Enables the Script Debugger for the current user. The Script Debugger pauses on breakpoints.</td>
</tr>
<tr>
<td>Resume script execution - F9</td>
<td>▶</td>
<td>Advances from the current breakpoint to the next breakpoint. If there are no other breakpoints, the script runs to completion.</td>
</tr>
<tr>
<td>Step over next function call - OPTION+F9</td>
<td>♻</td>
<td>Advances to the next evaluated line of script based on current conditions. The Script Debugger skips any lines of code that do not need to run because their conditions are not met. For example, when the condition of an <code>if</code> statement is not true, the debugger skips the code block for the condition.</td>
</tr>
<tr>
<td>Step into next function call - OPTION+F10</td>
<td>↓</td>
<td>When the Script Debugger pauses on a method call, this control allows the user to advance to the first line of executed code within the method call. Stepping into a method updates the current position within the call stack. If the user does not have read access to the method call, then this control acts like step over instead.</td>
</tr>
<tr>
<td>Step out of current function - OPTION+F11</td>
<td>↑</td>
<td>When the Script Debugger pauses within a method call, this control allows the user to exit the current method call and return to the calling script from the call stack. If the user is not within a method call, then this control acts like step over instead.</td>
</tr>
</tbody>
</table>

### Console controls

<table>
<thead>
<tr>
<th>Control</th>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Console</td>
<td>🡁</td>
<td>Expands the Console.</td>
</tr>
<tr>
<td>Close Console</td>
<td>🡡</td>
<td>Collapses the Console.</td>
</tr>
<tr>
<td>Clear expressions</td>
<td>🔄</td>
<td>Clears all the expressions in the Console.</td>
</tr>
</tbody>
</table>
Console controls (continued)

<table>
<thead>
<tr>
<th>Control</th>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Re-execute expression</td>
<td>🔄</td>
<td>Re-executes the expression which is already executed.</td>
</tr>
</tbody>
</table>

Related information

Set or remove breakpoints

Evaluate expressions in runtime using Console

Define, declare, and verify new variables and functions while you debug a script in runtime using Console. The script execution must be paused in order to use Console.

Before you begin

- Review Limitations with using Console
- Role required: script_debugger, admin

Procedure

1. Launch Script Debugger in one of the following ways:

<table>
<thead>
<tr>
<th>Application</th>
<th>Navigation path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application navigator</td>
<td>Navigate to System Diagnostics &gt; Script Debugger.</td>
</tr>
<tr>
<td>Studio</td>
<td>Navigate to File &gt; Launch Script Debugger.</td>
</tr>
<tr>
<td>Syntax Editor</td>
<td>Click the Script Debugger icon 🔄.</td>
</tr>
</tbody>
</table>

The Script Debugger modal is displayed.

2. Trigger the script.
   For example, create a record to trigger an insert business rule script.
   The Script Debugger pauses the script on the first line that contains a breakpoint, and then you see the ServiceNow Script Debugger confirmation window.
3. Click **Start Debugging**. The focus shifts to the Script Debugger window and you see the target script that paused at the first breakpoint.

> **Note:** Make sure that the status of Script Debugger is **EXECUTION_PAUSED**. You can use Console only when the script execution is paused during debugging.

4. Click the **Open Console** icon to expand the Console pane. To start evaluating expressions, enter one or more expressions in the Console and press Enter. For example, enter `var x = 10;` and press Enter. To enter multiple lines of expressions, press Shift + Enter after each line and press Enter after the last expression. To clear all the expressions in the Console, click the clear console icon.

For more information on Console controls, see **Script Debugger step-through and console controls**.

<table>
<thead>
<tr>
<th>Expressions in the Console</th>
<th>Output in the Console</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>var x = 10;</code></td>
<td><code>x = 10</code></td>
</tr>
<tr>
<td><code>console.log(x);</code></td>
<td><code>10</code></td>
</tr>
<tr>
<td><code>var y = x + 5;</code></td>
<td><code>y = 15</code></td>
</tr>
<tr>
<td><code>console.log(y);</code></td>
<td><code>15</code></td>
</tr>
<tr>
<td><code>var z = y * 2;</code></td>
<td><code>z = 30</code></td>
</tr>
<tr>
<td><code>console.log(z);</code></td>
<td><code>30</code></td>
</tr>
</tbody>
</table>
After a statement is executed, it is stored in the browser cache. You can use the up arrow key to get the previous statement and down arrow key to get the next statement from the browser cache. You can configure the number of cached statements for a session from user preferences. For more information about user preferences settings, see Script Debugger and Session Log.

Related reference

Limitations with using Console

Limitations with using Console
You need to be aware of a few limitations when you use Console to evaluate expressions while debugging a script in runtime.

• The properties and values of an object don’t display in Console. When you try to display an object to Console, only the string value of the object appears.
• Console doesn’t support GlideSystem printing methods, such as info() and print().
• You can’t use the `this` keyword in Console.
• A script debugger timeout occurs when you evaluate expressions in Console.
• While executing long scripts, if you see the response Awaiting response from server, you can’t resume debugging or stop debugging using the resume or stop controls.

Launch the Script Debugger
Developers can launch the Script Debugger from the application navigator, Studio, or from the syntax editor.

Before you begin
Role required:
• admin
• script_debugger

Procedure
Select a path based on your starting point:

<table>
<thead>
<tr>
<th>Starting point</th>
<th>Navigation path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application navigator</td>
<td>Navigate to System Diagnostics &gt; Script Debugger.</td>
</tr>
</tbody>
</table>
The system opens the Script Debugger in a new window.

### Set or remove breakpoints

Set breakpoints or conditional breakpoints to pause scripts at specific lines, and remove breakpoints when you are done debugging them.

**Before you begin**

Role required:

- admin
- script_debugger

**About this task**

Breakpoints belong to the developer who sets them. Developers must set and remove their own breakpoints.

❓ **Note:** At a specific line, you can set either a logpoint or breakpoint but not both.

**Procedure**

1. Navigate to the server script to debug. For example, navigate to **System Definition** > **Business Rules**.

2. From the Syntax Editor, click the gutter next to a script line.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Set a breakpoint</strong></td>
<td>Click a blank line to set a breakpoint.</td>
</tr>
<tr>
<td><strong>Set a conditional breakpoint</strong></td>
<td>Right-click a blank line and click <strong>Add conditional breakpoint</strong> to set a conditional breakpoint.</td>
</tr>
<tr>
<td><strong>Remove a breakpoint</strong></td>
<td>Click a breakpoint to remove it.</td>
</tr>
<tr>
<td><strong>Remove a conditional breakpoint</strong></td>
<td>Right-click a conditional breakpoint and select <strong>Remove breakpoint</strong> to remove it.</td>
</tr>
</tbody>
</table>
3. From the Syntax Editor toolbar, click the **Open Script Debugger** icon. The system opens a Script Debugger window.

4. Trigger the script. For example, create a record to trigger an insert business rule script. The Script Debugger pauses the script on the first line containing a breakpoint, and the system displays a confirmation window.

5. Click **Start Debugging**. The system switches focus to the Script Debugger window and displays the target script paused at the first breakpoint. Console pane is enabled.

6. When debugging is complete, remove breakpoints from the script.

**Related reference**

- **Script Debugger step-through and console controls**
- **Set or remove logpoints**

Set breakpoints or conditional logpoints to log messages to the console at specific lines, and remove logpoints when you are done debugging them.

**Before you begin**

- Set the `glide.debug.log_point` system property to `true`. See Available system properties for more information.
- Role required: admin or script_debugger

**About this task**

Logpoints belong to the developer who sets them. Developers must set and remove their own logpoints.

**Note:** At a specific line, you can set either a logpoint or breakpoint but not both.
Procedure

1. Navigate to the server script to debug. For example, navigate to System Definition > Business Rules.
2. From the Syntax Editor, click the gutter next to a script line.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set a logpoint</td>
<td>Right-click a blank line and click Add logpoint to set a logpoint.</td>
</tr>
<tr>
<td>Remove a logpoint</td>
<td>Right-click a logpoint and select Remove logpoint to remove it.</td>
</tr>
<tr>
<td>Edit a logpoint</td>
<td>Right-click a logpoint and select Edit logpoint to edit it.</td>
</tr>
</tbody>
</table>

ℹ️ Note: The script entered for the logpoint must have the same format as that of the script in GSLog and GSInfo script includes.

3. From the Syntax Editor toolbar, click the Open Script Debugger icon 🚀.
4. On the Script Debugger window, trigger the script. For example, create a record to trigger an insert business rule script.

ℹ️ Note: You can also add logpoints in the Script Debugger window.
5. In the Script Debugger window, click Session Log to view the logpoints.
6. When debugging is complete, remove logpoints from the script.

Script Debugger status

The Script Debugger status determines what debugging actions are available and what information it can display.

The Script Debugger displays its status at the bottom left of the user interface.

Sample Script Debugger status

ℹ️ Status: EXECUTION_PAUSED
<table>
<thead>
<tr>
<th>Status</th>
<th>Occurs when</th>
<th>Description</th>
<th>Actions available</th>
</tr>
</thead>
<tbody>
<tr>
<td>WAITING FOR FIRST BREAKPOINT</td>
<td>- The user opens a Script Debugger window or tab.</td>
<td>The Script Debugger is ready to pause script and display debugging information.</td>
<td>- Pause script at the first breakpoint in the call stack.</td>
</tr>
<tr>
<td>EXECUTION_PAUSED</td>
<td>- The Script Debugger pauses on a breakpoint.</td>
<td>The Script Debugger has paused on a line of code, and the user can debug the script. Console is enabled.</td>
<td>- Resume processing until the Script Debugger reaches the next breakpoint.</td>
</tr>
<tr>
<td></td>
<td>- The user steps over, steps into, or steps out to the next line of evaluated code.</td>
<td></td>
<td>- Step through a script.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Display the call stack.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Display transaction information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Display variable values.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Evaluate expressions in Console during runtime.</td>
</tr>
<tr>
<td>WAITING FOR BREAKPOINT</td>
<td>- The user resumes processing until the Script Debugger reaches the next breakpoint.</td>
<td>The Script Debugger is searching for the next line of code at which to pause. Users will typically never see this status because the Script Debugger changes status.</td>
<td>- Pause script at the next breakpoint.</td>
</tr>
<tr>
<td></td>
<td>- The user steps through a script until the Script Debugger</td>
<td></td>
<td>- Pause script at the next script line requiring evaluation.</td>
</tr>
</tbody>
</table>
Possible Script Debugger status values (continued)

<table>
<thead>
<tr>
<th>Status</th>
<th>Occurs when</th>
<th>Description</th>
<th>Actions available</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>reaches the next line of code to evaluate or the transaction completes.</td>
<td>after it locates the next breakpoint or script line to evaluate.</td>
<td></td>
</tr>
</tbody>
</table>
| OFF    | • The user pauses the Script Debugger.  
        • The user closes the Script Debugger window or tab.  
        • The user session ends for any reason.  
        • The administrator resets all Script Debugger instances by navigating to the debugger_reset.do page. | The Script Debugger is inactive and does not pause scripts or display debugging information. | • Start the Script Debugger.  
        • Open a Script Debugger window or tab. |

Log entries
Every time a debug transaction finishes executing, the system creates a log entry for it with a DEBUGGED prefix. For example:

```
2016-08-15 15:57:32 (197) Default-thread-3 900F510167112200C4098C7942415A75 *** End  
#39, path: /my-app.do, user: admin, DEBUGGED total transaction time: 0:00:11.010,  
transaction processing time: 0:00:11.010, network: 0:00:00.000, chars: 6,058, uncompressed  
chars: 20,731, SQL time: 50 (count: 34), business rule: 0 (count: 0), phase 1  
form length 56,464, largest chunk written: 10,428, request parms size: 40, largest input  
read: 0
```
Transaction details

The Script Debugger displays transaction details for the current paused user session.

Transaction details are available in a dedicated resizeable section underneath the Call Stack on the bottom left of the Script Debugger.

The Script Debugger only displays transaction details when it pauses on a script. Developers can use transaction details to:

- Inspect the URL of the currently paused transaction.
- Inspect the request parameters for the currently paused transaction.
- Inspect network information about the current transaction.
- Inspect the user and session ID that initiated the debug transaction.

Related reference

Available transaction details

The Script Debugger provides a standard set of transaction details for developers to debug and troubleshoot scripts.
### Available transaction details

<table>
<thead>
<tr>
<th>Transaction element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>The URL of the currently paused transaction.</td>
</tr>
<tr>
<td>Request parameters</td>
<td>The list of request parameters for this transaction. Each transaction has its own list of request parameters, but record transactions typically include the field values used to insert, update, or delete a record.</td>
</tr>
<tr>
<td>instance</td>
<td>The instance name.</td>
</tr>
<tr>
<td>address</td>
<td>The IP address of the end-user client system.</td>
</tr>
<tr>
<td>session</td>
<td>The user session ID.</td>
</tr>
<tr>
<td>forward</td>
<td>The IP address of the load balancer.</td>
</tr>
<tr>
<td>query count</td>
<td>The number of database queries the Script Debugger has made.</td>
</tr>
<tr>
<td>thread</td>
<td>The name of the thread running the Script Debugger instance.</td>
</tr>
<tr>
<td>transactionid</td>
<td>The Sys ID of the current transaction.</td>
</tr>
<tr>
<td>token</td>
<td>The Script Debugger token of the currently paused transaction. The system uses this token to identify different Script Debugger instances.</td>
</tr>
<tr>
<td>name</td>
<td>The name of the currently paused transaction. You can use this name to identify transactions in the logs.</td>
</tr>
<tr>
<td>processor</td>
<td>The name of the processor processing the current transaction, if present.</td>
</tr>
<tr>
<td>method</td>
<td>The HTTP request method the currently paused transaction uses.</td>
</tr>
<tr>
<td>startTime</td>
<td>The date-time stamp when the Script Debugger instance started.</td>
</tr>
<tr>
<td>page</td>
<td>The current table or UI page associated with the transaction.</td>
</tr>
<tr>
<td>user</td>
<td>The user who triggered the debug transaction.</td>
</tr>
<tr>
<td>nodeid</td>
<td>The Sys ID of the node running the Script Debugger instance.</td>
</tr>
</tbody>
</table>

### Related information

**Transaction details**
Script Debugger multiple developer support

The Script Debugger allows multiple developers to debug their own transactions without affecting each other.

The Script Debugger only allows developers to see and interact with items related to their current debugging session such as:

- Breakpoints
- Call stack
- Console
- Transactions
- Status

The Script Debugger prevents one developer from seeing or modifying another debug session. Administrators, however, can impersonate another user, open the Script Debugger, and debug transactions generated by the impersonated user.

The Script Debugger displays the debug session user at the bottom left of the user interface.

Sample Script Debugger user

User: System Administrator

Concurrent Script Debugger usage

By default, the system supports debugging \[ (\text{The number of semaphores on the instance}) / 4 \] concurrent transactions. Administrators can specify the number of concurrent transactions the system can debug by setting the `glide.debugger.config.max_node_concurrency` system property. The system can debug up to \[ (\text{The number of semaphores on the instance}) - 2 \] concurrent transactions.

Administration of debugging sessions

Debugging sessions can remain actively debugging (in the EXECUTION_PAUSED or WAITING_FOR_BREAKPOINT statuses) until:

- The user pauses the Script Debugger.
- The user closes the Script Debugger.
- The user session ends.

Administrators can view the currently running debugger sessions by navigating to the page `xmlstats.do`. 
Administrators can stop all currently running debugging sessions by navigating to the page `debugger_reset.do`. Only users with the admin role can access this page.

**Related information**

**Script Debugger impersonation support**

You can use the Script Debugger while impersonating another user, but only if the impersonated user has the script_debugger role and has read access to the target script.

While impersonating another user, you can:

- See and change breakpoints that belong to the impersonated user.
- View and pause on scripts that the impersonated user has read access to.
- Evaluate expressions in Console on behalf of the impersonated user.

The Script Debugger step-through controls also use the read access of the impersonated user. For example, if the impersonated user does not have read access to a function in the call stack, any **Step into** action instead becomes a **Step over** action.

The impersonated debugging session lasts until:

- You stop impersonating the user.
- You log out or the user session ends.
- You pause the Script Debugger.
- You close the Script Debugger.

**Related information**

**Script Debugger multiple developer support**

**Script Debugger Scripts - Background support**

The Scripts - Background module does not support setting breakpoints directly in the script field. You can however, set breakpoints in the script objects called or triggered by the Scripts - Background module.

While running arbitrary JavaScript code in the **Scripts - Background** module, the Script Debugger can only pause scripts when you:
• Call a script include containing breakpoints.
• Trigger a business rule containing breakpoints.
• Trigger a script action containing breakpoints.

Domain separation and Script Debugger

Domain separation is supported in Script Debugger. Domain separation enables you to separate data, processes, and administrative tasks into logical groupings called domains. You can control several aspects of this separation, including which users can see and access data.

Support level: Basic

• Business logic: Ensure that data goes into the proper domain for the application’s service provider use cases.
• The application supports domain separation at run time. The domain separation includes separation from the user interface, cache keys, reporting, rollups, and aggregations.
• The owner of the instance must set up the application to function across multiple tenants.

Sample use case: When a service provider (SP) uses chat to respond to a tenant-customer’s message, the client must be able to see the SP’s response.

For more information on support levels, see Application support for domain separation.

How domain separation works in Script Debugger

Script Debugger is not a full application but rather, a feature in the Platform suite, meaning it works alongside other features, including domain separation.

Related information

Domain separation for service providers

Session debug

Enable session debugging to display debugging messages in the user interface.

You can enable all areas for abundant logging on the bottom of each page load, or you can enable each module one by one, for more specific information about what is happening during this session, and specifically, for the previous transaction. Select session debug options under System Diagnostics > Session
Debug. When enabled, session debugging is active during the user session or until disabled. To view debug logs, see Display debugging logs.

The system provides the following session debugging options.

<table>
<thead>
<tr>
<th>Debug option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable All</td>
<td>Displays all available debugging messages.</td>
</tr>
<tr>
<td>Disable All</td>
<td>Stops displaying all debugging messages.</td>
</tr>
<tr>
<td>Debug Business Rule</td>
<td>Displays debugging messages for business rules. If there are business rules from multiple applications affecting a table or record, the system displays which application the business rule comes from.</td>
</tr>
<tr>
<td>Debug Upgrade</td>
<td>Displays detailed information logged for records processed during the last family-to-family or patch version upgrade session. See Debug Upgrade.</td>
</tr>
<tr>
<td>Debug Business Rule (Details)</td>
<td>Displays debugging messages for business rules and any changes made by business rules. If there are business rules from multiple applications affecting a table or record, the system displays which application the business rule comes from.</td>
</tr>
<tr>
<td>Debug Log</td>
<td>Displays all log entries.</td>
</tr>
<tr>
<td>Debug Date/Time</td>
<td>Displays Date/Time failures when inputs do not match required formats.</td>
</tr>
<tr>
<td>Debug SQL</td>
<td>Displays debugging messages for SQL queries.</td>
</tr>
<tr>
<td>Debug SQL (Detailed)</td>
<td>Displays debugging messages for SQL statements and any changes made by SQL statements.</td>
</tr>
<tr>
<td>Debug Security</td>
<td>Displays debugging messages for access controls. If there are access controls from multiple applications affecting a table or record, the system displays which application the access controls comes from.</td>
</tr>
<tr>
<td>Debug Escalations</td>
<td>Displays debugging messages for SLA and SLO escalations.</td>
</tr>
<tr>
<td>Debug AI Search</td>
<td>Displays debugging messages for AI Search.</td>
</tr>
</tbody>
</table>
### Session debug options (continued)

<table>
<thead>
<tr>
<th>Debug option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debug Metric Statistics</td>
<td>Displays an aggregate view of performance data (slow transactions, scripts, queries, events, and mutexes). These aggregate metrics are sorted by transaction, to help identify items that affect page performance.</td>
</tr>
<tr>
<td>Debug Text Search</td>
<td>Displays debugging messages for search result relevance and indexing.</td>
</tr>
<tr>
<td>Debug UI Policies</td>
<td>Displays debugging messages for UI policies.</td>
</tr>
<tr>
<td>Disable UI Policies Debug</td>
<td>Stops displaying debugging messages for UI policies.</td>
</tr>
<tr>
<td>Debug UI Macro</td>
<td>Displays the start and end of the UI Macro in the DOM as HTML comments. The comments consist of table name and UI macro name.</td>
</tr>
<tr>
<td>Disable Debug UI Macro</td>
<td>Stops displaying the start and end of the UI Macro in the DOM as HTML comments.</td>
</tr>
<tr>
<td>Debug Data Policies</td>
<td>Displays debugging messages for data policies.</td>
</tr>
<tr>
<td>Debug Quotas</td>
<td>Displays debugging messages for transaction quotas.</td>
</tr>
<tr>
<td>Debug Homepage Render</td>
<td>Displays debugging messages for homepages.</td>
</tr>
<tr>
<td>Debug Scopes</td>
<td>Displays debugging messages for entering or exiting application scopes when running script objects.</td>
</tr>
</tbody>
</table>

### Display debugging logs

Display session debug logs to help diagnose script and application problems.

**Before you begin**

Role required: none
Procedure

1. Navigate to System Diagnostics > Script Debugger > Session Debug and click Enable All.
2. Under Session Debug, click Debug Log. The Debug log displays.

Debugging applications

Application developers can display debug messages about configuration records to help them troubleshoot issues. The Debug Scopes module provides information about the system switching between custom applications to run server-side scripts.

The system offers the following debugging options to help application developers determine how applications affect configuration records.

<table>
<thead>
<tr>
<th>Application debug options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debugging option</td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>Debug Business Rule</td>
</tr>
<tr>
<td>Debug Business Rule (Details)</td>
</tr>
<tr>
<td>Debug Security</td>
</tr>
<tr>
<td>Debug Scopes</td>
</tr>
<tr>
<td>Enable Session Debug</td>
</tr>
</tbody>
</table>

When multiple applications contribute to the debug output, the system adds a new section called Apps to the display a list of the applications writing to the
session log. Clicking on the check box next to the application name hides or displays the application’s associated debug messages.

**Sample application debug output of business rules**

Debugging scopes

Application developers can use the **Debug Scopes** module to display information about when the system switches between custom applications to run server-side scripts.

When enabled, the system displays a message whenever the system switches to a custom application to run a server-side script.

**Sample debug scopes output from the incident table**

Every time the system runs a server-side script object it enters the script's scope context. When the script finishes running, the script exits the scope context. The debugging messages track changes to the script scope context.

The debugging message displays a greater than character > each time the system enters a script object's context, and displays a less than character < every time the system exits a script object's context. In cases where one script calls another the debugging message adds another greater than character to the path for each call. For example, if a business rule calls a script include, which in turn calls another script object there would three characters in the path such as:
> Entering scope [x_app_one]
>> Entering scope [x_app_two]
>> Entering scope [x_app_three]

> Note: The system does not display entering or exiting messages for script objects in the global scope.

Application developers may want to enable other debugging options to in conjunction with this option to see information about the possible source of the server-side script such as Debug Business Rule.

**Debugging business rules**

Debugging business rules can be achieved with resources available in the ServiceNow product.

**1. Tools**

The first step in the process is to identify tools which will help you figure out what's wrong.

### Debugging tools

<table>
<thead>
<tr>
<th>Debugging tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Dictionary</td>
<td>Navigate to System Definition &gt; Dictionary. The dictionary provides a list of all tables within your instance and can be invaluable when trying to locate information.</td>
</tr>
<tr>
<td>System Log</td>
<td>Navigate to System Logs &gt; System Log. You can place alert statements in your business rule which can write information to the log.</td>
</tr>
<tr>
<td>Debug Business Rule (Details)</td>
<td>Navigate to System Diagnostics &gt; Session Debug &gt; Debug Business Rule (Details). This debugging module displays the results business rules. Use this module to see if conditions are being met and values are being set as expected.</td>
</tr>
<tr>
<td>Alert Messages</td>
<td>There are several system functions that allow you to print messages to the page, the field or the log file. See Scripting alert, info, and error messages.</td>
</tr>
<tr>
<td>Business Rule Examples</td>
<td>Sometimes you can find what you're looking for in scripts others have written, including business rule error messages, or by building an OR query.</td>
</tr>
</tbody>
</table>
Debugging tools (continued)

<table>
<thead>
<tr>
<th>Debugging tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideRecord Information</td>
<td>This is the basic syntax used to query the database for information. See Querying tables in script. GlideRecord also includes aggregation support.</td>
</tr>
</tbody>
</table>

2. Variables

The next step is to gain some insight into the behavior of your business rule. For every action except an insert, you will more than likely use a query to get your record(s).

```javascript
var rec = new GlideRecord('incident');
rec.addQuery('active',true);
rec.query();
while (rec.next()) {
  gs.print(rec.number + ' exists');
}
```

To verify whether your query is actually returning records you can use `gs.addInfoMessage` to display information at the top of the screen.

```javascript
var rec = new GlideRecord('incident');
rec.addQuery('active',true);
rec.query();
gs.addInfoMessage("This is rec.next: " + rec.next());
while (rec.next()) {
  gs.print(rec.number + ' exists');
}
```

If your query returns no records you see the following:

This is rec.next: false

Use this technique to verify every variable within your business rule contains expected values.

⚠️ Tip: If necessary, break your script down into individual pieces and verify each piece works separate from the whole and then put them all back together one step at a time.

3. Locating information

The last step is to make sure you know where to find the information your rule is looking for.
In the ServiceNow application, one table can extend another table. This means when searching for information, you might need to query the parent table for the extended table’s sys_id to find what you seek.

A good example is the sc_task table, which extends the task table. The script below queries the extended table (sc_task) for the current sys_id and then query the parent table (task) for records with the matching sys_id, and then prints out the work notes field.

```javascript
var kids = new GlideRecord('sc_task');
kids.query();
go.addInfoMessage("This is requested item number: " + current.number);
go.print("This is the requested item number: " + current.number);

while (kids.next()) {
    var parents = new GlideRecord('task');
    parents.addQuery('sys_id', '=', kids.sys_id);
    parents.query();

    while(parents.next()) {
        go.addInfoMessage("This is task number: " + parents.number);
        go.print("This is task number: " + parents.number);
        go.addInfoMessage("These are the work notes: " + parents.work_notes);
        go.print("These are the work notes: " + parents.work_notes);
    }
}
```

**Debugging classifications**

You must add a system property to enable classification debugging.

**Debugging classifications**

The resulting log entries list the name of each classifier that runs, along with all the names and values that are available to the criteria in the classifier. To log debugging information about classifications, add the following system property.

<table>
<thead>
<tr>
<th>System Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>glide.discovery.debug.ci_identification</td>
<td>Enables debugging information for process classification.</td>
</tr>
</tbody>
</table>
### Field watcher

The field watcher tool tracks and displays all actions that the system performs on a selected form field.

Administrators can use the field watcher to figure out what happens to the field and how the value of the field changes when an event such as the firing of a business rule or enforcement of a data policy, takes place. Administrators can also impersonate non-admin users to debug what happens when those users make changes on an instance. Only one field can be watched at a time. Non-admin users with the impersonator role have access to the field watcher feature.

### How the field watcher works

The Field Watcher tool logs activity when any of the following events occur on a field:

- The default value is set on the field.
- User access rights for the field change due to an ACL or dictionary setting.
- A data policy prevents the value from being set.
- A reference qualifier query of the field value executes.
- A UI policy changes a field to or from read-only, visible, mandatory, or editable.
- A dependent value in another field restricts field choices.
- The value of the field is set or changed based on:
  - Assignment rules
  - Actions from an engine, such as the workflow engine
  - Business rules
  - User entries
  - Client scripts
  - UI actions
Note: The field watcher works only on form fields. It cannot be used on list fields. Also, field watcher is not available on password-protected fields or encrypted fields. Field watcher is only available within the UI frame. The option to watch a field does not appear in the context menu if you open a record outside of the UI frame, for example, in a new tab.

Use field watcher

Access field-level debugging information using the field watcher.

Procedure

1. Navigate to the form for which you want to view field-level debugging information.

2. Activate field watcher by right-clicking any field label on a form and select Watch - '<field name>'.

   The debug icon (工期) appears next to the field label. From this point on, the field watcher records every action taken on the selected field. For example, if you are watching a Priority field, if the priority is changed from Moderate to Low and the record is updated, the field watcher will display information about that change.

3. View the field watcher log by clicking the debug icon.
   A new pane opens at the bottom of the screen, showing a field watcher tab. It may also show tabs for JavaScript Logging and JavaScript Debugger.

4. Click the Field Watcher tab, if needed.

5. Stop watching a field by right-clicking the field and selecting Unwatch - '<field name>'. To watch another field, right-click that field and select Watch - '<field name>'.
6. Clear the field watcher log by clicking the clear log button ( ).

7. Resize the field watcher pane by dragging the splitter bar up or down. Dragging the splitter bar to the bottom of the screen closes the field watcher pane. Reopen the pane by clicking the debug icon again.

Field watcher tab details

The field watcher displays field information and configuration options. The left-side of the Field Watcher tab shows the following information for the watched field.

- **Table**: table to which the field belongs.
- **Element**: field label.
- **Type**: type of data stored in the field.
- **Dependent**: field on which the current field depends.
- **Reference**: table from which the field’s value originates, if applicable.
- **Reference Qual**: reference qualifiers that may be restricting data on the field.
- **Attributes**: attributes on the field as specified in the dictionary entry for that field.

On the right-side of the Field Watcher tab, select the types of activity information you want to see for the selected field. Clear the check box for any type of information that is not needed.

Watching a hidden field

Administrators may need to watch a hidden field.

Procedure

1. Use the dictionary to determine the column name of the field.
2. Elevate privileges to the security_admin role.
3. Navigate to **System Definition > Scripts Background**.
4. In **Run script (JavaScript executed on server)**, enter the following command:

   ```javascript
gs.getSession (). setWatchField ( "hidden_field" ) ;
   ```

   Replace hidden_field with the column name of the hidden field.
5. Navigate to the form containing the missing field.

   The Field Watcher tab output contains information about the hidden field.
Viewing information for the watched field

When information for a watched field is changed and the record is updated, the field watcher tab displays relevant information at the bottom.

Field watcher viewing data

Field watcher information includes:

- **Timestamp**: time the field was changed using the HH:MM:SS (ms) format.
  - Orange text: server-side changes, such as ACLs.
  - Blue text: client-side changes, such as client scripts.

- **Type of object that changed the field and its associated name**: The type of item that changed on the field; for example, **CLIENT SCRIPT**, **BUSINESS RULE**, or **ACL**. In the case of scripts, business rules, or other configuration-type fields, field watcher displays the name of the script or business rule that changed the field, if any. Click the name to go directly to the record for that item.

- **Old and new values**: The old and new values for the field, if the value changed. Field watcher does not record the value if it was inserted in the form by default at the time the record was created.

- **Additional information**: Call tracing information, such as the name of the script engine or workflow that changed the field. Click the plus icon to expand the selection.
  - Orange text: Indicates server-side activity.
  - Blue text: Indicates client-side activity.

**Example: Watching the incident priority**

The following example shows what happens to the **Priority** field on the incident form when both the **Impact** and **Urgency** fields change.

The Incident form has two client-side data lookups change the priority. Additionally, server-side ACLs and the data lookup engine fire when the record...
is saved. Finally, a client-side UI policy sets the **Priority** field back to read-only, which is the default setting.

### Watching the incident priority

<table>
<thead>
<tr>
<th>Original values</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Priority:1 - Critical</td>
</tr>
<tr>
<td>• Impact:1 - High</td>
</tr>
<tr>
<td>• Urgency:1 - High</td>
</tr>
</tbody>
</table>

#### First Change

1. The user changes the **Impact** value to **3 - Low**.
2. The priority automatically changes to **3 - Moderate** based on the **Priority Lookup** data lookup definition used by default in ServiceNow incidents.

**Note:** At this point, the record has not been saved.

#### Second Change

1. The user changes the **Urgency** value to **2 - Medium**.
2. The priority automatically changes to **4 - Low** based on the same **Priority Lookup** data lookup definition.
3. The user saves the record by right-clicking the form header and choosing **Save**.

#### Field watcher example

<table>
<thead>
<tr>
<th>Table: Incident Type</th>
<th>Element: Priority</th>
<th>Reference: Priority Qualifiers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>integer</td>
<td></td>
</tr>
</tbody>
</table>

First Change:

- **1:** 14:26:26 (602)  
  DATA_LOOKUP - change of incident impact
  1 → 3

- **2:** 14:26:26 (458)  
  DATA_LOOKUP - change of incident urgency
  3 → 4

Second Change:

- **1:** 13:38:38 (290)  
  UI ACTION: **Save**

- **2:** 13:38:38 (600)  
  ACL - recordincident.priority/write
  true

- **3:** 13:38:38 (600)  
  ACL - recordincident.priority/create
  true

**Note:** The values that change from 1 to 3, and then from 3 to 4, refer to the numerical values in the choice list.
**Writing to the debug log**

To write to the debug log in your client-side JavaScript, or UI policies, make a call to the global function `jslog()`.

An example of using `jslog()` in JavaScript:

```javascript
function logData (r ) {
    lastLogDate  = r. responseXML. documentElement. getAttribute ( "last_log_entry" ) ;
    var items  = r. responseXML. getElementsByTagName ( "log" ) ;
    jslog ( "response=" + r. responseText ) ;
}
```

Additionally, when client scripts run, the name of the client script and timing information is displayed. This can be useful in determining which scripts are running and whether they are impacting performance.

**Debug UI policies**

Enabling the `glide.ui.ui_policy_debug` property lets you monitor the processing of UI actions.

Here are some sample log events from an incident policy that sets fields to read-only if the incident_state is closed.

```
GlideFieldPolicy: Evaluating condition
GlideFieldPolicy:     incident_state (7) = 7 -> true
GlideFieldPolicy: -->>> TRUE
GlideFieldPolicy: Setting opened_at disabled to true
GlideFieldPolicy: Setting opened_by disabled to true
GlideFieldPolicy: Setting closed_at disabled to true
GlideFieldPolicy: Setting closed_by disabled to true
GlideFieldPolicy: Setting company disabled to true
```

**Access the JavaScript log**

JavaScript that runs on the browser, such as client scripts, can include a call to `jslog()` to send information to the JavaScript Log. Users with the admin role can access this log.
Procedure

1. Open the JavaScript log by navigating to the appropriate location for your version of the UI.

| UI16                | a. Click the gear icon in the banner frame.  
|                    | b. Click the **Developer** section.  
|                    | c. Toggle the **JavaScript Log and Field Watcher** switch.  
| UI15                | a. Click the gear icon in the banner frame.  
|                    | b. Click **JavaScript Log and Field Watcher**.  
| UI11                | Click the debug icon ( Aç ) in the banner frame.  

A new pane opens at the bottom of the screen. It shows the JavaScript Log tab and may also show the Field Watcher tab.

2. If needed, select the **JavaScript Log** tab.

3. Click the clear icon ( X ) to clear the contents of the log, as needed.

**JavaScript debug window**

The JavaScript debug window appears in a bottom pane of the user interface when an administrator turns on debugging.

Use the debug window to access these tools.

- **JavaScript Log**: JavaScript that runs on the browser, such as client scripts, can include a call to `jslog()` to send information to the JavaScript log.
- **Field Watcher**: a tool that tracks and displays all actions that the system performs on a selected form field.

![JavaScript debug window](image)
Using the JavaScript debug window

The JavaScript debug window enables access to the JavaScript Log and the Field Watcher tools.

About this task

The steps to access the JavaScript debug window depend on which UI version you are using.

Procedure

1. Open the JavaScript debug window by navigating to the appropriate location for your version of the UI.

<table>
<thead>
<tr>
<th>UI 16</th>
<th>a. Click the gear icon in the banner frame.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>b. Click the Developer section.</td>
</tr>
<tr>
<td></td>
<td>c. Toggle the JavaScript Log and Field Watcher switch.</td>
</tr>
<tr>
<td>UI 15</td>
<td>a. Click the gear icon in the banner frame.</td>
</tr>
<tr>
<td></td>
<td>b. Click JavaScript Log and Field Watcher.</td>
</tr>
<tr>
<td>UI 11</td>
<td>Click the debug icon (扳手) in the banner frame.</td>
</tr>
</tbody>
</table>

The JavaScript debug window opens at the bottom of the screen. The tab that is currently active in the window is the last tab that was active when the window was closed.

2. Click a tab to use one of the debug window features.

   • JavaScript Log
   • Field Watcher

Related information

   Writing to the debug log
   Field watcher

JS Code Coverage Debug

The JS Code Coverage Debug application allows administrators and application developers to log the scripts triggered during a user session and then review which lines of code the system ran.

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Users with the js_coverage_debugger role can debug scripts without having to set breakpoints or review onscreen debug messages. Instead, the system saves script usage data in the JavaScript Code Coverage [sys_js_code_coverage] table. Each JavaScript Code Coverage record contains:

- The user session that called the script
- The script record the system called identified by table, sys_id, and script field
- The script record the system called identified by type and name
- The transaction that called the script
- The start time of the transaction
- The contents of the script field highlighted to indicate which lines the system ran

### Sample code coverage highlighting

```
var build = gs.getProperty("glide.buildname");
if (GlideStringUtil.notNil(build))
  build = build.substring(0,1).toLowerCase();

var collision = new GlideRecord("sys_embedded_help_content");
collision.addActiveQuery();
collision.addQuery("page", current.page);
collision.addQuery("modifier", current.modifier);
collision.addQuery("product", current.product);
if (current.isValidRecord() & GlideStringUtil.notNil(current.sys_id))
  collision.addQuery("sys_id", "!=", current.sys_id);
else
  collision.addNullQuery("qualifier");
if (current.version == "all" & GlideStringUtil.notNil(build))
  collision.addQuery("version", build).addOrCondition("version", "all");
else
  collision.addQuery("version", current.version);
collision.query();
if (collision.next()) {
  if (collision.canWrite())
    gs.addErrorMessage(gs.getMessage("Embedded Help Composite Key Validation"));
```

### JS Code Coverage highlighting

The JS Code Coverage application highlights script fields to indicate whether the system ran or skipped each line.
Sample code highlighting

```javascript
var build = gs.getProperty("glide.buildname");
if (GlideStringUtil.notNull(build))
    build = build.substring(0,1).toLowerCase();

var collision = new GlideRecord("sys_embedded_help_content");
collision.addActiveQuery();
collision.addQuery("page", current.page);
collision.addQuery("modifier", current.modifier);
collision.addQuery("product", current.product);
if (current.isValidRecord() & GlideStringUtil.notNull(current.sys_id))
    collision.addQuery("sys_id", "!=" current.sys_id);

if (GlideStringUtil.notNull(current.qualifier))
    collision.addQuery("qualifier", current.qualifier);
else
    collision.addNullQuery("qualifier");

if (current.version == "all" & GlideStringUtil.notNull(build))
collision.addQuery("version", build).addOrCondition("version", "all");
else
    collision.addQuery("version", current.version);
collision.query();
if (collision.next()) {
    if (collision.canWrite())
        gs.addErrorMessage(gs.getMessage("Embedded Help Composite Key Validation"));
```

The color of the highlight indicates how the system evaluated the code line.

### Meaning of code highlighting

<table>
<thead>
<tr>
<th>Highlight color</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Green</td>
<td>This is an executable line of code that the system ran during the session.</td>
</tr>
<tr>
<td>Red</td>
<td>This is an executable line of code that the system skipped for some reason. The system may have skipped an executable line of code because the necessary script conditions were not met or because the script function was never called. You may want to use the Script Debugger to determine why the system skipped the line of executable code.</td>
</tr>
<tr>
<td>Gray</td>
<td>This is a non-executable line of code such as white space, code comment, or a portion of an expression split across multiple lines that cannot run on its own.</td>
</tr>
</tbody>
</table>

Administrators and application developers can use this information to conduct more targeted debugging activities such as using the Script Debugger to determine why script conditions are not being met.
Activate JS Code Coverage Debug

You can activate the JS Code Coverage Debug plugin (com.glide.js.coverage) if you have the admin role.

Before you begin
Role required: admin

Procedure
1. Navigate to System Applications > All Available Applications > All.
2. Find the plugin using the filter criteria and search bar.
   You can search for the plugin by its name or ID. If you cannot find a plugin, you might have to request it from ServiceNow personnel. For more information, see Request a plugin.
3. Click Install, and then in the Activate Plugin dialog box, click Activate.

   Note: When domain separation and delegated admin are enabled in an instance, the administrative user must be in the global domain. Otherwise, the following error appears: Application installation is unavailable because another operation is running: Plugin Activation for <plugin name>.

What to do next
To see the components the plugin installed, refresh the plugin form and select the Plugin Files related list.

Debug with JS Code Coverage Debug

Use JS Code Coverage Debug to record a user session and then review which scripts and lines of code the system ran.

Before you begin
Role required: admin or js_coverage_debugger

Procedure
1. Navigate to JS Code Coverage Debug > Enable Coverage.
   The system logs which scripts and code lines the system runs as well as displays session debug messages in the JS Code Coverage namespace.
2. Navigate to the table or page whose logic you want to test.

**Example**
For example, navigate to Incident > Create New.

3. Trigger the script or scripts you want to test.

**Example**
For example, create an incident with an associate CI item to test several business rules.

4. When you have completed testing, navigate to JS Code Coverage Debug > Disable Coverage.
The system stops logging script and code lines run.

The system displays the list of coverage data associated with the current user session.
6. Select the script or transaction you want to review.

### JavaScript Code Coverage fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Script Name</td>
<td>Displays the script run by table name, sys_id value, and script field.</td>
</tr>
<tr>
<td>Script Reference</td>
<td>Displays the script run by script type and name.</td>
</tr>
<tr>
<td>Transaction</td>
<td>Displays the transaction that called the script by thread ID and URI.</td>
</tr>
</tbody>
</table>

**Example**

For example, select the **Script Reference** Business Rule: incident events. The system displays the JS Code Coverage Debug record.

```javascript
if (current.operation() == 'insert' && current.comments.changes) {
  gs.eventQueue("incidentcommented", current, gs.getUserID(), gs.getUserName());
}
```

```javascript
if (current.operation() == 'insert') {
  gs.eventQueue("incident.inserted", current, gs.getUserID(), gs.getUserName());
}
```

```javascript
if (current.operation() == 'update') {
  gs.eventQueue("incident.updated", current, gs.getUserID(), gs.getUserName());
}
```

```javascript
if (current.assigned_to.nil() && current.assigned_to.changes()) {
  gs.eventQueue("incident.assigned", current,
  current.assigned_to.getDisplayValue(), previous.assigned_to.getDisplayValue());
}
```

```javascript
if (current.assignment_group.nil() && current.assignment_group.changes()) {
  gs.eventQueue("incident.assignment_group", current,
  previous.assignment_group.getDisplayValue());
}
```

```javascript
if (current.priority.changes() && current.priority == 1) {
  gs.eventQueue("incident.priority.1", current, current.priority,
  previous.priority);
}
```
7. Review the **Script** field to determine which lines of code the system ran.

**Example**
For example, the business rule added the incident.inserted event to the event queue.

**Results**
You determine which lines of code the system ran.

**What to do next**
Use the code coverage information to do more targeted debugging activities such as set breakpoints and review variable values with the Script Debugger.

### Packages Call Removal tool

The Packages Call Removal Tool provides modules to identify fields that might contain scripts, find scripts that contain Packages calls to ServiceNow Java classes, and to examine proposed script changes that eliminate those Packages calls. You can substitute your own changes in place of the proposed changes.

Packages calls to ServiceNow Java classes will be prevented in a future release. The Packages Call Removal tool helps prepare your instance to use the new API and includes the following scripts and pages:

- The Find Packages Fields script scans scripts for Packages calls to ServiceNow Java classes.
- The Find Packages Calls script proposes changes that remove Packages calls or replace them with GlideScriptable names.
- The Packages Call Items page lists, and enables you to work on, proposed changes to scripts.

The tool is available to users with the administrator role.

The tool can generate errors as it tries to generate preferred, scriptable alternatives for Packages calls to ServiceNow Java classes.

**Note:** If you intend to capture and migrate the changes that result from running the Packages Call Removal Tool, you should first create an update set.

### Activate the Packages Call Removal Tool

You must activate the Packages Call Removal Tool plugin to access the tool.
Before you begin
Role required: admin

Procedure
1. Navigate to System Applications > All Available Applications > All.
2. Find the Packages Call Removal Tool plugin using the filter criteria and search bar.
   You can search for the plugin by its name or ID. If you cannot find a plugin, you might have to request it from ServiceNow personnel. For more information, see Request a plugin.
3. Click Install, and then in the Activate Plugin dialog box, click Activate.

   Note: When domain separation and delegated admin are enabled in an instance, the administrative user must be in the global domain. Otherwise, the following error appears: Application installation is unavailable because another operation is running: Plugin Activation for <plugin name>.

Find a Packages call
After you run the Find Packages Fields script to define the list of fields to search for Packages calls, run the Find Packages Calls script to generate the list of fields with Packages calls to ServiceNow Java classes. The script also proposes changes.
Procedure

1. Click **(3) Find Packages Calls (script)** to execute a script that searches the fields for packages calls and populates the Packages Call Items list with proposed changes.

Example

2. Click **(4) Packages Call Items** to display the list of affected fields on the Packages Call Items page.

The Packages Call Items page lists the items with Packages calls, shows each item's current state, the table that contains the field, the affected record, the number of Packages calls contained in the field's script, and the number of errors that occurred when the proposed script was generated.
3. On the Packages Call Items page, click a record for any of the listed items to open the form and revise as described in Replacing Packages Calls.

As you work through the list, the **State** of the items are updated and the items are grouped by State:

- **Proposed**: A proposed revision exists for the field.
- **Error(red)**: One or more errors occurred when the proposed script was generated.
- **Rejected(gray)**: A proposed change has been rejected.
- **Completed(green)**: The field has been successfully revised to remove Packages calls to ServiceNow Java classes.
- **Canceled (gray)**: Since the proposed change was generated, either the original script has been changed to no longer require modification, or the original record no longer exists.

**Remove Packages calls from scripts**

The Packages Call Items page lists, and enables you to work on, proposed changes to scripts.
About this task
The Packages Call Items page lists all Packages calls to ServiceNow Java classes. From the page, you can open any item in a form to examine the proposed changes. You can accept a proposed change as-is, or edit the code as necessary to remove the Packages calls.

Initially, all items without errors are in the Proposed state. When you take an action on a field, its state is updated. As you work through the list, each field is tracked by its current state: Proposed, Error, Rejected, Completed, or Canceled. You work through all of the fields until all of the Packages calls to ServiceNow Java classes have been removed.

Procedure
1. To explore the details of an item in the list, click the link in the **State** column. The Package Migration Item form shows each line of the Original (current) version of the script and of the Proposed version. Lines with proposed changes are highlighted in yellow.

![Package Migration Item form](image)

2. Work through the items until all are in the **Completed**, **Canceled**, or **Rejected** state.

<table>
<thead>
<tr>
<th>To accomplish this for an item</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Accept the proposed change</strong></td>
<td><strong>Click Execute Change.</strong></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>To accomplish this for an item</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reject the proposed change</td>
<td>Click <strong>Reject Change</strong>. The change goes into the Rejected state. Rejecting a field allows you to defer action. To perform the change later, click <strong>Undo Rejection</strong>.</td>
</tr>
</tbody>
</table>
| Make the proposed change       | Click **Edit Proposed Version** to an editable script field where you can make changes. Below the edit field, a Diff pane displays line-by-line differences between the original and proposed versions. After you save changes, code is highlighted in the Diff pane as follows:  
  - **Yellow** indicates code differences between the original version and the proposed version.  
  - **Green** indicates lines added to the proposed version that were not in the original version.  
  - **Red** indicates lines in the original version that have been removed from the proposed version.  
  a. Make changes in the **Proposed version** pane.  
  b. To save the change and rerun the comparison, click **Save**. If the change generates an error, ServiceNow puts the field into the **Error** state.  
  c. To revert a change, click the text in the **State** column to reopen the field on the Packages Call Items page, and then click **Revert Change**. |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Glide object name

The names of the new Glide objects that replace Packages calls are derived from the Java package name used in the Packages call.

About this task

Although the tool automatically substitutes the appropriate new scriptable name for each Packages call it encounters, in some circumstances it can be useful to know how to manually replace a Packages call with its new scriptable object equivalent. Use the steps below to determine the new script object name from the Packages call name. The replacement objects include the same methods and properties as the objects they replace.

To determine the new object name:

Procedure

1. Note the third term in the Java package name. This is usually glide, but is sometimes snc or glideapp.

2. Drop all of the prefix terms, leaving only the last. For example, Packages.com.glide.monitor.AbstractBucketCollector becomes AbstractBucketCollector.

3. Capitalize the first letter of the term noted in the first step above and add it to the front of the term defined in step 2. For example, Packages.com.glide.monitor.AbstractBucketCollector becomes GlideAbstractBucketCollector and Packages.com.snc.cmdb.BaselineCMDB becomes SncBaselineCMDB.

4. Verify that the name is valid by executing a gs.print() command in Scripts Background, specifying only the name with no quotes. For example:

   gs.print( SncBaselineCMDB );

*** Script: [JavaClass com.snc.cmdb.BaselineCMDB]

Note that there are exceptions to this rule, such as "Glide", which replaces "Packages.com.glide.Glide", "TestExtension", which replaces "Packages.com.glide.junit.misc.TestExtension"; and "UINotification", which replaces "Packages.com.glide.ui.UINotification".

Glide object replacement list

This table lists the Glide classes and the Packages calls they replace.
**Note:** The publication of this list does not imply that these scriptable objects are for use by customers, consultants, and partners. The use of the Glide prefix does not imply that these scriptable objects are in the same category as or have the same status as objects such as GlideRecord. Except where documentation is provided in the API Reference, these undocumented APIs are not intended for general use, and ServiceNow, Inc., does not make any commitment to document them, answer questions about them, or maintain them indefinitely in their current form. Over time, ServiceNow, Inc., intends to migrate a subset of the functionality represented by these objects into the documented API and remove the rest.

### GlideScriptable object replacement list

<table>
<thead>
<tr>
<th>GlideScriptable Class</th>
<th>Packages Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>Glide</td>
<td>Packages.com.glide.Glide</td>
</tr>
<tr>
<td>GlideAbstractDomainProvider</td>
<td>Packages.com.glide.db.domain.AbstractDomainProvider</td>
</tr>
<tr>
<td>GlideAbstractListener</td>
<td>Packages.com.glide.listener.AbstractListener</td>
</tr>
<tr>
<td>GlideActionManager</td>
<td>Packages.com.glide.ui.action.ActionManager</td>
</tr>
<tr>
<td>GlideAJAXScheduleItem</td>
<td>Packages.com.glide.schedules.AJAXScheduleItem</td>
</tr>
<tr>
<td>GlideAJAXSchedulePage</td>
<td>Packages.com.glide.schedules.AJAXSchedulePage</td>
</tr>
<tr>
<td>GlideAlertActions</td>
<td>Packages.com.glide.alerts.AlertActions</td>
</tr>
<tr>
<td>GlideappADSIloader</td>
<td>Packages.com.glideapp.ecc.ADSILoader</td>
</tr>
<tr>
<td>GlideappAJAXMapPage</td>
<td>Packages.com.glideapp.google_maps.AJAXMapPage</td>
</tr>
<tr>
<td>GlideappCalculationHelper</td>
<td>Packages.com.glideapp.servicecatalog.CalculationHelper</td>
</tr>
<tr>
<td>GlideappCart</td>
<td>Packages.com.glideapp.servicecatalog.Cart</td>
</tr>
<tr>
<td>GlideappCartItem</td>
<td>Packages.com.glideapp.servicecatalog.CartItem</td>
</tr>
<tr>
<td>GlideappCatalogCategoryBatcher</td>
<td>Packages.com.glideapp.servicecatalog.CategoryBatcher</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>GlideappCatalogItem</td>
<td>Packages.com.glideapp.servicecatalog.CatalogItem</td>
</tr>
<tr>
<td>GlideappCategory</td>
<td>Packages.com.glideapp.servicecatalog.Category</td>
</tr>
<tr>
<td>GlideappCategoryPopper</td>
<td>Packages.com.glideapp.servicecatalog.CategoryPopper</td>
</tr>
<tr>
<td>GlideappCatItemPopper</td>
<td>Packages.com.glideapp.servicecatalog.CatItemPopper</td>
</tr>
<tr>
<td>GlideappChartParameters</td>
<td>Packages.com.glideapp.chart.ChartParameters</td>
</tr>
<tr>
<td>GlideappChatRoom</td>
<td>Packages.com.glideapp.live.db.ChatRoom</td>
</tr>
<tr>
<td>GlideappCheckBoxQuestion</td>
<td>Packages.com.glideapp.questionset.CheckBoxQuestion</td>
</tr>
<tr>
<td>GlideappCMDBHelper</td>
<td>Packages.com.glideapp.ecc.CMDBHelper</td>
</tr>
<tr>
<td>GlideappCMDBSoftwareHelper</td>
<td>Packages.com.glideapp.ecc.CMDBSoftwareHelper</td>
</tr>
<tr>
<td>GlideappContainerAwareQuestionSet</td>
<td>Packages.com.glideapp.questionset.ContainerAwareQuestionSet</td>
</tr>
<tr>
<td>GlideappDateQuestion</td>
<td>Packages.com.glideapp.questionset.DateQuestion</td>
</tr>
<tr>
<td>GlideappDateTimeQuestion</td>
<td>Packages.com.glideapp.questionset.DateTimeQuestion</td>
</tr>
<tr>
<td>GlideappDeliveryPlan</td>
<td>Packages.com.glideapp.servicecatalog.DeliveryPlan</td>
</tr>
<tr>
<td>GlideappECCInputMessage</td>
<td>Packages.com.glideapp.ecc.ECCInputMessage</td>
</tr>
<tr>
<td>GlideappECCOutputMessage</td>
<td>Packages.com.glideapp.ecc.ECCOutputMessage</td>
</tr>
<tr>
<td>GlideappECCQueueConnector</td>
<td>Packages.com.glideapp.ecc.ECCQueueConnector</td>
</tr>
<tr>
<td>GlideappECCQueueProcessor</td>
<td>Packages.com.glideapp.ecc.ECCQueueProcessor</td>
</tr>
<tr>
<td>GlideappECCResponseMessage</td>
<td>Packages.com.glideapp.ecc.ECCResponseMessage</td>
</tr>
<tr>
<td>GlideappExpandableText</td>
<td>Packages.com.glideapp.live_feed.HTMLTransformers.ExpandableText</td>
</tr>
<tr>
<td>GlideappFixes</td>
<td>Packages.com.glideapp.servicecatalog.Fixes</td>
</tr>
<tr>
<td>GlideappHome</td>
<td>Packages.com.glideapp.home.Home</td>
</tr>
<tr>
<td>GlideappHomePage</td>
<td>Packages.com.glideapp.home.HomePage</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>---------------------------------------------------------</td>
</tr>
<tr>
<td>GlideappIECC</td>
<td>Packages.com.glideapp.ecc.IECC</td>
</tr>
<tr>
<td>GlideappIOUpgrade</td>
<td>Packages.com.glideapp.servicecatalog.IOUpgrade</td>
</tr>
<tr>
<td>GlideappItemOptionsQuestionSet</td>
<td>Packages.com.glideapp.servicecatalog.ItemOptionsQuestionSet</td>
</tr>
<tr>
<td>GlideappJMSECCReceiver</td>
<td>Packages.com.glideapp.jms.JMSECCReceiver</td>
</tr>
<tr>
<td>GlideappJMSECCSender</td>
<td>Packages.com.glideapp.jms.JMSECCSender</td>
</tr>
<tr>
<td>GlideappKBIncludes</td>
<td>Packages.com.glideapp.knowledge.KBIncludes</td>
</tr>
<tr>
<td>GlideApplication</td>
<td>Packages.com.glide.sys.Application</td>
</tr>
<tr>
<td>GlideApplicationModule</td>
<td>Packages.com.glide.processors.Application</td>
</tr>
<tr>
<td>GlideappListCollectorQuestion</td>
<td>Packages.com.glideapp.questionset.ListCollectorQuestion</td>
</tr>
<tr>
<td>GlideappLiveFeedEventHandler</td>
<td>Packages.com.glideapp.live_feed.LiveFeedEventHandler</td>
</tr>
<tr>
<td>GlideappLiveFeedUIAction</td>
<td>Packages.com.glideapp.live_feed.LiveFeedUIAction</td>
</tr>
<tr>
<td>GlideappLiveProfile</td>
<td>Packages.com.glideapp.live_common.LiveProfile</td>
</tr>
<tr>
<td>GlideappLiveUtils</td>
<td>Packages.com.glideapp.live.LiveUtils</td>
</tr>
<tr>
<td>GlideappLookupSelectQuestion</td>
<td>Packages.com.glideapp.questionset.LookupSelectQuestion</td>
</tr>
<tr>
<td>GlideappQuestion</td>
<td>Packages.com.glideapp.questionset.Question</td>
</tr>
<tr>
<td>GlideappQuestionChoice</td>
<td>Packages.com.glideapp.questionset.QuestionChoice</td>
</tr>
<tr>
<td>GlideappQueueHelper</td>
<td>Packages.com.glideapp.ecc.QueueHelper</td>
</tr>
<tr>
<td>GlideappQueueReader</td>
<td>Packages.com.glideapp.ecc.QueueReader</td>
</tr>
<tr>
<td>GlideappReferenceQuestion</td>
<td>Packages.com.glideapp.questionset.Reference</td>
</tr>
<tr>
<td>GlideappRequestItemWorkflow</td>
<td>Packages.com.glideapp.servicecatalog.RequestItemWorkflow</td>
</tr>
<tr>
<td>GlideappRequestNew</td>
<td>Packages.com.glideapp.servicecatalog.RequestNew</td>
</tr>
<tr>
<td>GlideappScriptHelper</td>
<td>Packages.com.glideapp.servicecatalog.ScriptHelper</td>
</tr>
</tbody>
</table>
## GlideScriptable object replacement list (continued)

<table>
<thead>
<tr>
<th>GlideScriptable Class</th>
<th>Packages Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideappSequencedQuestionSet</td>
<td>Packages.com.glideapp.questionset.SequencedQuestionSet</td>
</tr>
<tr>
<td>GlideappTaskApprovalHelper</td>
<td>Packages.com.glideapp.servicecatalog.TaskApprovalHelper</td>
</tr>
<tr>
<td>GlideappTime Ago</td>
<td>Packages.com.glideapp.live_feed.TimeAgo</td>
</tr>
<tr>
<td>GlideappUpgradeQuestions</td>
<td>Packages.com.glideapp.survey.UpgradeQuestions</td>
</tr>
<tr>
<td>GlideappValveProcessor</td>
<td>Packages.com.glideapp.servicecatalog.valve.ValveProcessor</td>
</tr>
<tr>
<td>GlideappVariable</td>
<td>Packages.com.glideapp.servicecatalog.variables.Variable</td>
</tr>
<tr>
<td>GlideappVariablePoolQuestionSet</td>
<td>Packages.com.glideapp.servicecatalog.variables.VariablePoolQuestionSet</td>
</tr>
<tr>
<td>GlideappWizardIntercept</td>
<td>Packages.com.glideapp.wizard.WizardIntercept</td>
</tr>
<tr>
<td>GlideappWMI Loader</td>
<td>Packages.com.glideapp.ecc.WMI Loader</td>
</tr>
<tr>
<td>GlideappWorkflow</td>
<td>Packages.com.glideapp.workflow.Workflow</td>
</tr>
<tr>
<td>GlideappWorkflowHelper</td>
<td>Packages.com.glideapp.workflow.WorkflowHelper</td>
</tr>
<tr>
<td>GlideappYesNoQuestion</td>
<td>Packages.com.glideapp.questionset.YesNo</td>
</tr>
<tr>
<td>GlideAQueryExplanation</td>
<td>Packages.com.glide.db.explain.AQueryExplanation</td>
</tr>
<tr>
<td>GlideArchiver</td>
<td>Packages.com.glide.db.auxiliary.Archiver</td>
</tr>
<tr>
<td>GlideArchiveRecord</td>
<td>Packages.com.glide.db.auxiliary.ArchiveRecord</td>
</tr>
<tr>
<td>GlideArchiveRestore</td>
<td>Packages.com.glide.db.auxiliary.ArchiveRestore</td>
</tr>
<tr>
<td>GlideArchiveStatus</td>
<td>Packages.com.glide.db.auxiliary.ArchiveStatus</td>
</tr>
<tr>
<td>GlideArchiveTable</td>
<td>Packages.com.glide.db.auxiliary.ArchiveTable</td>
</tr>
<tr>
<td>GlideARecurrence</td>
<td>Packages.com.glide.schedule.recurrence..</td>
</tr>
<tr>
<td>GlideAttachmentIndexDocument</td>
<td>Packages.com.glide.lucene.attachments.AttachmentIndexDocument</td>
</tr>
<tr>
<td>GlideAttachmentIndexTypes</td>
<td>Packages.com.glide.lucene.attachments.AttachmentIndexTypes</td>
</tr>
<tr>
<td>GlideAttributes</td>
<td>Packages.com.glide.util.GlideAttributes</td>
</tr>
<tr>
<td>GlideAuditDelete</td>
<td>Packages.com.glide.audit.AuditDelete</td>
</tr>
<tr>
<td>GlideAuditor</td>
<td>Packages.com.glide.script.Auditor</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>GlideScriptable Class</th>
<th>Packages Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideAutomationEncrypter</td>
<td>Packages.com.glide.util.AutomationEncrypter</td>
</tr>
<tr>
<td>GlideBaseTag</td>
<td>Packages.com.glide.ui.jelly.tags.BaseTag</td>
</tr>
<tr>
<td>GlideBootstrap</td>
<td>Packages.com.glide.db.impex.Bootstrap</td>
</tr>
<tr>
<td>GlideBoundedIntProperty</td>
<td>Packages.com.glide.util.BoundedIntProperty</td>
</tr>
<tr>
<td>GlideCacheManager</td>
<td>Packages.com.glide.sys.cache.CacheManager</td>
</tr>
<tr>
<td>GlideCalendar</td>
<td>Packages.com.glide.schedule.GlideCalendar</td>
</tr>
<tr>
<td>GlideCalendarWeekEntry</td>
<td>Packages.com.glide.calendar.GlideCalendar</td>
</tr>
<tr>
<td>GlideCanceledUITransaction</td>
<td>Packages.com.glide.ui.CanceledUITransaction</td>
</tr>
<tr>
<td>GlideCascadeFromDelete</td>
<td>Packages.com.glide.db.CascadeFromDelete</td>
</tr>
<tr>
<td>GlideCatalogCloneWorker</td>
<td>Packages.com.glide.catalog.cloner.CatalogCloneWorker</td>
</tr>
<tr>
<td>GlideChannel</td>
<td>Packages.com.glide.channel.Channel</td>
</tr>
<tr>
<td>GlideChannelManager</td>
<td>Packages.com.glide.channel.ChannelManager</td>
</tr>
<tr>
<td>GlideChannelMessage</td>
<td>Packages.com.glide.channel.ChannelMessage</td>
</tr>
<tr>
<td>GlideChartFieldColors</td>
<td>Packages.com.glide.ui.chart.dataset.ChartFieldColors</td>
</tr>
<tr>
<td>GlideChartGeneratorFactory</td>
<td>Packages.com.glide.ui.chart.ChartGeneratorFactory</td>
</tr>
<tr>
<td>GlideChartUtil</td>
<td>Packages.com.glide.ui.chart.dataset.ChartUtil</td>
</tr>
<tr>
<td>GlideChartValue</td>
<td>Packages.com.glide.ui.chart.dataset.ChartValue</td>
</tr>
<tr>
<td>GlideChecksum</td>
<td>Packages.com.glide.util.Checksum</td>
</tr>
<tr>
<td>GlideChoice</td>
<td>Packages.com.glide.choice.Choice</td>
</tr>
<tr>
<td>GlideChoiceListGenerator</td>
<td>Packages.com.glide.choice.ChoiceListGenerator</td>
</tr>
<tr>
<td>GlideClientBrowserTimes</td>
<td>Packages.com.glide.client_transaction.ClientBrowserTimes</td>
</tr>
<tr>
<td>GlideClientNetworkTimes</td>
<td>Packages.com.glide.client_transaction.ClientNetworkTimes</td>
</tr>
</tbody>
</table>
## GlideScriptable object replacement list (continued)

<table>
<thead>
<tr>
<th>GlideScriptable Class</th>
<th>Packages Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideClusterState</td>
<td>Packages.com.glide.cluster.ClusterState</td>
</tr>
<tr>
<td>GlideClusterSynchronizer</td>
<td>Packages.com.glide.cluster.ClusterSynchronizer</td>
</tr>
<tr>
<td>GlideCMSLinkHelper</td>
<td>Packages.com.glide.cms.CMSLinkHelper</td>
</tr>
<tr>
<td>GlideCMSPageLink</td>
<td>Packages.com.glide.cms.CMSPageLink</td>
</tr>
<tr>
<td>GlideCollectionEnumerator</td>
<td>Packages.com.glide.util.CollectionEnumerator</td>
</tr>
<tr>
<td>GlideCollectionQueryCalculator</td>
<td>Packages.com.glide.ui.CollectionQueryCalculator</td>
</tr>
<tr>
<td>GlideCollisionDetector</td>
<td>Packages.com.glide.update.collisions.CollisionDetector</td>
</tr>
<tr>
<td>GlideColumnAttributes</td>
<td>Packages.com.glide.db.impex.ColumnAttributes</td>
</tr>
<tr>
<td>GlideCompanyResolver</td>
<td>Packages.com.glide.misc.CompanyResolver</td>
</tr>
<tr>
<td>GlideCompiler</td>
<td>Packages.com.glide.script.Compiler</td>
</tr>
<tr>
<td>GlideCompositeElement</td>
<td>Packages.com.glide.db.CompositeElement</td>
</tr>
<tr>
<td>GlideConfiguration</td>
<td>Packages.com.glide.notification.Configuration</td>
</tr>
<tr>
<td>GlideContentConfig</td>
<td>Packages.com.glide.cms.ContentConfig</td>
</tr>
<tr>
<td>GlideContentPage</td>
<td>Packages.com.glide.cms.ContentPage</td>
</tr>
<tr>
<td>GlideContentSite</td>
<td>Packages.com.glide.cms.ContentSite</td>
</tr>
<tr>
<td>GlideContentType</td>
<td>Packages.com.glide.cms.ContentType</td>
</tr>
<tr>
<td>GlideContextMenu</td>
<td>Packages.com.glide.db_context_menu.ContextMenu</td>
</tr>
<tr>
<td>GlideContextMenuItem</td>
<td>Packages.com.glide.db_context_menu.ContextMenuItem</td>
</tr>
<tr>
<td>GlideController</td>
<td>Packages.com.glide.script.GlideController</td>
</tr>
<tr>
<td>GlideConverter</td>
<td>Packages.com.glide.currency.Converter</td>
</tr>
<tr>
<td>GlideCookieMan</td>
<td>Packages.com.glide.ui.CookieMan</td>
</tr>
<tr>
<td>GlideCounter</td>
<td>Packages.com.glide.util.Counter</td>
</tr>
<tr>
<td>GlideCredentials</td>
<td>Packages.com.glide.communications.crypto.Credentials</td>
</tr>
<tr>
<td>GlideCryptoService</td>
<td>Packages.com.glide.security.CryptoService</td>
</tr>
<tr>
<td>GlideCSVExporter</td>
<td>Packages.com.glide.generators.CSVExporter</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td>GlideDatabaseVerifier</td>
<td>Packages.com.glide.db.DatabaseVerifier</td>
</tr>
<tr>
<td>GlideDatabaseViewLink</td>
<td>Packages.com.glide.database_views.DatabaseViewLink</td>
</tr>
<tr>
<td>GlideDataSource</td>
<td>Packages.com.glide.db.impex.datasource.DataSource</td>
</tr>
<tr>
<td>GlideDate</td>
<td>Packages.com.glide.glideobject.GlideDate</td>
</tr>
<tr>
<td>GlideDateTime</td>
<td>Packages.com.glide.glideobject.GlideDateTime</td>
</tr>
<tr>
<td>GlideDateUtil</td>
<td>Packages.com.glide.util.DateUtil</td>
</tr>
<tr>
<td>GlideDBAction</td>
<td>Packages.com.glide.db.DBAction</td>
</tr>
<tr>
<td>GlideDBAggregateQuery</td>
<td>Packages.com.glide.db.DBAggregateQuery</td>
</tr>
<tr>
<td>GlideDBAggregateUtil</td>
<td>Packages.com.glide.db.DBAggregateUtil</td>
</tr>
<tr>
<td>GlideDBCategoryDebug</td>
<td>Packages.com.glide.secondary_db_pools.DBCategoryDebug</td>
</tr>
<tr>
<td>GlideDBChangeManager</td>
<td>Packages.com.glide.db.change.DBChangeManager</td>
</tr>
<tr>
<td>GlideDBCompositeAction</td>
<td>Packages.com.glide.db.DBCompositeAction</td>
</tr>
<tr>
<td>GlideDBConfiguration</td>
<td>Packages.com.glide.db.DBConfiguration</td>
</tr>
<tr>
<td>GlideDBConfigurationManager</td>
<td>Packages.com.glide.db.DBConfigurationManager</td>
</tr>
<tr>
<td>GlideDBConfigurationManagerEventHandler</td>
<td>Packages.com.glide.db.DBConfigurationManagerEventHandler</td>
</tr>
<tr>
<td>GlideDBConfigurationParms</td>
<td>Packages.com.glide.db.DBConfigurationParms</td>
</tr>
<tr>
<td>GlideDBConfigurationV2Migrator</td>
<td>Packages.com.glide.db.DBConfigurationV2Migrator</td>
</tr>
<tr>
<td>GlideDBConnection</td>
<td>Packages.com.glide.db.pool.DBConnection</td>
</tr>
<tr>
<td>GlideDBConnectionPool</td>
<td>Packages.com.glide.db.pool.DBConnectionPool</td>
</tr>
<tr>
<td>GlideDBConnectionPooler</td>
<td>Packages.com.glide.db.pool.DBConnectionPooler</td>
</tr>
<tr>
<td>GlideDBDelete</td>
<td>Packages.com.glide.db.DBDelete</td>
</tr>
<tr>
<td>GlideDBI</td>
<td>Packages.com.glide.db.DBI</td>
</tr>
<tr>
<td>GlideDBImageProvider</td>
<td>Packages.com.glide.db_image.DBImageProvider</td>
</tr>
<tr>
<td>GlideDBIMySQL</td>
<td>Packages.com.glide.db.rdbms.mysql.DBIMySQL</td>
</tr>
<tr>
<td>GlideDBIndex</td>
<td>Packages.com.glide.db.DBIndex</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>GlideDBKeyStoreFactory</td>
<td>Packages.com.glide.certificates.DBKeyStoreFactory</td>
</tr>
<tr>
<td>GlideDBMacro</td>
<td>Packages.com.glide.ui.jelly.tags.form.DBMacro</td>
</tr>
<tr>
<td>GlideDBMicroStats</td>
<td>Packages.com.glide.db.DBMicroStats</td>
</tr>
<tr>
<td>GlideDBMultiTargetAction</td>
<td>Packages.com.glide.db.DBMultiTargetAction</td>
</tr>
<tr>
<td>GlideDBObjectManager</td>
<td>Packages.com.glide.db.DBObjectManager</td>
</tr>
<tr>
<td>GlideDBObjectToken</td>
<td>Packages.com.glide.db.DBObjectToken</td>
</tr>
<tr>
<td>GlideDBPoolTest</td>
<td>Packages.com.glide.secondary_db_pools</td>
</tr>
<tr>
<td>GlideDBPropertiesConfig</td>
<td>Packages.com.glide.db.DBPropertiesConfig</td>
</tr>
<tr>
<td>GlideDBQuery</td>
<td>Packages.com.glide.db.DBQuery</td>
</tr>
<tr>
<td>GlideDBTypes</td>
<td>Packages.com.glide.db.DBTypes</td>
</tr>
<tr>
<td>GlideDBUpdate</td>
<td>Packages.com.glide.db.DBUpdate</td>
</tr>
<tr>
<td>GlideDBUtil</td>
<td>Packages.com.glide.db.DBUtil</td>
</tr>
<tr>
<td>GlideDBView</td>
<td>Packages.com.glide.db.DBView</td>
</tr>
<tr>
<td>GlideDebugEvaluator</td>
<td>Packages.com.glide.jsdebug.DebugEvaluator</td>
</tr>
<tr>
<td>GlideDefaultUpdateSaver</td>
<td>Packages.com.glide.update.saver.DefaultUpdateSaver</td>
</tr>
<tr>
<td>GlideDiagram</td>
<td>Packages.com.glide.diagrammer.Diagram</td>
</tr>
<tr>
<td>GlideDiagramAction</td>
<td>Packages.com.glide.diagrammer.Diagram</td>
</tr>
<tr>
<td>GlideDiagramEdge</td>
<td>Packages.com.glide.diagrammer.Diagram</td>
</tr>
<tr>
<td>GlideDiagramElement</td>
<td>Packages.com.glide.diagrammer.Diagram</td>
</tr>
<tr>
<td>GlideDiagramNode</td>
<td>Packages.com.glide.diagrammer.Diagram</td>
</tr>
<tr>
<td>GlideDistUpgradeRunner</td>
<td>Packages.com.glide.dist_upgrade.runner.DistUpgrade</td>
</tr>
<tr>
<td>GlideDomain</td>
<td>Packages.com.glide.db.domain.Domain</td>
</tr>
<tr>
<td>GlideDomainDisplay</td>
<td>Packages.com.glide.db.domain.DomainDisplay</td>
</tr>
<tr>
<td>GlideDomainHierarchy</td>
<td>Packages.com.glide.db.domain.DomainHierarchy</td>
</tr>
<tr>
<td>GlideDomainNumberProvider</td>
<td>Packages.com.glide.db.domain.DomainNumberProvider</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>GlideDomainPathDisplay</td>
<td>Packages.com.glide.db.domain.DomainPathDisplay</td>
</tr>
<tr>
<td>GlideDomainPathProvider</td>
<td>Packages.com.glide.db.domain.DomainPathProvider</td>
</tr>
<tr>
<td>GlideDomainSpoolProvider</td>
<td>Packages.com.glide.db.domain.DomainSpoolProvider</td>
</tr>
<tr>
<td>GlideDomainTree</td>
<td>Packages.com.glide.db.domain.DomainTree</td>
</tr>
<tr>
<td>GlideDomainUtil</td>
<td>Packages.com.glide.db.domain.DomainUtil</td>
</tr>
<tr>
<td>GlideDuration</td>
<td>Packages.com.glide.glideobject.GlideDuration</td>
</tr>
<tr>
<td>GlideECBDownloader</td>
<td>Packages.com.glide.currency.ECBDownloader</td>
</tr>
<tr>
<td>GlideECCQueueTransformer</td>
<td>Packages.com.glide.db.impex.ECCQueueTransformer</td>
</tr>
<tr>
<td>GlideElement</td>
<td>Packages.com.glide.script.GlideElement</td>
</tr>
<tr>
<td>GlideElementDescriptor</td>
<td>Packages.com.glide.db.ElementDescriptor</td>
</tr>
<tr>
<td>GlideElementIterator</td>
<td>Packages.com.glide.util.ElementIterator</td>
</tr>
<tr>
<td>GlideElementXMLSerializer</td>
<td>Packages.com.glide.script.GlideElementXMLSerializer</td>
</tr>
<tr>
<td>GlideEmail</td>
<td>Packages.com.glide.notification.Email</td>
</tr>
<tr>
<td>GlideEmailAction</td>
<td>Packages.com.glide.notification.outbound.EmailAction</td>
</tr>
<tr>
<td>GlideEmailFormatter</td>
<td>Packages.com.glide.notification.outbound.EmailFormatter</td>
</tr>
<tr>
<td>GlideEmailInbound</td>
<td>Packages.com.glide.notification.inbound.EmailInbound</td>
</tr>
<tr>
<td>GlideEmailOutbound</td>
<td>Packages.com.glide.notification.outbound.EmailOutbound</td>
</tr>
<tr>
<td>GlideEmailReader</td>
<td>Packages.com.glide.notification.inbound.EmailReader</td>
</tr>
<tr>
<td>GlideEmailSender</td>
<td>Packages.com.glide.notification.outbound.EmailSender</td>
</tr>
<tr>
<td>GlideEmailWatermark</td>
<td>Packages.com.glide.notification.EmailWatermark</td>
</tr>
<tr>
<td>GlideEmitter</td>
<td>Packages.com.glide.ui.jelly.Emitter</td>
</tr>
<tr>
<td>GlideEncrypter</td>
<td>Packages.com.glide.util.Encrypter</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>----------------------------------------------------</td>
</tr>
<tr>
<td>GlideEncryptionWrapperDB</td>
<td>Packages.com.glide.sys.security.EncryptionWrapperDB</td>
</tr>
<tr>
<td>GlideEncryptionWrapperDBAdmin</td>
<td>Packages.com.glide.sys.security.EncryptionWrapperDBAdmin</td>
</tr>
<tr>
<td>GlideEncryptionWrapperNAE</td>
<td>Packages.com.glide.sys.security.EncryptionWrapperNAE</td>
</tr>
<tr>
<td>GlideEncryptionWrapperNAEAdmin</td>
<td>Packages.com.glide.sys.security.EncryptionWrapperNAEAdmin</td>
</tr>
<tr>
<td>GlideEscalationManager</td>
<td>Packages.com.glide.escalation.EscalationManager</td>
</tr>
<tr>
<td>GlideEscalationTimerJobMarkII</td>
<td>Packages.com.glide.job.EscalationTimerJobMarkII</td>
</tr>
<tr>
<td>GlideEvaluator</td>
<td>Packages.com.glide.script.Evaluator</td>
</tr>
<tr>
<td>GlideEvent</td>
<td>Packages.com.glide.policy.Event</td>
</tr>
<tr>
<td>GlideEventManager</td>
<td>Packages.com.glide.policy.EventManager</td>
</tr>
<tr>
<td>GlideExcelExporter</td>
<td>Packages.com.glide.generators.ExcelExporter</td>
</tr>
<tr>
<td>GlideExcelLoader2</td>
<td>Packages.com.glide.db.impex.ExcelLoader2</td>
</tr>
<tr>
<td>GlideExecutionPlan</td>
<td>Packages.com.glide.execution_plan.ExecutionPlan</td>
</tr>
<tr>
<td>GlideExpressionWrapper</td>
<td>Packages.com.glide.ui.jelly.GlideExpressionWrapper</td>
</tr>
<tr>
<td>GlideExtensionPoint</td>
<td>Packages.com.glide.sys.ExtensionPoint</td>
</tr>
<tr>
<td>GlideFieldList</td>
<td>Packages.com.glide.processors.FieldList</td>
</tr>
<tr>
<td>GlideFile</td>
<td>Packages.com.glide.script.proxy.File</td>
</tr>
<tr>
<td>GlideFileUtil</td>
<td>Packages.com.glide.util.FileUtil</td>
</tr>
<tr>
<td>GlideFilter</td>
<td>Packages.com.glide.script.Filter</td>
</tr>
<tr>
<td>GlideFilterList</td>
<td>Packages.com.glide.script.FilterList</td>
</tr>
<tr>
<td>GlideFixCatalogPlans</td>
<td>Packages.com.glide.fixes.FixCatalogPlans</td>
</tr>
<tr>
<td>GlideFixDeliveryPlans</td>
<td>Packages.com.glide.fixes.FixDeliveryPlans</td>
</tr>
<tr>
<td>GlideFixGroups</td>
<td>Packages.com.glide.fixes.FixGroups</td>
</tr>
<tr>
<td>GlideFixItemOptionsAgain</td>
<td>Packages.com.glide.fixes.FixItemOptionsAgain</td>
</tr>
<tr>
<td>GlideFixSpellCheck</td>
<td>Packages.com.glide.fixes.FixSpellCheck</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>GlideFixStuff</td>
<td>Packages.com.glide.fixes.FixStuff</td>
</tr>
<tr>
<td>GlideFixUsers</td>
<td>Packages.com.glide.fixes.FixUsers</td>
</tr>
<tr>
<td>GlideForm</td>
<td>Packages.com.glide.ui.GlideForm</td>
</tr>
<tr>
<td>GlideFormCommon</td>
<td>Packages.com.glide.ui.GlideFormCommon</td>
</tr>
<tr>
<td>GlideFormulator</td>
<td>Packages.com.glide.ui.GlideFormulator</td>
</tr>
<tr>
<td>GlideGauge</td>
<td>Packages.com.glide.report.Gauge</td>
</tr>
<tr>
<td>GlideGovernor</td>
<td>Packages.com.glide.sys.util.Governor</td>
</tr>
<tr>
<td>GlideGregorianCalendar</td>
<td>Packages.com.glide.util.GlideGregorianCal</td>
</tr>
<tr>
<td>GlideGroup</td>
<td>Packages.com.glide.sys.Group</td>
</tr>
<tr>
<td>GlideGroupByListTag</td>
<td>Packages.com.glide.ui.jelly.tags.form.Group</td>
</tr>
<tr>
<td>GlideGuid</td>
<td>Packages.com.glide.util.Guid</td>
</tr>
<tr>
<td>GlideHierarchicalReference</td>
<td>Packages.com.glide.glideobject.Hierarchical</td>
</tr>
<tr>
<td>GlideHistorySet</td>
<td>Packages.com.glide.audit.HistorySet</td>
</tr>
<tr>
<td>GlideHistoryTag2</td>
<td>Packages.com.glide.ui.jelly.tags.mergedata</td>
</tr>
<tr>
<td>GlideHostUtil</td>
<td>Packages.com.glide.util.HostUtil</td>
</tr>
<tr>
<td>GlideHTTPClient</td>
<td><a href="https://www.youtube.com/watch?feature=player_detailpage&amp;v=9qYl_pBZej">https://www.youtube.com/watch?feature=player_detailpage&amp;v=9qYl_pBZej</a></td>
</tr>
<tr>
<td>GlideHTTPRequest</td>
<td>Packages.com.glide.communications.HTTPRequest</td>
</tr>
<tr>
<td>GlideHTTPResponse</td>
<td>Packages.com.glide.communications.HTTPResponse</td>
</tr>
<tr>
<td>GlideI18NStyle</td>
<td>Packages.com.glide.ui.I18NStyle</td>
</tr>
<tr>
<td>GlideICALUtil</td>
<td>Packages.com.glide.policy.ICALUtil</td>
</tr>
<tr>
<td>GlideConstants</td>
<td>Packages.com.glide.util.IConstants</td>
</tr>
<tr>
<td>GlideIGlideRecord</td>
<td>Packages.com.glide.util.IGlideRecord</td>
</tr>
<tr>
<td>GlideImageLoader</td>
<td>Packages.com.glide.script.ImageLoader</td>
</tr>
<tr>
<td>GlideImpersonate</td>
<td>Packages.com.glide.sys.Impersonate</td>
</tr>
<tr>
<td>GlideImportLog</td>
<td>Packages.com.glide.db.impex.ImportLog</td>
</tr>
<tr>
<td>GlideImportMap</td>
<td>Packages.com.glide.db.impex.ImportMap</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>----------------------------------------------------</td>
</tr>
<tr>
<td>GlideImportMapField</td>
<td>Packages.com.glide.db.impex.ImportMapField</td>
</tr>
<tr>
<td>GlideImportSet</td>
<td>Packages.com.glide.system_import_set.ImportSet</td>
</tr>
<tr>
<td>GlideImportSetLoader</td>
<td>Packages.com.glide.system_import_set.ImportSetLoader</td>
</tr>
<tr>
<td>GlideImportSetRun</td>
<td>Packages.com.glide.system_import_set.ImportSetRun</td>
</tr>
<tr>
<td>GlideImportSetTransformer</td>
<td>Packages.com.glide.system_import_set.ImportSetTransformer</td>
</tr>
<tr>
<td>GlideImportSetTransformerWorker</td>
<td>Packages.com.glide.system_import_set.ImportSetTransformerWorker</td>
</tr>
<tr>
<td>GlideIndexDescriptor</td>
<td>Packages.com.glide.db.IndexDescriptor</td>
</tr>
<tr>
<td>GlideIndexUtils</td>
<td>Packages.com.glide.db.IndexUtils</td>
</tr>
<tr>
<td>GlideIntegerTime</td>
<td>Packages.com.glide.glideobject.IntegerTime</td>
</tr>
<tr>
<td>GlideInternalElementTypeChoiceList</td>
<td>Packages.com.glide.script.InternalElementTypeChoiceList</td>
</tr>
<tr>
<td>GlideInternalMonitor</td>
<td>Packages.com.glide.ui.monitor.InternalMonitor</td>
</tr>
<tr>
<td>GlideIOMonitor</td>
<td>Packages.com.glide.ui.monitor.IOMonitor</td>
</tr>
<tr>
<td>GlideIOStats</td>
<td>Packages.com.glide.db.IOStats</td>
</tr>
<tr>
<td>GlideIPAddressUtil</td>
<td>Packages.com.glide.util.IPAddressUtil</td>
</tr>
<tr>
<td>GlideIQueryCondition</td>
<td>Packages.com.glide.util.IQueryCondition</td>
</tr>
<tr>
<td>GlideIRow</td>
<td>Packages.com.glide.db.meta.IRow</td>
</tr>
<tr>
<td>GlideIRQuerySummary</td>
<td>Packages.com.glide.db.ir.IRQuerySummary</td>
</tr>
<tr>
<td>GlideIRQuerySummarySimple</td>
<td>Packages.com.glide.db.ir.IRQuerySummarySimple</td>
</tr>
<tr>
<td>GlideISecurityManager</td>
<td>Packages.com.glide.sys.security.ISecurityManager</td>
</tr>
<tr>
<td>GlideITableIterator</td>
<td>Packages.com.glide.db.access.ITableIterator</td>
</tr>
<tr>
<td>GlideJDBCLoader</td>
<td>Packages.com.glide.db.impex.JDBCLoader</td>
</tr>
<tr>
<td>GlideJDBCProbeTestWorker</td>
<td>Packages.com.glide.db.impex.JDBCProbeTestWorker</td>
</tr>
<tr>
<td>GlideJID</td>
<td>Packages.com.glide.xmppp.JID</td>
</tr>
<tr>
<td>GlideJSTestUtil</td>
<td>Packages.com.glide.autotester.JSTestUtil</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>GlideJSUtil</td>
<td>Packages.com.glide.script.JSUtil</td>
</tr>
<tr>
<td>GlideLabelEventHandler</td>
<td>Packages.com.glide.labels.LabelEventHandler</td>
</tr>
<tr>
<td>GlideLabelGenerator</td>
<td>Packages.com.glide.db.LabelGenerator</td>
</tr>
<tr>
<td>GlideLabelUtil</td>
<td>Packages.com.glide.labels.LabelUtil</td>
</tr>
<tr>
<td>GlideLDAP</td>
<td>Packages.com.glide.sys.ldap.LDAP</td>
</tr>
<tr>
<td>GlideLDAPConfig</td>
<td>Packages.com.glide.sys.ldap.LDAPConfig</td>
</tr>
<tr>
<td>GlideLDAPConfigurations</td>
<td>Packages.com.glide.sys.ldap.LDAPConfig</td>
</tr>
<tr>
<td>GlideLDAPErrorAnalyzer</td>
<td>Packages.com.glide.sys.ldap.LDAPErrorAnalyzer</td>
</tr>
<tr>
<td>GlideLDAPGroups</td>
<td>Packages.com.glide.sys.ldap.LDAPGroups</td>
</tr>
<tr>
<td>GlideLDAPRefresh</td>
<td>Packages.com.glide.sys.ldap.LDAPRefresh</td>
</tr>
<tr>
<td>GlideLDAPResult</td>
<td>Packages.com.glide.sys.ldap.LDAPResult</td>
</tr>
<tr>
<td>GlideLDAPTarget</td>
<td>Packages.com.glide.sys.ldap.LDAPTarget</td>
</tr>
<tr>
<td>GlideLDAPTransformQueue</td>
<td>Packages.com.glide.sys.ldap.LDAPTransformQueue</td>
</tr>
<tr>
<td>GlideLDAPUsers</td>
<td>Packages.com.glide.sys.ldap.LDAPUsers</td>
</tr>
<tr>
<td>GlideLDAPUserUpdate</td>
<td>Packages.com.glide.sys.ldap.LDAPUserUpdate</td>
</tr>
<tr>
<td>GlideList</td>
<td>Packages.com.glide.glideobject.GlideList</td>
</tr>
<tr>
<td>GlideListGroupProperties</td>
<td>Packages.com.glide.list_v2.ListGroupProperties</td>
</tr>
<tr>
<td>GlideListLabel</td>
<td>Packages.com.glide.ui.ListLabel</td>
</tr>
<tr>
<td>GlideListProperties</td>
<td>Packages.com.glide.list_v2.ListProperties</td>
</tr>
<tr>
<td>GlideListSearchQuery</td>
<td>Packages.com.glide.ui.ListSearchQuery</td>
</tr>
<tr>
<td>GlideLoader</td>
<td>Packages.com.glide.db.impex.Loader</td>
</tr>
<tr>
<td>GlideLoadTestDirector</td>
<td>Packages.com.glide.load_test.LoadTestDirector</td>
</tr>
<tr>
<td>GlideLocale</td>
<td>Packages.com.glide.sys.GlideLocale</td>
</tr>
<tr>
<td>GlideLocaleLoader</td>
<td>Packages.com.glide.currency.LocaleLoader</td>
</tr>
<tr>
<td>GlideLock</td>
<td>Packages.com.glide.script.Lock</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>GlideLog</td>
<td>Packages.com.glide.util.Log</td>
</tr>
<tr>
<td>GlideLogCleanup</td>
<td>Packages.com.glide.util.LogCleanup</td>
</tr>
<tr>
<td>GlideLRUCache</td>
<td>Packages.com.glide.sys.cache.LRUCache</td>
</tr>
<tr>
<td>GlideLuceneTextIndexEvent</td>
<td>Packages.com.glide.lucene.TextIndexEvent</td>
</tr>
<tr>
<td>GlideMarkupWriter</td>
<td>Packages.com.glide.util.MarkupWriter</td>
</tr>
<tr>
<td>GlideMemoryActive</td>
<td>Packages.com.glide.ui.monitor.MemoryActive</td>
</tr>
<tr>
<td>GlideMemoryCache</td>
<td>Packages.com.glide.ui.monitor.MemoryCache</td>
</tr>
<tr>
<td>GlideMemoryRecord</td>
<td>Packages.com.glide.script.GlideMemoryRecord</td>
</tr>
<tr>
<td>GlideMemorySwap</td>
<td>Packages.com.glide.ui.monitor.MemorySwap</td>
</tr>
<tr>
<td>GlideMemoryTable</td>
<td>Packages.com.glide.util.MemoryTable</td>
</tr>
<tr>
<td>GlideMemoryTotal</td>
<td>Packages.com.glide.ui.monitor.MemoryTotal</td>
</tr>
<tr>
<td>GlideMetaData</td>
<td>Packages.com.glide.script.MetaData</td>
</tr>
<tr>
<td>GlideMIDServerInfoAccessor</td>
<td>Packages.com.glide.script.MIDServerInfoAccessor</td>
</tr>
<tr>
<td>GlideMobileExtensions</td>
<td>Packages.com.glide.ui.MobileExtensions</td>
</tr>
<tr>
<td>GlideModule</td>
<td>Packages.com.glide.sys.Module</td>
</tr>
<tr>
<td>GlideMultipleAction</td>
<td>Packages.com.glide.db.MultipleAction</td>
</tr>
<tr>
<td>GlideMultipleDelete</td>
<td>Packages.com.glide.db.MultipleDelete</td>
</tr>
<tr>
<td>GlideMultipleInsert</td>
<td>Packages.com.glide.db.MultipleInsert</td>
</tr>
<tr>
<td>GlideMultipleUpdate</td>
<td>Packages.com.glide.db.MultipleUpdate</td>
</tr>
<tr>
<td>GlideMutex</td>
<td>Packages.com.glide.sys.lock.Mutex</td>
</tr>
<tr>
<td>GlideMySQLWatch</td>
<td>Packages.com.glide.sys.stats.MySQLWatch</td>
</tr>
<tr>
<td>GlideNumber</td>
<td>Packages.com.glide.script.glide_elements.</td>
</tr>
<tr>
<td>GlideNumberManager</td>
<td>Packages.com.glide.db.NumberManager</td>
</tr>
<tr>
<td>GlideObjectManager</td>
<td>Packages.com.glide.glideobject.GlideObject</td>
</tr>
<tr>
<td>GlideObjectUtil</td>
<td>Packages.com.glide.util.ObjectUtil</td>
</tr>
</tbody>
</table>
### GlideScriptable object replacement list (continued)

<table>
<thead>
<tr>
<th>GlideScriptable Class</th>
<th>Packages Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideOrderingDefinitionCreator</td>
<td>Packages.com.glide.sorting.OrderingDefinitionCreator</td>
</tr>
<tr>
<td>GlideOrderingManager</td>
<td>Packages.com.glide.sorting.OrderingManager</td>
</tr>
<tr>
<td>GlideOverLoadedChoices</td>
<td>Packages.com.glide.script.OverLoadedChoices</td>
</tr>
<tr>
<td>GlidePivotTableSummaryTableWriter</td>
<td>Packages.com.glide.ui.chart.dataset.PivotTableSummaryWriter</td>
</tr>
<tr>
<td>GlidePlugin</td>
<td>Packages.com.glide.sys.Plugin</td>
</tr>
<tr>
<td>GlidePluginManager</td>
<td>Packages.com.glide.sys.PluginManager</td>
</tr>
<tr>
<td>GlidePluginManagerWorker</td>
<td>Packages.com.glide.sys.PluginManagerWorker</td>
</tr>
<tr>
<td>GlidePluginUtils</td>
<td>Packages.com.glide.sys.PluginUtils</td>
</tr>
<tr>
<td>GlidePOP3Reader</td>
<td>Packages.com.glide.notification.inbound.POP3Reader</td>
</tr>
<tr>
<td>GlidePOP3ReaderJob</td>
<td>Packages.com.glide.job.POP3ReaderJob</td>
</tr>
<tr>
<td>GlidePopup</td>
<td>Packages.com.glide.ui.Popup</td>
</tr>
<tr>
<td>GlidePriceGenerator</td>
<td>Packages.com.glide.currency.PriceGenerator</td>
</tr>
<tr>
<td>GlidePriceLoader</td>
<td>Packages.com.glide.currency.PriceLoader</td>
</tr>
<tr>
<td>GlideProcessor</td>
<td>Packages.com.glide.processors.Processor</td>
</tr>
<tr>
<td>GlideProgressMonitor</td>
<td>Packages.com.glide.worker.ProgressMonitor</td>
</tr>
<tr>
<td>GlideProgressWorker</td>
<td>Packages.com.glide.worker.ProgressWorker</td>
</tr>
<tr>
<td>GlideProperties</td>
<td>Packages.com.glide.util.GlideProperties</td>
</tr>
<tr>
<td>GlideProperty</td>
<td>Packages.com.glide.util.GlideProperty</td>
</tr>
<tr>
<td>GlidePublicPage</td>
<td>Packages.com.glide.ui.PublicPage</td>
</tr>
<tr>
<td>GlideQueryBreadcrumbs</td>
<td>Packages.com.glide.misc.QueryBreadcrumbs</td>
</tr>
<tr>
<td>GlideQueryCondition</td>
<td>Packages.com.glide.db.conditions.QueryCondition</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>GlideQueryFormatter</td>
<td>Packages.com.glide.ui.jelly.tags.form.QueryFormatter</td>
</tr>
<tr>
<td>GlideQueryString</td>
<td>Packages.com.glide.db.QueryString</td>
</tr>
<tr>
<td>GlideQueryTerm</td>
<td>Packages.com.glide.db.QueryTerm</td>
</tr>
<tr>
<td>GlideRecord</td>
<td>Packages.com.glide.script.GlideRecord</td>
</tr>
<tr>
<td>GlideRecordCache</td>
<td>Packages.com.glide.sys.RecordCache</td>
</tr>
<tr>
<td>GlideRecordEnsurer</td>
<td>Packages.com.glide.misc.RecordEnsurer</td>
</tr>
<tr>
<td>GlideRecordFactory</td>
<td>Packages.com.glide.script.GlideRecordFactory</td>
</tr>
<tr>
<td>GlideRecordKeySetLoader</td>
<td>Packages.com.glide.script.GlideRecordKeySetLoader</td>
</tr>
<tr>
<td>GlideRecordLock</td>
<td>Packages.com.glide.script.GlideRecordLock</td>
</tr>
<tr>
<td>GlideRecordPopupGenerator</td>
<td>Packages.com.glide.calendar.RecordPopupGenerator</td>
</tr>
<tr>
<td>GlideRecordRollback</td>
<td>Packages.com.glide.script.GlideRecordRollback</td>
</tr>
<tr>
<td>GlideRecordSet</td>
<td>Packages.com.glide.script.GlideRecordSet</td>
</tr>
<tr>
<td>GlideRecordSimpleSerializer</td>
<td>Packages.com.glide.script.GlideRecordSimpleSerializer</td>
</tr>
<tr>
<td>GlideRecordXMLSerializer</td>
<td>Packages.com.glide.script.GlideRecordXMLSerializer</td>
</tr>
<tr>
<td>GlideReferenceField</td>
<td>Packages.com.glide.script.ReferenceField</td>
</tr>
<tr>
<td>GlideRegexUtil</td>
<td>Packages.com.glide.util.RegexUtil</td>
</tr>
<tr>
<td>GlideRegisterEscalationEvents</td>
<td>Packages.com.glide.fixes.RegisterEscalationEvents</td>
</tr>
<tr>
<td>GlideRelatedListReconciler</td>
<td>Packages.com.glide.misc.RelatedListReconciler</td>
</tr>
<tr>
<td>GlideRelationship</td>
<td>Packages.com.glide.sys.Relationship</td>
</tr>
<tr>
<td>GlideRelationships</td>
<td>Packages.com.glide.db.Relationships</td>
</tr>
<tr>
<td>GlideRelationshipUtil</td>
<td>Packages.com.glide.sys.RelationshipUtil</td>
</tr>
<tr>
<td>GlideRemoteGlideRecord</td>
<td>Packages.com.glide.communications.RemoteGlideRecord</td>
</tr>
<tr>
<td>GlideRenderProperties</td>
<td>Packages.com.glide.ui.RenderProperties</td>
</tr>
<tr>
<td>GlideReplaceUpdateFiles</td>
<td>Packages.com.glide.util.ReplaceUpdateFiles</td>
</tr>
<tr>
<td>GlideReplicationEngine</td>
<td>Packages.com.glide.replicator.ReplicationEngine</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>GlideRequestMap</td>
<td>Packages.com.glide.util.RequestMap</td>
</tr>
<tr>
<td>GlideRevertToOutOfBox</td>
<td>Packages.com.glide.update.RevertToOutOfBox</td>
</tr>
<tr>
<td>GlideRhinoHelper</td>
<td>Packages.com.glide.script.GlideRhinoHelper</td>
</tr>
<tr>
<td>GlideRhinoScope</td>
<td>Packages.com.glide.script.RhinoScope</td>
</tr>
<tr>
<td>GlideRhinoScopeHandler</td>
<td>Packages.com.glide.script.GlideRhinoScopeHandler</td>
</tr>
<tr>
<td>GlideRhinoTestCase</td>
<td>Packages.com.glide.autotester.RhinoTestCase</td>
</tr>
<tr>
<td>GlideRRDBAlertProcessor</td>
<td>Packages.com.glide.rrdb.alerts.RRDBAlertProcessor</td>
</tr>
<tr>
<td>GlideRRDBDefinition</td>
<td>Packages.com.glide.rrdb.RRDBDefinition</td>
</tr>
<tr>
<td>GlideRunScriptJob</td>
<td>Packages.com.glide.job.RunScriptJob</td>
</tr>
<tr>
<td>GlideSchedule</td>
<td>Packages.com.glide.schedules.Schedule</td>
</tr>
<tr>
<td>GlideScheduleDateTime</td>
<td>Packages.com.glide.glideobject.ScheduleDateTime</td>
</tr>
<tr>
<td>GlideScheduleDateTimeSpan</td>
<td>Packages.com.glide.schedules.ScheduleDateTimeSpan</td>
</tr>
<tr>
<td>GlideScheduleItem</td>
<td>Packages.com.glide.schedules.ScheduleItem</td>
</tr>
<tr>
<td>GlideScheduler</td>
<td>Packages.com.glide.schedule.GlideScheduler</td>
</tr>
<tr>
<td>GlideScheduleTimeMap</td>
<td>Packages.com.glide.schedules.ScheduleTimeMap</td>
</tr>
<tr>
<td>GlideScriptedProgressWorker</td>
<td>Packages.com.glide.worker.ScriptedProgressWorker</td>
</tr>
<tr>
<td>GlideScriptEvaluator</td>
<td>Packages.com.glide.script.ScriptEvaluator</td>
</tr>
<tr>
<td>GlideScriptListener</td>
<td>Packages.com.glide.listener.ScriptListener</td>
</tr>
<tr>
<td>GlideScriptProcessor</td>
<td>Packages.com.glide.processors.ScriptProcessor</td>
</tr>
<tr>
<td>GlideScriptRecordUtil</td>
<td>Packages.com.glide.script.GlideRecordUtil</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>GlideScriptSystemUtilDB</td>
<td>Packages.com.glide.script.GlideSystemUtilDB</td>
</tr>
<tr>
<td>GlideScriptViewManager</td>
<td>Packages.com.glide.ui.ViewManager</td>
</tr>
<tr>
<td>GlideSearchQueryFormatter</td>
<td>Packages.com.glide.text_search.SearchQueryFormatter</td>
</tr>
<tr>
<td>GlideSelfCleaningMutex</td>
<td>Packages.com.glide.sys.lock.SelfCleaningMutex</td>
</tr>
<tr>
<td>GlideServiceAPIWrapper</td>
<td>Packages.com.glide.service_api.ServiceAPIWrapper</td>
</tr>
<tr>
<td>GlideServlet</td>
<td>Packages.com.glide.ui.GlideServlet</td>
</tr>
<tr>
<td>GlideServletRequest</td>
<td>Packages.com.glide.ui.GlideServletRequest</td>
</tr>
<tr>
<td>GlideServletStatus</td>
<td>Packages.com.glide.ui.ServletStatus</td>
</tr>
<tr>
<td>GlideSession</td>
<td>Packages.com.glide.sys.GlideSession</td>
</tr>
<tr>
<td>GlideSessionDebug</td>
<td>Packages.com.glide.sys.SessionDebug</td>
</tr>
<tr>
<td>GlideSessions</td>
<td>Packages.com.glide.ui.Sessions</td>
</tr>
<tr>
<td>GlideSessionSandbox</td>
<td>Packages.com.glide.script.GlideSessionSandbox</td>
</tr>
<tr>
<td>GlideShellCommand</td>
<td>Packages.com.glide.util.ShellCommand</td>
</tr>
<tr>
<td>GlideSimmerDown</td>
<td>Packages.com.glide.db.change.command.SimmerDown</td>
</tr>
<tr>
<td>GlideSimmerUp</td>
<td>Packages.com.glide.db.change.command.SimmerUp</td>
</tr>
<tr>
<td>GlideSimpleDateFormatEx</td>
<td>Packages.com.glide.util.SimpleDateFormatEx</td>
</tr>
<tr>
<td>GlideSimpleHTTPClient</td>
<td>Packages.com.glide.communications.SimpleHTTPClient</td>
</tr>
<tr>
<td>GlideSimpleScriptListener</td>
<td>Packages.com.glide.listener.SimpleScriptListener</td>
</tr>
<tr>
<td>GlideSMTPConnection</td>
<td>Packages.com.glide.notification.outbound.SMTPConnection</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## GlideScriptable object replacement list (continued)

<table>
<thead>
<tr>
<th>GlideScriptable Class</th>
<th>Packages Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideSMTPSender</td>
<td>Packages.com.glide.notification.outbound</td>
</tr>
<tr>
<td>GlideSMTPSenderJob</td>
<td>Packages.com.glide.job.SMTPSenderJob</td>
</tr>
<tr>
<td>GlideSOAPDocument</td>
<td>Packages.com.glide.communications.soap</td>
</tr>
<tr>
<td>GlideSOAPRequest</td>
<td>Packages.com.glide.communications.soap</td>
</tr>
<tr>
<td>GlideSOAPResponse</td>
<td>Packages.com.glide.communications.soap</td>
</tr>
<tr>
<td>GlideSOAPSecurity</td>
<td>Packages.com.glide.processors.soap.SOAPSecurity</td>
</tr>
<tr>
<td>GlideSOAPSigner</td>
<td>Packages.com.glide.communications.soap</td>
</tr>
<tr>
<td>GlideSocket</td>
<td>Packages.com.glide.script.proxy.Socket</td>
</tr>
<tr>
<td>GlidesoftGlideAttributesImpl</td>
<td>Packages.com.glidesoft.util.xml.GlideAttribute</td>
</tr>
<tr>
<td>GlidesoftXMLMemoryTable</td>
<td>Packages.com.glidesoft.util.xml.XMLMemoryTable</td>
</tr>
<tr>
<td>GlideSQLChildMonitor</td>
<td>Packages.com.glide.monitor.sql.SQLChildMonitor</td>
</tr>
<tr>
<td>GlideSQLDebug</td>
<td>Packages.com.glide.ui.diagnostics.SQLDebug</td>
</tr>
<tr>
<td>GlideSQLDeleteMonitor</td>
<td>Packages.com.glide.monitor.sql.SQLDeleteMonitor</td>
</tr>
<tr>
<td>GlideSQLInsertMonitor</td>
<td>Packages.com.glide.monitor.sql.SQLInsertMonitor</td>
</tr>
<tr>
<td>GlideSQLResponseMonitor</td>
<td>Packages.com.glide.monitor.sql.SQLResponseMonitor</td>
</tr>
<tr>
<td>GlideSQLSelectMonitor</td>
<td>Packages.com.glide.monitor.sql.SQLSelectMonitor</td>
</tr>
<tr>
<td>GlideSQLUpdateMonitor</td>
<td>Packages.com.glide.monitor.sql.SQLUpdateMonitor</td>
</tr>
<tr>
<td>GlideSSHClient</td>
<td>Packages.com.glide.communications.SSHClient</td>
</tr>
<tr>
<td>GlideStack</td>
<td>Packages.com.glide.sys.GlideStack</td>
</tr>
<tr>
<td>GlideStatistician</td>
<td>Packages.com.glide.sys.stats.Statistician</td>
</tr>
<tr>
<td>GlideStatsInfo</td>
<td>Packages.com.glide.monitor.StatsInfo</td>
</tr>
<tr>
<td>GlideStatus</td>
<td>Packages.com.glide.util.GlideStatus</td>
</tr>
<tr>
<td>GlideStopWatch</td>
<td>Packages.com.glide.util.StopWatch</td>
</tr>
<tr>
<td>GlideStorageUtils</td>
<td>Packages.com.glide.db.meta.StorageUtils</td>
</tr>
<tr>
<td>GlideStringCache</td>
<td>Packages.com.glide.sys.cache.StringCache</td>
</tr>
<tr>
<td>GlideStringInputStream</td>
<td>Packages.com.glide.util.StringInputStream</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>GlideStringList</td>
<td>Packages.com.glide.collections.StringList</td>
</tr>
<tr>
<td>GlideStringUtil</td>
<td>Packages.com.glide.util.StringUtil</td>
</tr>
<tr>
<td>GlideSubQuery</td>
<td>Packages.com.glide.db.conditions.SubQuery</td>
</tr>
<tr>
<td>GlideSubstituteURL</td>
<td>Packages.com.glide.notification.substitution.url</td>
</tr>
<tr>
<td>GlideSummaryTableGroupReader</td>
<td>Packages.com.glide.ui.chart.dataset.SummaryTableGroupReader</td>
</tr>
<tr>
<td>GlideSummaryTableOrderedReader</td>
<td>Packages.com.glide.ui.chart.dataset.SummaryTableOrderedReader</td>
</tr>
<tr>
<td>GlideSummaryTableReader</td>
<td>Packages.com.glide.ui.chart.dataset.SummaryTableReader</td>
</tr>
<tr>
<td>GlideSummaryTableWriter</td>
<td>Packages.com.glide.ui.chart.dataset.SummaryTableWriter</td>
</tr>
<tr>
<td>GlideSynchronizedLRUCache</td>
<td>Packages.com.glide.sys.cache.SynchronizedLRUCache</td>
</tr>
<tr>
<td>GlideSysAttachment</td>
<td>Packages.com.glide.ui.SysAttachment</td>
</tr>
<tr>
<td>GlideSysAttachmentInputStream</td>
<td>Packages.com.glide.ui.SysAttachmentInputStream</td>
</tr>
<tr>
<td>GlideSysBRThreadMonitor</td>
<td>Packages.com.glide.monitor.threads.SysBRThreadMonitor</td>
</tr>
<tr>
<td>GlideSysChoice</td>
<td>Packages.com.glide.script.SysChoice</td>
</tr>
<tr>
<td>GlideSysConcurrencyMonitor</td>
<td>Packages.com.glide.monitor.threads.SysConcurrencyMonitor</td>
</tr>
<tr>
<td>GlideSysCPUPThreadMonitor</td>
<td>Packages.com.glide.monitor.threads.SysCPUPThreadMonitor</td>
</tr>
<tr>
<td>GlideSysDateUtil</td>
<td>Packages.com.glide.sys.util.SysDateUtil</td>
</tr>
<tr>
<td>GlideSysDBThreadMonitor</td>
<td>Packages.com.glide.monitor.threads.SysDBThreadMonitor</td>
</tr>
<tr>
<td>GlideSysField</td>
<td>Packages.com.glide.db.SysField</td>
</tr>
<tr>
<td>GlideSysForm</td>
<td>Packages.com.glide.ui.SysForm</td>
</tr>
<tr>
<td>GlideSysForms</td>
<td>Packages.com.glide.ui.SysForms</td>
</tr>
<tr>
<td>GlideSysList</td>
<td>Packages.com.glide.ui.SysList</td>
</tr>
<tr>
<td>GlideSysListControl</td>
<td>Packages.com.glide.ui.SysListControl</td>
</tr>
<tr>
<td>GlideSysLog</td>
<td>Packages.com.glide.sys.SysLog</td>
</tr>
<tr>
<td>GlideSYSMany2Many</td>
<td>Packages.com.glide.db.SYSMany2Many</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>--------------------------------------------------------------------</td>
</tr>
<tr>
<td>GlideSysSection</td>
<td>Packages.com.glide.ui.SysSection</td>
</tr>
<tr>
<td>GlideSysSemaphore</td>
<td>Packages.com.glide.sys.util.SysSemaphore</td>
</tr>
<tr>
<td>GlideSystem</td>
<td>Packages.com.glide.script.GlideSystem</td>
</tr>
<tr>
<td>GlideSystemDateUtil</td>
<td>Packages.com.glide.script.system.GlideSystemDateUtil</td>
</tr>
<tr>
<td>GlideSystemUtil</td>
<td>Packages.com.glide.util.SystemUtil</td>
</tr>
<tr>
<td>GlideSystemUtilDB</td>
<td>Packages.com.glide.script.system.GlideSystemUtilDB</td>
</tr>
<tr>
<td>GlideSystemUtilScript</td>
<td>Packages.com.glide.script.system.GlideSystemUtilScript</td>
</tr>
<tr>
<td>GlideSysThreadMonitor</td>
<td>Packages.com.glide.monitor.threads.SysThreadMonitor</td>
</tr>
<tr>
<td>GlideTable</td>
<td>Packages.com.glide.db.meta.Table</td>
</tr>
<tr>
<td>GlideTableChoiceList</td>
<td>Packages.com.glide.script.TableChoiceList</td>
</tr>
<tr>
<td>GlideTableCleaner</td>
<td>Packages.com.glide.misc.TableCleaner</td>
</tr>
<tr>
<td>GlideTableCleaner.Job</td>
<td>Packages.com.glide.job.TableCleanerJob</td>
</tr>
<tr>
<td>GlideTableCreator</td>
<td>Packages.com.glide.db.impex.TableCreator</td>
</tr>
<tr>
<td>GlideTableDescriptor</td>
<td>Packages.com.glide.db.TableDescriptor</td>
</tr>
<tr>
<td>GlideTableGroupMover</td>
<td>Packages.com.glide.db.auxiliary.TableGroupMover</td>
</tr>
<tr>
<td>GlideTableManager</td>
<td>Packages.com.glide.db.TableManager</td>
</tr>
<tr>
<td>GlideTableMover</td>
<td>Packages.com.glide.db.auxiliary.TableMover</td>
</tr>
<tr>
<td>GlideTableParentChange</td>
<td>Packages.com.glide.db.table.TableParentChange</td>
</tr>
<tr>
<td>GlideTableParentColumnChange</td>
<td>Packages.com.glide.db.table.TableParentColumnChange</td>
</tr>
<tr>
<td>GlideTaskToken</td>
<td>Packages.com.glide.execution_plan.TaskToken</td>
</tr>
<tr>
<td>GlideTemplate</td>
<td>Packages.com.glide.script.Template</td>
</tr>
<tr>
<td>GlideTestAgent</td>
<td>Packages.com.glide.autotester.GlideTestAgent</td>
</tr>
<tr>
<td>GlideTextIndexEvent</td>
<td>Packages.com.glide.ts.event.TextIndexEvent</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GlideThreadAttributes</td>
<td>Packages.com.glide.ui.GlideThreadAttributes</td>
</tr>
<tr>
<td>GlideThreadUtil</td>
<td>Packages.com.glide.util.ThreadUtil</td>
</tr>
<tr>
<td>GlideTime</td>
<td>Packages.com.glide.glideobject.GlideTime</td>
</tr>
<tr>
<td>GlideTimelineFrameSeparator</td>
<td>Packages.com.glide.schedules.TimelineFrameSeparator</td>
</tr>
<tr>
<td>GlideTimelineItem</td>
<td>Packages.com.glide.schedules.TimelineItem</td>
</tr>
<tr>
<td>GlideTimelineSpan</td>
<td>Packages.com.glide.schedules.TimelineSpan</td>
</tr>
<tr>
<td>GlideTomcatHelper</td>
<td>Packages.com.glide.startup.TomcatHelper</td>
</tr>
<tr>
<td>GlideTransaction</td>
<td>Packages.com.glide.sys.Transaction</td>
</tr>
<tr>
<td>GlideTransactionManager</td>
<td>Packages.com.glide.sys.TransactionManager</td>
</tr>
<tr>
<td>GlideTransferAuditDataHelper</td>
<td>Packages.com.glide.audit.TransferAuditDataHelper</td>
</tr>
<tr>
<td>GlideTransformer</td>
<td>Packages.com.glide.db.impex.transformer</td>
</tr>
<tr>
<td>GlideTreePicker</td>
<td>Packages.com.glide.ui.TreePicker</td>
</tr>
<tr>
<td>GlideTSAnalysisViewer</td>
<td>Packages.com.glide.ts.cluster.TSAnalysisViewer</td>
</tr>
<tr>
<td>GlideTSAnalyticsProcessor</td>
<td>Packages.com.glide.ts.cluster.TSAnalyticsProcessor</td>
</tr>
<tr>
<td>GlideTSChainsHandler</td>
<td>Packages.com.glide.ts.trends.TSChainsHandler</td>
</tr>
<tr>
<td>GlideTSChainsLoader</td>
<td>Packages.com.glide.ts.trends.TSChainsLoader</td>
</tr>
<tr>
<td>GlideTSChainsPusher</td>
<td>Packages.com.glide.ts.trends.TSChainsPusher</td>
</tr>
<tr>
<td>GlideTSChainsSummarizer</td>
<td>Packages.com.glide.ts.trends.TSChainsSummarizer</td>
</tr>
<tr>
<td>GlideTSClusterDefinitions</td>
<td>Packages.com.glide.ts.cluster.TSClusterDefinitions</td>
</tr>
<tr>
<td>GlideTSDebug</td>
<td>Packages.com.glide.ts.util.TSDebug</td>
</tr>
<tr>
<td>GlideTSDidYouMean</td>
<td>Packages.com.glide.ts.util.TSDidYouMean</td>
</tr>
<tr>
<td>GlideTSGlobalKeywordSummarizer</td>
<td>Packages.com.glide.ts.trends.TSGlobalKeywordSummarizer</td>
</tr>
<tr>
<td>GlideTSGlobalKeywordSummarizer</td>
<td>Packages.com.glide.ts.trends.TSGlobalKeywordSummarizer</td>
</tr>
<tr>
<td>GlideTSIndexStatistician</td>
<td>Packages.com.glide.ts.stats.TSIndexStatistician</td>
</tr>
<tr>
<td>GlideTSIndexStopGenerator</td>
<td>Packages.com.glide.ts.stats.TSIndexStopGenerator</td>
</tr>
<tr>
<td>GlideTSIndexTables</td>
<td>Packages.com.glide.ts.indexer.TSIndexTables</td>
</tr>
<tr>
<td>GlideTSKeywordHandler</td>
<td>Packages.com.glide.ts.trends.TSKeywordHandler</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### GlideScriptable object replacement list (continued)

<table>
<thead>
<tr>
<th>GlideScriptable Class</th>
<th>Packages Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlideTSKeywordLoader</td>
<td>Packages.com.glide.ts.trends.TSKeywordLoader</td>
</tr>
<tr>
<td>GlideTSKeywordPusher</td>
<td>Packages.com.glide.ts.trends.TSKeywordPusher</td>
</tr>
<tr>
<td>GlideTSMoversViewer</td>
<td>Packages.com.glide.ts.cluster.TSMoversViewer</td>
</tr>
<tr>
<td>GlideTSSearchStatistician</td>
<td>Packages.com.glide.ts.stats.TSSearchStatistician</td>
</tr>
<tr>
<td>GlideTSSearchSummary</td>
<td>Packages.com.glide.ts.trends.TSSearchSummary</td>
</tr>
<tr>
<td>GlideTSTopSearches</td>
<td>Packages.com.glide.ts.util.TSTopSearches</td>
</tr>
<tr>
<td>GlideTSUtil</td>
<td>Packages.com.glide.ts.util.TSUtil</td>
</tr>
<tr>
<td>GlideTSVersion</td>
<td>Packages.com.glide.ts.TSVersion</td>
</tr>
<tr>
<td>GlideUIAction</td>
<td>Packages.com.glide.ui.action.UIAction</td>
</tr>
<tr>
<td>GlideUISession</td>
<td>Packages.com.glide.ui.Session</td>
</tr>
<tr>
<td>GlideUnloader</td>
<td>Packages.com.glide.db.impex.Unloader</td>
</tr>
<tr>
<td>GlideUpdateManager2</td>
<td>Packages.com.glide.update.UpdateManager2</td>
</tr>
<tr>
<td>GlideUpdateSet</td>
<td>Packages.com.glide.update.UpdateSet</td>
</tr>
<tr>
<td>GlideUpdateSetController</td>
<td>Packages.com.glide.system_update_set.UpdateSet</td>
</tr>
<tr>
<td>GlideUpdateSetPreviewer</td>
<td>Packages.com.glide.update.UpdateSetPreviewer</td>
</tr>
<tr>
<td>GlideUpdateSetWorker</td>
<td>Packages.com.glide.update.UpdateSetWorker</td>
</tr>
<tr>
<td>GlideUpdateSyncher</td>
<td>Packages.com.glide.policy.UpdateSyncher</td>
</tr>
<tr>
<td>GlideUpdateTableChoiceList</td>
<td>Packages.com.glide.script.UpdateTableChoiceList</td>
</tr>
<tr>
<td>GlideUpgrade</td>
<td>Packages.com.glide.sys.Upgrade</td>
</tr>
<tr>
<td>GlideUpgradeArtifactManager</td>
<td>Packages.com.glide.misc.UpgradeArtifactManager</td>
</tr>
<tr>
<td>GlideUpgradeLog</td>
<td>Packages.com.glide.update.UpgradeLog</td>
</tr>
<tr>
<td>GlideURI</td>
<td>Packages.com.glide.ui.GlideURI</td>
</tr>
<tr>
<td>GlideURL</td>
<td>Packages.com.glide.util.GlideURL</td>
</tr>
<tr>
<td>GlideURLUTF8Encoder</td>
<td>Packages.com.glide.util.URLUTF8Encoder</td>
</tr>
<tr>
<td>GlideURLUtil</td>
<td>Packages.com.glide.util.URLUtil</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>----------------------------------------------------</td>
</tr>
<tr>
<td>GlideUser</td>
<td>Packages.com.glide.sys.User</td>
</tr>
<tr>
<td>GlideUserGroup</td>
<td>Packages.com.glide.sys.UserGroup</td>
</tr>
<tr>
<td>GlideUtil</td>
<td>Packages.com.glide.util.GlideUtil</td>
</tr>
<tr>
<td>GlideWarDeleter</td>
<td>Packages.com.glide.misc.WarDeleter</td>
</tr>
<tr>
<td>GlideWarDownloader</td>
<td>Packages.com.glide.misc.WarDownloader</td>
</tr>
<tr>
<td>GlideWhiteListManager</td>
<td>Packages.com.glide.script.api.GlideWhiteListManager</td>
</tr>
<tr>
<td>GlideWikiModel</td>
<td>Packages.com.glide.wiki.GlideWikiModel</td>
</tr>
<tr>
<td>GlideWorkerThread</td>
<td>Packages.com.glide.worker.WorkerThread</td>
</tr>
<tr>
<td>GlideWorkerThreadManager</td>
<td>Packages.com.glide.sys.WorkerThreadManager</td>
</tr>
<tr>
<td>GlideWSClient</td>
<td>Packages.com.glide.util.WSClient</td>
</tr>
<tr>
<td>GlideWSDefinition</td>
<td>Packages.com.glide.wsdlreader.WSDefinition</td>
</tr>
<tr>
<td>GlideWSDLReader</td>
<td>Packages.com.glide.wsdlreader.GlideWSDLReader</td>
</tr>
<tr>
<td>GlideXMLChoiceListSerializer</td>
<td>Packages.com.glide.choice.XMLChoiceListSerializer</td>
</tr>
<tr>
<td>GlideXMLDocument</td>
<td>Packages.com.glide.util.XMLDocument</td>
</tr>
<tr>
<td>GlideXMLElementIterator</td>
<td>Packages.com.glide.util.XMLElementIterator</td>
</tr>
<tr>
<td>GlideXMLGlideRecordSerializer</td>
<td>Packages.com.glide.processors.xmlhttp.XMLGlideRecordSerializer</td>
</tr>
<tr>
<td>GlideXMLParameters</td>
<td>Packages.com.glide.util.XMLParameters</td>
</tr>
<tr>
<td>GlideXMLStats</td>
<td>Packages.com.glide.ui.XMLStats</td>
</tr>
<tr>
<td>GlideXMLSysMetaSerializer</td>
<td>Packages.com.glide.processors.xmlhttp.XMLSysMetaSerializer</td>
</tr>
<tr>
<td>GlideXMLUtil</td>
<td>Packages.com.glide.util.XMLUtil</td>
</tr>
<tr>
<td>GlideZipUtil</td>
<td>Packages.com.glide.util.ZipUtil</td>
</tr>
<tr>
<td>RhinoHelper</td>
<td>Packages.com.glide.script.RhinoHelper</td>
</tr>
<tr>
<td>SecurelyAccess</td>
<td>Packages.com.glide.sys.util.SecurelyAccess</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>ServiceAPI</td>
<td>Packages.com.glide.service_api.ServiceAPI</td>
</tr>
<tr>
<td>SncAddress32Bit</td>
<td>Packages.com.snc.commons.networks.Address32Bit</td>
</tr>
<tr>
<td>SncAliasApplier</td>
<td>Packages.com.snc.field_normalization.AliasApplier</td>
</tr>
<tr>
<td>SncAppFiles</td>
<td>Packages.com.snc.apps.api.AppFiles</td>
</tr>
<tr>
<td>SncApplicationFileListener</td>
<td>Packages.com.snc.apps.db.ApplicationFileListener</td>
</tr>
<tr>
<td>SncAppsUI</td>
<td>Packages.com.snc.apps.api.AppsUI</td>
</tr>
<tr>
<td>SncASensor</td>
<td>Packages.com.snc.discovery.sensor.ASensor</td>
</tr>
<tr>
<td>SncAuthentication</td>
<td>Packages.com.snc.authentication.digest.Authentication</td>
</tr>
<tr>
<td>SncBaselineCMDB</td>
<td>Packages.com.snc.cmdb.BaselineCMDB</td>
</tr>
<tr>
<td>SncBulkCopy</td>
<td>Packages.com.snc.ha.clone.BulkCopy</td>
</tr>
<tr>
<td>SncClassifiedProcess</td>
<td>Packages.com.snc.discovery.sensor.ClassifiedProcess</td>
</tr>
<tr>
<td>SncClassify</td>
<td>Packages.com.snc.discovery.sensor.snmp.Classify</td>
</tr>
<tr>
<td>SncCloneController</td>
<td>Packages.com.snc.ha.clone.CloneController</td>
</tr>
<tr>
<td>SncCloneInstance</td>
<td>Packages.com.snc.ha.clone.instance.Instance</td>
</tr>
<tr>
<td>SncCloneLogger</td>
<td>Packages.com.snc.ha.clone.CloneLogger</td>
</tr>
<tr>
<td>SncCloneTask</td>
<td>Packages.com.snc.ha.clone.CloneTask</td>
</tr>
<tr>
<td>SncCloneUtils</td>
<td>Packages.com.snc.ha.clone.CloneUtils</td>
</tr>
<tr>
<td>SncConfiguration</td>
<td>Packages.com.snc.field_normalization.db.Configuration</td>
</tr>
<tr>
<td>SncConfigurations</td>
<td>Packages.com.snc.field_normalization.Configurations</td>
</tr>
<tr>
<td>SncConnectionTest</td>
<td>Packages.com.snc.ha.connectivity.ConnectionTest</td>
</tr>
<tr>
<td>SncDBChangeManagerFactoryHA</td>
<td>Packages.com.snc.db.clone.change.DBCManagerFactory</td>
</tr>
<tr>
<td>SncDeviceHistory</td>
<td>Packages.com.snc.discovery.logging.DeviceHistory</td>
</tr>
<tr>
<td>SncDiscoveryCancel</td>
<td>Packages.com.snc.discovery.DiscoveryCancel</td>
</tr>
<tr>
<td>SncDiscoveryClassification</td>
<td>Packages.com.snc.discovery.DiscoveryClassification</td>
</tr>
<tr>
<td>SncDiscoveryLog</td>
<td>Packages.com.snc.discovery.logging.DiscoveryLog</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>----------------------------------------------------</td>
</tr>
<tr>
<td>SncDiscoveryRanges</td>
<td>Packages.com.snc.commons.networks.DiscoveryRanges</td>
</tr>
<tr>
<td>SncDiscoveryRangesDB</td>
<td>Packages.com.snc.discovery.DiscoveryRangesDB</td>
</tr>
<tr>
<td>SncDiscoveryReconciler</td>
<td>Packages.com.snc.discovery.database.DiscoveryReconciler</td>
</tr>
<tr>
<td>SncDiscoverySNMPClassification</td>
<td>Packages.com.snc.discovery.sensor.snmp.DiscoverySNMPClassification</td>
</tr>
<tr>
<td>SncDiscoveryUtils</td>
<td>Packages.com.snc.discovery.DiscoveryUtils</td>
</tr>
<tr>
<td>SncDropBackupTablesTask</td>
<td>Packages.com.snc.ha.clone.instance.DropBackupTablesTask</td>
</tr>
<tr>
<td>SncDropTablesTask</td>
<td>Packages.com.snc.ha.clone.DropTablesTask</td>
</tr>
<tr>
<td>SncEC2Properties</td>
<td>Packages.com.snc.ec2.EC2Properties</td>
</tr>
<tr>
<td>SncECMDBUtil</td>
<td>Packages.com.snc.cmdb.ECMDBUtil</td>
</tr>
<tr>
<td>SncElrondClient</td>
<td>Packages.com.snc.customer_logon.ElrondClient</td>
</tr>
<tr>
<td>SncExtantDataJob</td>
<td>Packages.com.snc.field_normalization.db.ExtantDataJob</td>
</tr>
<tr>
<td>SncExtantDataJobState</td>
<td>Packages.com.snc.field_normalization.ExtantDataJobState</td>
</tr>
<tr>
<td>SncFailoverController</td>
<td>Packages.com.snc.da.gateway.replication.FailoverController</td>
</tr>
<tr>
<td>SncFileTree</td>
<td>Packages.com.snc.apps.file.FileTree</td>
</tr>
<tr>
<td>SncGatewayCache</td>
<td>Packages.com.snc.da.gateway.GatewayCache</td>
</tr>
<tr>
<td>SncGatewayClone</td>
<td>Packages.com.snc.da.gateway.clone.GatewayClone</td>
</tr>
<tr>
<td>SncGatewayConnectivity</td>
<td>Packages.com.snc.da.gateway.GatewayConnectivity</td>
</tr>
<tr>
<td>SncGatewayPluginStartup</td>
<td>Packages.com.snc.da.gateway.replication.GatewayPluginStartup</td>
</tr>
<tr>
<td>SncGlideGateways</td>
<td>Packages.com.snc.da.gateway.GlideGateways</td>
</tr>
<tr>
<td>SncHAClone</td>
<td>Packages.com.snc.ha.clone.HAClone</td>
</tr>
<tr>
<td>SncHAConnectionTest</td>
<td>Packages.com.snc.ha.connectivity.HAConnectionTest</td>
</tr>
<tr>
<td>SncHADatabaseCheck</td>
<td>Packages.com.snc.ha.tablecheck.HADatabaseCheck</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>SncHAPairingUtils</td>
<td>Packages.com.snc.ha.HAPairingUtils</td>
</tr>
<tr>
<td>SncHAReplicationController</td>
<td>Packages.com.snc.ha.HAREplicationController</td>
</tr>
<tr>
<td>SncHAReplicationQueueSnapshotBuilder</td>
<td>Packages.com.snc.ha.tablecheck.HAREplicationQueue</td>
</tr>
<tr>
<td>SncHATableCheck</td>
<td>Packages.com.snc.ha.tablecheck.HATableCheck</td>
</tr>
<tr>
<td>SncHATableCheckThread</td>
<td>Packages.com.snc.ha.tablecheck.HATableCheckThread</td>
</tr>
<tr>
<td>SncHATableQuickCheck</td>
<td>Packages.com.snc.ha.tablecheck.HATableQuickCheck</td>
</tr>
<tr>
<td>SncHATableRepair</td>
<td>Packages.com.snc.ha.tablecheck.HATableRepair</td>
</tr>
<tr>
<td>SncHAUtils</td>
<td>Packages.com.snc.ha.HAUUtils</td>
</tr>
<tr>
<td>SncHostname</td>
<td>Packages.com.snc.discovery.utils.Hostname</td>
</tr>
<tr>
<td>SncInstanceClone</td>
<td>Packages.com.snc.ha.clone.instance.InstanceClone</td>
</tr>
<tr>
<td>SncInstanceConnectionTest</td>
<td>Packages.com.snc.ha.connectivity.InstanceConnection</td>
</tr>
<tr>
<td>SncInstanceRollback</td>
<td>Packages.com.snc.ha.clone.instance.InstanceRollback</td>
</tr>
<tr>
<td>SncIPAddressV4</td>
<td>Packages.com.snc.commons.networks.IPAddressV4</td>
</tr>
<tr>
<td>SncIPAddressV6</td>
<td>Packages.com.snc.commons.networks.IPAddressV6</td>
</tr>
<tr>
<td>SncIPAuthenticator</td>
<td>Packages.com.snc.ipauthenticator.IPAuthenticator</td>
</tr>
<tr>
<td>SncIPIterator</td>
<td>Packages.com.snc.commons.networks.IPIterator</td>
</tr>
<tr>
<td>SncIPList</td>
<td>Packages.com.snc.commons.networks.IPList</td>
</tr>
<tr>
<td>SncIPMetaCollection</td>
<td>Packages.com.snc.commons.networks.IPMetaCollection</td>
</tr>
<tr>
<td>SncIPNetmaskV4</td>
<td>Packages.com.snc.commons.networks.IPNetmaskV4</td>
</tr>
<tr>
<td>SncIPNetworkV4</td>
<td>Packages.com.snc.commons.networks.IPNetworkV4</td>
</tr>
<tr>
<td>SncIPRangeV4</td>
<td>Packages.com.snc.commons.networks.IPRangeV4</td>
</tr>
<tr>
<td>SncJRobinGraphDef</td>
<td>Packages.com.snc.jrobin.JRobinGraphDef</td>
</tr>
<tr>
<td>SncLayer7Connections</td>
<td>Packages.com.snc.discovery.Layer7Connections</td>
</tr>
<tr>
<td>SncMACAddress</td>
<td>Packages.com.snc.commons.networks.MACAddress</td>
</tr>
<tr>
<td>SncMACAddressMfr</td>
<td>Packages.com.snc.commons.networks.MACAddressMfr</td>
</tr>
<tr>
<td>SncMakeAndModel</td>
<td>Packages.com.snc.cmdb.MakeAndModel</td>
</tr>
<tr>
<td>GlideScriptable Class</td>
<td>Packages Call</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SncMIDConfigParameter</td>
<td>Packages.com.snc.commons.MIDConfigParameter</td>
</tr>
<tr>
<td>SncMIDServerRangesDB</td>
<td>Packages.com.snc.discovery.MIDServerRangesDB</td>
</tr>
<tr>
<td>SncNormalCoalescer</td>
<td>Packages.com.snc.field_normalization.NormalCoalescer</td>
</tr>
<tr>
<td>SncNormalizer</td>
<td>Packages.com.snc.field_normalization.Normalizer</td>
</tr>
<tr>
<td>SncNormalValueChanger</td>
<td>Packages.com.snc.field_normalization.NormalValueChanger</td>
</tr>
<tr>
<td>SncNotifySNC</td>
<td>Packages.com.snc.system.NotifySNC</td>
</tr>
<tr>
<td>SncOnCallRotation</td>
<td>Packages.com.snc.on_call_rotation.OnCallRotation</td>
</tr>
<tr>
<td>SncPendingValueCollector</td>
<td>Packages.com.snc.field_normalization.PendingValueCollector</td>
</tr>
<tr>
<td>SncPreferences</td>
<td>Packages.com.snc.field_normalization.Preferences</td>
</tr>
<tr>
<td>SncPrintServerHelper</td>
<td>Packages.com.snc.discovery.database.PrintServerHelper</td>
</tr>
<tr>
<td>SncProbe</td>
<td>Packages.com.snc.discovery.Probe</td>
</tr>
<tr>
<td>SncProbeRunTime</td>
<td>Packages.com.snc.discovery.perfmon.ProbeRunTime</td>
</tr>
<tr>
<td>SncRBSensorProcessor</td>
<td>Packages.com.snc.discovery_automation.RBSensorProcessor</td>
</tr>
<tr>
<td>SncReadTest</td>
<td>Packages.com.snc.ha.ReadTest</td>
</tr>
<tr>
<td>SncReclassifyCI</td>
<td>Packages.com.snc.cmdb.ReclassifyCI</td>
</tr>
<tr>
<td>SncRelationships</td>
<td>Packages.com.snc.cmdb.Relationships</td>
</tr>
<tr>
<td>SncReplicationAdvisor</td>
<td>Packages.com.snc.db.replicate.ReplicationAdvisor</td>
</tr>
<tr>
<td>SncReplicationEngine</td>
<td>Packages.com.snc.db.replicate.ReplicationEngine</td>
</tr>
<tr>
<td>SncReplicationQueue</td>
<td>Packages.com.snc.db.replicate.ReplicationQueue</td>
</tr>
<tr>
<td>SncRequestCredentials</td>
<td>Packages.com.snc.customer_logon.RequestCredentials</td>
</tr>
<tr>
<td>SncRuleApplier</td>
<td>Packages.com.snc.field_normalization.RuleApplier</td>
</tr>
<tr>
<td>SncRuleToPending</td>
<td>Packages.com.snc.field_normalization.RuleToPending</td>
</tr>
<tr>
<td>SncSAMCounter</td>
<td>Packages.com.snc.software_asset_management.SAMCounter</td>
</tr>
<tr>
<td>SncScheduleDropBackupTablesTask</td>
<td>Packages.com.snc.ha.clone.instance.ScheduleDropBackupTablesTask</td>
</tr>
<tr>
<td>SncScrapeIANAEnterpriseNumbers</td>
<td>Packages.com.snc.discovery.database.ScrapeIANAEnterpriseNumbers</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### GlideScriptable object replacement list (continued)

<table>
<thead>
<tr>
<th>GlideScriptable Class</th>
<th>Packages Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>SncScrapeIENICCodes</td>
<td>Packages.com.snc.discovery.database.ScrapeIENICCodes</td>
</tr>
<tr>
<td>SncSendNotificationTask</td>
<td>Packages.com.snc.ha.clone.instance.SendNotificationTask</td>
</tr>
<tr>
<td>SncSensorProcessor</td>
<td>Packages.com.snc.discovery.SensorProcessor</td>
</tr>
<tr>
<td>SncSerialNumber</td>
<td>Packages.com.snc.discovery.SerialNumber</td>
</tr>
<tr>
<td>SncSerialNumberList</td>
<td>Packages.com.snc.discovery.SerialNumberList</td>
</tr>
<tr>
<td>SncSessionMate</td>
<td>Packages.com.snc.discovery.SessionMate</td>
</tr>
<tr>
<td>SncSimmerControl</td>
<td>Packages.com.snc.ha.clone.instance.SimmerControl</td>
</tr>
<tr>
<td>SncTableEditor</td>
<td>Packages.com.snc.apps.api.TableEditor</td>
</tr>
<tr>
<td>SncTableRotation</td>
<td>Packages.com.snc.db.replicate.TableRotation</td>
</tr>
<tr>
<td>SncTableRotationExtension</td>
<td>Packages.com.snc.db.replicate.TableRotationExtension</td>
</tr>
<tr>
<td>SncTableRotationExtensions</td>
<td>Packages.com.snc.db.replicate.TableRotationExtensions</td>
</tr>
<tr>
<td>SncTableRotationWatcher</td>
<td>Packages.com.snc.db.replicate.TableRotationWatcher</td>
</tr>
<tr>
<td>SncTransformApplier</td>
<td>Packages.com.snc.field_normalization.TransformApplier</td>
</tr>
<tr>
<td>SncTreeBuilder</td>
<td>Packages.com.snc.apps.tree.TreeBuilder</td>
</tr>
<tr>
<td>SncTriggerSynchronizer</td>
<td>Packages.com.snc.automation.TriggerSynchronizer</td>
</tr>
<tr>
<td>SncValue</td>
<td>Packages.com.snc.field_normalization.db.Value</td>
</tr>
<tr>
<td>TestExtension</td>
<td>Packages.com.glide.junit.misc.TestExtension</td>
</tr>
<tr>
<td>UINotification</td>
<td>Packages.com.glide.ui.UINotification</td>
</tr>
</tbody>
</table>

### Packages Call Removal Tool error types

Possible error types generated by the Packages Call Removal Tool.

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class Name Could</td>
<td>A line of script has what appears to be a Packages call (for example, the line contains Packages.com.glide) but on</td>
<td>This error type generally does not present a problem, but is nevertheless flagged as an error in case the script references...</td>
</tr>
</tbody>
</table>
### Error types (continued)

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Be Parsed</td>
<td>examination there is insufficient text to parse out a class name that corresponds to a scriptable object name.</td>
<td>class that the parser, for whatever reason, cannot parse.</td>
</tr>
<tr>
<td>Class Does Not Exist</td>
<td>A line of script has an identifiable Packages call to a Java class that does not exist. This may be because it was incorrectly typed, or because the Java class no longer exists. Either way, the line of script is currently not doing anything and is potentially generating errors whenever it is run.</td>
<td>This error type usually identifies a script that is not doing what its author intended, or what it ever did. The original script should be revisited, and the invalid Packages call removed.</td>
</tr>
<tr>
<td>Class Is Not a Scriptable Class</td>
<td>A line of script has an identifiable Packages call to an existing ServiceNow Java class that is not currently marked as scriptable. The call cannot be replaced, because there is currently no corresponding scriptable name.</td>
<td>To convert this script, either rewrite the script so that it does not use the Java class or wait until ServiceNow, Inc., provides a scriptable name for the class.</td>
</tr>
<tr>
<td>Class Does Not Have a Scriptable Name</td>
<td>A line of script has an identifiable Packages call to an existing ServiceNow Java class that does not currently have a scriptable name.</td>
<td>This error is less likely to occur than a Class Is Not a Scriptable Class error; however, if it occurs, to convert this script, either rewrite the script so that it does not use the Java class or wait until ServiceNow, Inc., provides a scriptable name for the class.</td>
</tr>
<tr>
<td>Variable Name Being Assigned Conflicts With a Scriptable Name</td>
<td>A variable name has the same name as a scriptable name. For example, <code>var GlideSession = Packages.com.glide.sys.GlideSession.get();</code> would generate this error because the variable name <code>GlideSession</code> is the same as the scriptable name for the GlideSession Java class.</td>
<td>Before this script can be converted, the variable name must be changed wherever it occurs in the script. For this example, the line could be changed to <code>var gsession = Packages.com.glide.sys.GlideSession.get();</code> to remove the conflict. Also, be sure to change the variable name elsewhere in the script wherever it is used.</td>
</tr>
<tr>
<td>Internal Error - Unable To Find String</td>
<td>The tool is unable to find the script text it is currently evaluating for Packages call replacement.</td>
<td>This error type is very unlikely. If it does occur, examine your code and correct any errors. If the error message still appears, open an incident with Technical Support.</td>
</tr>
</tbody>
</table>
Error types (continued)

<table>
<thead>
<tr>
<th>Error message</th>
<th>Description</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>To Be Replaced</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Now® Experience UI Builder

Now® Experience UI Builder is a web user interface builder. Use UI Builder to build pages for CSM Configurable Workspace, App Engine Studio-generated workspaces and portals, or custom web experiences using Now® Experience Components and custom web components.

[i] Note:

UI Builder is not yet capable of building or configuring out-of-the-box Service portals, like Employee Center. Continue to use the Service Portal Designer instead.

UI Builder overview
UI Builder overview

UI Builder quick start
UI Builder quick start video

Key features

Create pages for workspace or portal experiences

Create or customize pages for workspace and portal experiences. A page is a collection of components that make up a workspace or portal UI.
Create a variant of a page

Create variants of pages to target experiences for different audiences. For example, you can create a home page for agents, and a variant for managers at the same URL.

Use a full library of components

Use the UI Builder library of components to build your pages. You can add them to your page. Configure them any way you need and then connect your company data to them.

Connect data to your components
Bind data to your components using data resources to dynamically expose your data from tables, records, or other elements on your page. Data resources let you reuse your components.

**Modify the layout of a page**

Create any type of page design by changing the layout of the page.

**Add styling to your component**

Use CSS to change the visual style of a component in your workspace or portal experience. For example, you can add a background color to a container component.
Get started

<table>
<thead>
<tr>
<th>Explore</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn about UI Builder concepts and features.</td>
<td>Learn how to build pages in UI Builder.</td>
</tr>
</tbody>
</table>

Exploring UI Builder

Explore the areas of Now® Experience UI Builder to get you set up and creating pages quickly.

Check out the UI Builder quick start topic to understand the basics of UI Builder. If you want a deeper understanding of UI Builder, go through the tutorial to learn how to create your first page.

Find information relating to UI Builder audiences, security and roles settings like application scope and domain separation, as well as experience settings.

See how you can leverage UI Builder with CSM Workspace and portal experiences.

UI Builder quick start

This quick start guides you through the process of creating your first page in Now® Experience UI Builder. Creating your first page is the first step in understanding how to build user interface pages for your workspace or custom portal experiences.

Before you begin
Role required: ui_builder_admin

In this UI Builder quick start, you perform the following tasks to build your first page in UI Builder.

- Start UI Builder.
- Create a page for your workspace or custom portal experience. For more information about creating pages, see Create a page.
- Change the layout of the page to have two columns. For more information about layouts, see Work with layouts.
• Build your page by adding two container components. For more information about components, see Work with components.
• Rename your container components in the content tree.
• Add a heading component in one container. Add an image component in a second container.
• Save your page.
• Preview your page to see how it looks in a browser.

Procedure
1. Navigate to Now Experience Framework > UI Builder. You can also type UI Builder directly in the Filter navigator.

2. Select an experience you want to work in from the UI Builder home page. If you do not see any experiences listed in which to work, contact your administrator to get access to an experience, or create an experience. For more information, see Create an experience.
3. Create a page.
To do this task | Do the following

| a. Click **Menu** and select **Create Page**. |
| - UI Builder | - Menu | - Help | - URL (now/demo) |
| - Create page |
| - Create variant |
| - Duplicate variant |
| - Configure |
| - Edit page settings |
| - Edit variant settings |
| - Edit experience settings |
| - Developer |

| b. Type the name of your new page in **Create a page > Name of page**. The name can be anything you want. In this example, you can type **My First Page**. |

| c. In the **Path** field, a path is automatically entered based on your page name. In this example, the Path is **my-first-page**. The path is the URL of the page. You can change this path to anything you want, but the path must be unique. The path has to be lowercase and contain no spaces. The **URL preview** shows what the path of your page will be. |

| d. Skip the page template option as you are creating a blank page to start. |

| e. The application scope defaults to the scope that a user is currently in within the Now Platform®. For more information, see **Security and roles in UI Builder**. |

| f. Select **Create**. |

| g. After your page is created, you can change advanced settings such as the **URL settings** and **Variant** settings. For this quick start, you can skip these advanced options and select **Done**. For more information on changing advanced settings, see **Create a page: Advanced settings**. |
Congratulations! You have created your first page! The page is empty of content. You add components to the page to build functionality to it. Components are the building blocks of a page. UI Builder comes with many components ready for you to add to your page. Components can be as simple as a **Heading**, or as complex as a **List**.

4. Change the layout of the page to a two-column layout so your two containers show side by side on the page. For more information about layouts, see Work with layouts.

<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change the layout of the page to two</td>
<td><strong>a.</strong> Select the <strong>Body</strong> level of the <strong>Content</strong> tree to the left of the main</td>
</tr>
<tr>
<td>columns</td>
<td>page area of UI Builder.</td>
</tr>
<tr>
<td></td>
<td><strong>b.</strong> From <strong>Page configuration</strong> to the right of the main page area, select</td>
</tr>
<tr>
<td></td>
<td><strong>Layout</strong> to see the layout options for the page.</td>
</tr>
<tr>
<td></td>
<td><strong>c.</strong> Select the two-column layout option.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>5.</strong> Add a container component to the left column of your page. A container is what holds your components. Think of a container as an area of the page where you add information, images, or functionality (your components). You can have as many containers on a page as you want, with as many containers within containers, with as many components in the containers.</td>
<td>You can add components to the page in different ways. For more information on adding components to your page, see Ways to add a component to a page.</td>
</tr>
</tbody>
</table>
To do this task

<table>
<thead>
<tr>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. Select the + button in the middle of the left column on your page. Type <code>container</code> into the search and select the <code>Container</code> component.</td>
</tr>
<tr>
<td>b. Name your left container to make it easier to identify in the content tree as you work.</td>
</tr>
<tr>
<td>i. Select <code>Container 1</code> in the <code>Content</code> tree to highlight the container.</td>
</tr>
<tr>
<td>ii. Select the pencil icon in the right configuration panel and in the <code>Component label</code> field, type <code>Container for heading</code>.</td>
</tr>
<tr>
<td>iii. Change the <code>Component ID</code> to <code>container_for_heading</code> and select <code>Apply</code>. For more information on component IDs, see <code>Component ID</code>.</td>
</tr>
</tbody>
</table>
To do this task

<table>
<thead>
<tr>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notice the container name changes to <strong>Container for Heading</strong> in the content tree. The content tree is an important concept. The content tree is an easy way to see the structured layout of your page. The content tree is especially important when you have multiple components on your page. You select the component in your content tree to highlight the component on the page, making it easier to build your page. You can do a text search for a component.</td>
</tr>
</tbody>
</table>

You have successfully added your first container component to your page.

6. Add a **Heading** component to your left column container.
To do this task | Do the following
--- | ---
You can add components to the page in different ways. For more information on the ways you can add components to your page, see [Ways to add a component to a page](#).

**a.** Select the + button in the middle of your container and type `Heading` in the search.

![](image)

**b.** Select the **Heading** component to add it to your container. A heading component is a way to add text or a title to your page.

c. Select your new **Heading** component to highlight it.

d. Select the pencil icon to see the **Component label** and **Component ID**. You can change these to anything you want, as long as they are unique.

e. Configure your **Heading** component. Component authors configure components to expose the appropriate properties required to set up their components.

   **i.** In the **Label** field, enter the text of your heading, such as *My new heading*.

   **ii.** Leave the Level as 1, and Style as **Header-primary**.
### To do this task

<table>
<thead>
<tr>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add a container component to your right-column container</strong></td>
</tr>
</tbody>
</table>

#### iii. The Style changes the size of the heading text. For example, if you select **Header-secondary**, the text is smaller. Different headings sizes are useful if you have two headings and want the second heading smaller than the primary heading. For more information on configuring components, see **Configure components**.

For more information about configuring components, see, **Now Experience Components**.

f. Select **Save**. Save your page often as you work.

You have added a heading component to your page.

7. Add a second container in the right column of your page.

You can add components to the page in different ways. For more information on the ways you can add components to your page, see **Ways to add a component to a page**.
To do this task | Do the following
--- | ---
a. Similar to before, select the + button in the right-column container. Type `Container` in the search and select `Container` to add a new container to your container.
b. Rename the container.

i. Select the `Container 1` component in the content tree to highlight your new container on the page.

ii. Select the pencil icon in the right configuration panel and name your `Component label` to `My second container`.

iii. Change the `Component ID` to `my_second_container`. For more information on component IDs, see `Component ID`.

iv. Select `Apply`.

c. Look in the content tree to see your changes. Notice the container name changes to `My second container`. You can easily find your second container by name. Or you can do a text search for the component.
<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>d.</strong> Select <strong>Save</strong>. It is always a good idea to save your page often as you work.</td>
<td></td>
</tr>
</tbody>
</table>

**8.** Add an image component to your page. An image is a nice way to add a visual for your page. In this quick start, you add a stock photo that comes with UI Builder. But you can add any image that is available to you.

<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>8.</strong> Add an image component to your right-column container</td>
<td>You can add components to the page in different ways. For more information on the ways you can add components to your page, see <em>Ways to add a component to a page.</em></td>
</tr>
<tr>
<td></td>
<td><strong>a.</strong> Add the image component to the second container. Select the + in your container.</td>
</tr>
<tr>
<td></td>
<td><strong>b.</strong> In the <strong>Search</strong> field, start typing <strong>Image</strong> until you see the <strong>Image</strong> component, then select the component. A default image is loaded. You can add your own image by adding a URL to the image. You can use images hosted on the Internet or images in the <strong>Now Platform®</strong>. If you use an image hosted on an external site, you must use <strong>https</strong> protocol for security.</td>
</tr>
<tr>
<td></td>
<td><strong>c.</strong> Name your <strong>Image</strong> component to make it easier to identify it in the content tree as you work.</td>
</tr>
<tr>
<td></td>
<td><strong>i.</strong> Select the pencil icon in the right configuration panel to change the component details.</td>
</tr>
<tr>
<td></td>
<td><strong>ii.</strong> In the <strong>Component label</strong> field, type <strong>My image</strong>.</td>
</tr>
</tbody>
</table>
To do this task | Do the following
---|---
**iii.** For Component ID, type `my_image`. For more information on component IDs, see **Component ID**.
**iv.** Select **Apply**.
**v.** Save your page.

For more information about configuring components, see, **Now Experience Components**.

You added an image component to your page.

**9.** Save your new page one last time.

<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
</table>
| **Save your page** | **a.** In the far right corner of UI Builder, select **Save**.  
**b.** Now that you saved your page, you can access it in your experience at any time. |

**10.** Generate a preview of your page.
<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. In the right corner of the URL field, select <strong>Open</strong>.</td>
<td><strong>b.</strong> Your page generates a preview of how it looks in your Workspace or portal experience. The page preview opens in another tab of your browser.</td>
</tr>
</tbody>
</table>

Open your page to preview it

![Image of UI Builder interface with a preview of a page with a heading and an image.]()

Congratulations! You completed the UI Builder quick start.

**Results**

1. Created your first page in UI Builder.
2. Changed the layout of your page to have two columns.
3. Added your first container component to the left column. You changed the label for the container.
4. Added a **Heading** component as a title to your left component. You changed the text of the heading. You also changed the label.
5. Added another container component to your right column. You changed the label for the container.
6. Added an **Image** component. You changed the label of the image component.
7. Saved your new page.
8. Previewed your page in the browser.

You successfully completed the UI Builder quick start!

**UI Builder tutorial**

Learn how to use the basics of Now® Experience UI Builder to create a page called My Tutorial.

**Before you begin**

Role required: ui_builder_admin

In this UI Builder tutorial, you perform the following tasks to build a page in UI Builder.

- Start UI Builder.
- Create a page for your workspace or custom portal experience. For more information about creating pages, see Create a page.
- Change the layout of the page to have two columns. For more information, see Work with layouts.
- Build your page by adding two container components.
- Rename your container components in the content tree.
- Add a **Heading** component and a **Button** component to the first container. Add a **data visualization component** to the second container. For more information about components, see Work with components.
- Configure your components as follows:
  - Link the button to the ServiceNow® website.
  - Connect the **Data visualization** component to a data source to display task data. For more information about data resources, see Work with data resources.
- Save your page often.
- Preview your page to see how it looks in a browser.
Procedure

1. Navigate to **Now Experience Framework > UI Builder**. You can also type **UI Builder** directly in the **Filter navigator**.

2. From the UI Builder home page, choose an experience you want to work in. If you do not see any experiences listed in which to work, contact your administrator to get access to an experience or create an experience. For more information, see Create an experience.

3. Create a page.

<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>a.</strong> Click <strong>Menu</strong> and select <strong>+ Create Page</strong>.</td>
<td></td>
</tr>
<tr>
<td><strong>b.</strong> Enter the name of your new page. The name can be anything you want. In this example, you can type <strong>My tutorial page</strong>.</td>
<td></td>
</tr>
<tr>
<td><strong>c.</strong> In the <strong>Path</strong> field, a path is automatically entered based on your page name. In this example, the <strong>Path</strong> is <strong>my-tutorial-page</strong>. The path is the URL of the page. You can change this path</td>
<td></td>
</tr>
<tr>
<td>To do this task</td>
<td>Do the following</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------</td>
</tr>
</tbody>
</table>
| to anything you want, but the path must be unique. The path has to be lowercase and contain no spaces. The **URL preview** shows what the path of your page will be. | Create a page  
First, let's set up this page.  

<table>
<thead>
<tr>
<th>Name of page</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>My tutorial page</td>
<td>my-tutorial-page</td>
</tr>
</tbody>
</table>

URL preview  
/now/demo/my-tutorial-page  
d. Skip the page template option as you are creating a blank page to start.  
e. The application scope defaults to the scope that a user is currently in within the Now Platform®. For more information, see Security and roles in UI Builder.  
f. Select **Create**.  
g. After your page is created, you can change advanced settings such as the **URL settings** and **Variant** settings. For this quick start, you can skip these advanced options and select **Done**. For more information on changing advanced settings, see Create a page: Advanced settings.
Congratulations! You created your tutorial page! The page is empty of content. You add components to the page to build functionality to it. Components are the building blocks of a page. UI Builder comes with many components ready for you to add to your page. Components can be as simple as a **Heading**, or as complex as a **List**.

4. Change the layout of the page to a two-column layout so your two containers show side by side on the page. For more information about layouts, see **Work with layouts**.

<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Change the layout of the page to two columns</strong></td>
<td>a. Select the <strong>Body</strong> level of the <strong>Content</strong> tree to highlight the page.</td>
</tr>
<tr>
<td></td>
<td>b. In <strong>Page configuration</strong> to the right of the main page area, select <strong>Layout</strong> to see the layout options for the page.</td>
</tr>
<tr>
<td>To do this task</td>
<td>Do the following</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------</td>
</tr>
<tr>
<td>c. Select the two-column layout option.</td>
<td></td>
</tr>
<tr>
<td><strong>Change layout of page</strong></td>
<td></td>
</tr>
</tbody>
</table>

5. Add a container component to the left column of your page. A container is what holds your components. Think of a container as an area of the page where you add information, images, or functionality (your components). You can have as many containers on a page as you want, with as many containers within containers, with as many components in the containers.

<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. Select the + Add component button in the middle of the left column on your page. Type container into the search and select the Container component.</td>
<td>You can add components to the page in different ways. For more information on the ways you can add components to your page, see Ways to add a component to a page.</td>
</tr>
</tbody>
</table>
b. Name your left container to make it easier to identify in the content tree as you work.
   i. Select Container 1 in the Content tree to highlight the container.
   ii. Select the pencil icon in the right configuration panel and in the Component label field, type Left container.
   iii. Change the Component ID to left_container and select Apply. For more information on component IDs, see Component ID.
<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
</table>

Notice the container name changes to **Left container** in the content tree. The content tree is an easy way to see the structured layout of your page. The content tree is especially important when you have multiple components on your page. You select the component in your content tree to highlight the component on the page, making it easier to build your page. You can also do a text search for a component.

You have successfully added your first container component to your page.

6. Add a **Heading** component, and a **Button** component to your left column container.
To do this task | Do the following
---|---
You can add components to the page in different ways. For more information on the ways you can add components to your page, see Ways to add a component to a page.

a. Select the + button in the middle of your container and type `Heading` in the search.

b. Select the `Heading` component to add it to your container. A heading component is a way to add text or a title to your page.

c. Select your new `Heading` component to highlight it.

d. Select the pencil icon to see the `Component label` and `Component ID`. You can change these to anything you want, as long as they are unique.

e. Configure your `Heading` component. Component authors configure components to expose the appropriate properties required to set up their components.

i. In the `Label` field, enter `My Tutorial` as the heading text.

ii. Leave the Level as 1, and Style as `Header-primary`. The Style changes the class of the heading text, like font, weight, and color. For
<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
</table>

- **Example:** If you select **Header-secondary**, the text is smaller. Different headings sizes are useful if you have two headings and want the second heading smaller than the primary heading.

Add and configure a button component to your left-column container

Add a **Button** component to your page. Configure the button by changing the text and the way it looks. Then change what happens when you click the button by adding an event handler for the button. The event handler adds an action to the button to link to the ServiceNow® home page. So when you click the button, the ServiceNow® home page opens in a new tab of your browser. For more information about configuring components, see [Now Experience Components](#).

1. Select the `+` on the bottom edge of your container to add a component beneath the container.
2. In the **Search** field, start typing **Button** until you see the **Button** component, then select the component to add it to your container. For more information on adding components to
<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td>your page, see Ways to add a component to a page.</td>
<td></td>
</tr>
<tr>
<td>c. Select the pencil icon to see the Component label and Component ID. You can change these to anything you want, as long as they are unique.</td>
<td></td>
</tr>
<tr>
<td>d. Configure the button component as follows.</td>
<td></td>
</tr>
<tr>
<td>i. Change the Size to Large.</td>
<td></td>
</tr>
<tr>
<td>ii. In the Label field, type Home Page to change the name of your button.</td>
<td></td>
</tr>
<tr>
<td>iii. In the Icon drop down, select Home Outline. You can search for Home Outline instead of scrolling through the list.</td>
<td></td>
</tr>
<tr>
<td>e. Add an event handler to configure what action applies to the button.</td>
<td></td>
</tr>
<tr>
<td>i. Select the Events tab.</td>
<td></td>
</tr>
<tr>
<td>ii. Under Button clicked, select + Add a new event handler.</td>
<td></td>
</tr>
</tbody>
</table>
To do this task | Do the following
---|---

| iii. Under Inherited event handlers, select **Link to destination**. |
| iv. Select **Select destination**. |
| v. In the **Configure navigation** screen, select **External URL**. |
| vi. Type your URL. For this tutorial, type `https://www.servicenow.com` **Select OK**. |
| vii. Select **Add**. |

For more information about event handlers, see **Work with events**.

f. Select **Save**. It is always a good idea to save your page often as you work.

You added and configured the **Heading** and **Button** component for your page.
7. Add a second container in the right column of your page.

<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a container component to your right-column</td>
<td>You can add components to the page in different ways. For more information on the ways you can add components to your page, see Ways to add a component to a page.</td>
</tr>
<tr>
<td>a. Similar to before, select the + button in the right-column container. Type Container in the search and select Container to add a new container.</td>
<td></td>
</tr>
<tr>
<td>b. Rename the container.</td>
<td>i. Select the Container 1 component in the content tree to highlight your new container on the page.</td>
</tr>
<tr>
<td></td>
<td>ii. Select the pencil icon in the right configuration panel and name your Component label to My visualizations container.</td>
</tr>
<tr>
<td></td>
<td>iii. Change the Component ID to my_visualizations_container. For more information on component IDs, see Component ID.</td>
</tr>
<tr>
<td></td>
<td>iv. Select Apply.</td>
</tr>
</tbody>
</table>
To do this task | Do the following
--- | ---
c. Look in the content tree to see your changes. Notice the container name changes to **My visualizations container** in the content tree. You can easily find your second container by name. You can also do a text search for a component.
8. Add a **Data visualization** component to the container. A data visualization component contains data that you display in a visual manner. Configure the data visualization component parameters. Then, add a data resource to it.

<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>You can add components to the page in different ways. For more information on the ways you can add components to your page, see Ways to add a component to a page.</td>
</tr>
<tr>
<td>a.</td>
<td>Select the + to add a component to your second container.</td>
</tr>
<tr>
<td>b.</td>
<td>In the <strong>Search</strong> field, start typing Data visualization until you see the Data visualization component, then select the component to add it to your page.</td>
</tr>
<tr>
<td>c.</td>
<td>Select the pencil icon to see the <strong>Component label</strong> and <strong>Component ID</strong>. You can change these to anything you want, as long as they are unique.</td>
</tr>
<tr>
<td>d.</td>
<td>Add a data resource to your data visualization component to bring in customer account data.</td>
</tr>
</tbody>
</table>

Add and configure a data visualization component to your right-column container

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td>i. Select the data visualization component, then select the <strong>Config</strong> tab.</td>
<td></td>
</tr>
<tr>
<td>ii. Select <strong>Add data source</strong>.</td>
<td></td>
</tr>
<tr>
<td>iii. Type <code>Task [task]</code> in the search field and look for a task that is available on your UI Builder instance.</td>
<td></td>
</tr>
<tr>
<td>iv. Select <strong>Add this source</strong>.</td>
<td></td>
</tr>
<tr>
<td>v. Select <strong>Visualization type</strong>. Then select some different visualization types, such as <strong>Vertical bar</strong>, <strong>Stepline</strong>, and <strong>Line</strong>.</td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>vi.</strong> You can also add a title for the visualization.</td>
<td></td>
</tr>
<tr>
<td><strong>vii.</strong> Save your page.</td>
<td></td>
</tr>
</tbody>
</table>

You have added a **Data visualization** component to the container and configured it. You also added a data resource to bring in customer account data.

9. Save the new page one last time.

<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
</table>
| **Save your page** | a. In the far right corner of UI Builder, select **Save**.  
| | b. Now that you saved your page, you can access it in your experience at any time. |

10. Now preview your page to see what it looks like in a browser.

<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
</table>
| **Preview your page** | a. In the right corner of the URL field in UI Builder, select **Open**.  
| | b. Your page generates a preview of how it looks in your Workspace or portal experience. The |
To do this task | Do the following
--- | ---
|  | page preview opens in another tab of your browser.
  
  **c.** See the heading in the left column of your page.
  
  **d.** Select the **Home Page** button. The ServiceNow® home page opens in another browser tab. Close the browser tab to go back to your preview page.
  
  **e.** See the customer account data visualization in the right column of your page.

---

Congratulations! You completed the UI Builder tutorial.

**Results**

1. Created a page in UI Builder.
2. Changed the layout of your page to have two columns.
3. Added the first container component to the left column.
4. Changed the label for the container.
5. Added a **Heading** component as a title to your left component.
6. Added a button component below your heading component.
   • Configured your button.
   • Added an event handler for your button.
   • Added an event handler to set up a link to a web page. When you click the button, it opens the web page.

7. Added another container component to your right column.

8. Changed the label for the container.

9. Added a Data visualization component.
   • Configured the data visualization component.
   • Added a data resource to bring in task data.

10. Saved your new page.

11. Previewed your page in the browser.

**Understanding your audiences**

Learn how to apply the correct audiences to your Now® Experience UI Builder pages.

**Audiences**

Understanding your audiences in UI Builder is an important part of creating pages. The audience defines who can see your pages. You create pages for targeted experiences tailored to audiences and roles such as agents and managers. For example, you may want create a page for agents to solve issues for your employees. You want to ensure that only people who have the agent role can see the page.

Audiences are generally made of allow/deny lists that include role and domain. But, if the UX Framework User Criteria plugin is enabled, audiences can also target the following:

• Role
• Group
• User
• Company
• Department
You can set audiences to fit a specific role based on one or more criteria. For example, you could create an audience for an ITSM user in Europe who is not a manager.

Set an audience in UI Builder when you create your page or page variant. You can also set the priority of the audience record. The lower the number the higher the priority. Ensure the Active check box is selected to make this page active.

If you do not see an audience that you need, you can click Open audiences in platform to define an audience in the Now Platform®.

- Choosing an audience from the list in the Audience field.

- Selecting Open audiences in platform to edit or create an audience record in the Now Platform®.
• If you have multiple audiences, define the **Order** for each audience record. The order defines the importance of each audience record. For example, a CSMManager could have a higher priority than a CSM Consumer Agent.

• Ensuring the **Active** check box is selected to make this page active.

**Security and roles in UI Builder**

Set up the security and roles for your UI Builder instance.

Security and roles in Now® Experience UI Builder are controlled through your applications scope, domain separation, and protection policy settings.

**Roles**

Roles control access to features and capabilities in UI Builder. The admin role provides access to all features and capabilities. After access has been granted to a role, all the groups or users assigned to the role are granted the access. Roles can contain other roles, and any access granted to a role is granted to any role that contains it.

**Application scope**

Application scoping protects applications by identifying and restricting access to application files and data. Administrators can specify what parts of an application are accessible to other applications, which helps to protect data and application files. In UI Builder, System administrators and developers set application scope when creating a page. Note, you cannot change the application scope after creating the page, so choose your scope carefully.

Changing the scope in UI Builder also changes the scope in the Now Platform®, and vice-versa.

When creating a page, it is important for admins and developers to be aware of the scope you are in for the workspace or portal experience. Choose the correct application scope for your experience. The scope picker is to the right of the URL field. The scope defaults to the scope that the user is currently in within the Now Platform®.
If you change to a different scope while in a page, you will be notified that you are in a different application scope from the one the page was created in.

**Delegated developers for UI Builder**

If you have the application-specific admin role or the system-level admin role, you can delegate application development in ServiceNow® Studio to designated developers at the UI Builder application level.

For more information on how to designate UI Builder application development to a delegated developer, see Delegate development and deployment permissions to personnel.

For more information on delegated developers, see Delegated development and deployment.

For more information about adding an experience for UI Builder from ServiceNow® Studio, see Add an experience to UI Builder from Studio.

**Protection policy**

A protection policy prevents anyone from modifying and/or copying an application file or its related record. A protection policy is typically used when the author of an application is different than the company that will use the application. UI Builder will notify you if you try to modify a protected record.

**UI Builder and Customer Service Management Workspace**

Use Now® Experience UI Builder to create pages for your Customer Service Management Workspace experience.

A workspace is a collection of tasks and workflows in a single focused working area that enables a user to efficiently complete an entire job. It includes tools that a user can employ to quickly and easily assist customers and resolve questions and issues. A workspace also includes features that enable a user to be more efficient, including a multi-tab interface for managing multiple cases and a contextual display that provides quick orientation to the current task.

You can use UI Builder to create pages for your CSM Workspace experience. UI Builder provides CSM-related base system components that you can use to build workspace pages.

You do not create a CSM Workspace in UI Builder. You only use UI Builder to build pages for CSM Workspace experiences.

To create a page in UI Builder for a Workspace experience, you navigate to Now experience framework > UI Builder to open UI Builder.
The UI Builder home displays your available experiences under **My experiences**. If you are working in a Workspace experience, you should see the experience listed here.

For more information, see [Workspaces in Customer Service Management](#).

For information on how to change your Workspace experience settings from within UI Builder, see [UI Builder Workspace experience settings](#).

**UI Builder and portals**

Use Now® Experience UI Builder to create pages for your stand-alone, custom portal experiences.

You can use UI Builder to create and edit pages in your stand-alone, custom portal experiences. Creating pages for your custom portal experience is the same as creating pages for workspaces.
Note: UI Builder is not yet capable of building or configuring out-of-the-box Service portals, like Employee Center. Continue to use the Service Portal Designer instead.

To create a page in UI Builder for a portal experience, you navigate to Now experience framework > UI Builder to open UI Builder.

The UI Builder home displays your available experiences under My experiences. If you are working in a portal experience, you should see the portal experience listed here.

Once you select a portal experience, you can start editing pages, modifying pages, or creating new pages for that portal experience within UI Builder.

For information on how to change your portal experience settings from within UI Builder, see Portal experience settings.

The new framework for UI Builder
Learn about the framework that Now® Experience UI Builder is built on.
UI Builder is built on web standards, and provides a declarative way to build and configure pages for Workspace or Custom Portal. The framework is built on design patterns through reusable components, which allows customers a WYSIWYG experience for extending their experience.

The new framework that UI Builder sits on is powerful, extensible, and lets you build amazing pages for your experience.

**Preview a page in UI Builder**

Learn how to preview a page in Now® Experience UI Builder.

When you create or edit a page in UI Builder, you can open it as a preview to see what the page will look like as a web page. This is a great way to see how your page looks as you build it.

For example, if you add a Delete record button that opens a modal asking a user to confirm the deletion, you may want to open the page in runtime to ensure that the button and modal are displayed the way you want them to. To preview a page, first save your page, then select the Open button to the right of the URL field.

Your page will open as a web page. You can test your page as if it was live on a site. Click the components that you added to ensure they work as expected, check that the data resources are available, the layouts are set up correctly, and so on.

Previewing a page in UI Builder is a powerful tool to help you test your pages.

**Domain separation and UI Builder**

Domain separation is supported for UI Builder. Domain separation enables you to separate data, processes, and administrative tasks into logical groupings called domains. You can control several aspects of this separation, including which users can see and access data.

Role required: ui_builder_admin

**Support level: Standard**

- Includes **Basic** level support.

- Business logic: The service provider (SP) creates or modifies processes per customer. The use cases reflect proper use of the application by multiple SP customers in a single instance.

- The instance owner must configure the minimum viable product (MVP) business logic and data parameters per tenant as expected for the specific application.
Sample use case: An admin must be able to make comments required when a record closes for one tenant, but not for another.

For more information on support levels, see Application support for domain separation.

Overview

Now® Experience UI Builder is a WYSIWYG web user interface builder. UI Builder enables developers to build new pages or customize existing pages for web-based workspace and portal experiences using Now® Experience Components and custom web components. In addition, UI Builder supports Domain Separation, which is the ServiceNow® instance-wide multi-tenant architecture.

For more information on domain separation, see Domain separation explained.

Enable developers or dashboard builders in domain separated environments to safely create UI application screens or dashboards while in the same browser window. Domain Separation in UI Builder works similarly to application scope to help administrators safely create or edit in a multi-tenant environment.

It is important to understand a key principle to maintaining a stable, healthy, and scalable ServiceNow® instance, where Domain Separation is installed. The primary principle is standardization. Standardization means a common configuration that the majority of the instance operates by. When an instance has hundreds or thousands of domains, managing them successfully requires rigorous governance. Domain-specific configurations should be done only if they are deemed necessary by the instance owners. Generally, most instances should follow the common instance configuration. Doing so provides a more uniform experience across the instance. It also lets instance owners minimize code sprawl that slows the adoption of new ServiceNow® features included as part of release upgrades.

How domain separation works in UI Builder

Domain separation in UI Builder works similarly to application scope to help administrators safely create or edit in a multi-tenant environment.

UI Builder is comprised of a framework of interlocked components that you use to create web-based workspaces, dashboards, or portals. While the application supports domain separation, it does not mean every component or table is domain separated, which is important for instance owners to understand.

If the current domain does not match the domain of the variant or dashboard, the record is read-only. If a user has access to the domain, they can choose to switch their domain to the domain of the record. Alternatively, users can edit the
record. Editing the record temporarily forces the user session into that record’s domain. They can then make edits without fear of accidentally creating an override.

The following diagram shows what is (in green) and is not (in blue) domain separated in UI Builder.

Not shown in the diagram are viewports, declarative actions, and screen applicabilities, which are domain supported as process.

Data and Process/UI Separation are important when considering domain separation architecture. UI Builder fully supports data and process/UI separation, and any data (records) displayed in the web-based workspace, dashboard or portal experiences.

For example, a change request that belongs to the domain of Acme only shows for users who have access to the domain of Acme in an experience built using UI Builder. Conversely, if an application does not support data separation, its records won’t be domain separated by the workspace or portal experience.

Process/UI separation tables that form the underpinning framework in UI Builder are process separated, and a sys_override column exists on those tables. For example, if a page is created in Global, any changes to the logic created and saved in a sub domain results in an override.

For items that are not domain separated, any change to the logic globally affects any page or dashboard that references its content. Understanding domain separation is critical when interacting with these elements.
Domain Selection menu, messaging, and managing overrides

When designing a workspace, dashboard, or portal experience using the UI Builder (including Dashboard Builder), a system administrator or ui_builder_admin has access to a Domain Selection menu in UI Builder. A system administrator or ui_builder_admin should switch to the proper domain prior to creating, editing, or overriding a variant or dashboard page.

By default, the ui_builder_admin role does not have access to the Domain Selection menu; it must be coupled with a role that grants access, such as ITIL, or it can be added via system property. For more information, see Restrict access to the domain picker.

In addition, the Domain Selection menu also shows Expand/Collapse Domain Scope, that is displayed while the system administrator or ui_builder_admin is in Global. Select Expand to show any variant or dashboard that has been overridden, or exists as a standalone in a sub domain. Select Collapse to only show variants or dashboards created in Global.

Lastly, domain hierarchy is available from the Domain Selection menu. For deep-domain hierarchies, the user may have to collapse the branches of the domain hierarchy to physically select the domain. In these environments, perform a search to find the domain.

UI Builder has governance controls for editing and overriding variants or dashboards, similar to the way application scope is handled. Both application scope and domain scope are handled concurrently in UI Builder.

For example, if a variant was created in Global, but the current domain of the system administrator is set to Acme, then that variant is read-only. As long as that screen is not in a private scope that prevents editing, the system administrator or ui_builder_admin have two options. They can temporarily transact into the global domain if they have access to Global. Or, they can create an override.

You can edit the domain separation to make quick changes to the variant or dashboard and its content. When you edit the domain, you temporarily transact into the same domain scope as the variant or dashboard. Going into the same scope prevents accidental overrides when modifying certain settings such as Name, Order, Event Mappings, Page Definition configurations) tied to the variant. While in edit mode, not all settings are available in page management. For full capabilities, switch into the correct domain prior to editing the record.

Create Override allows a system administrator or ui_builder_admin to create an override of an existing variant or dashboard. Create an override of a variant or dashboard to perform an extensive copy of the page definition content, minus screen conditions and audiences in the user’s currently selected domain. The sys_override column is then updated appropriately.
Viewports which are variants in of themselves, are domain separated, and are typically nested inside page definition content. Some viewports may not copy over. For example, a viewport (displayed as a tab set) that was created as an override in a domain of a Global viewport would not be carried in the page definition content during the override creation process.

As screen conditions and audiences may be specific to a domain, this content is not carried over during the override creation process. A screen prompts the system administrator or ui_builder_admin to create screen conditions and audiences.

A user cannot create an override of a variant or dashboard in Global if the item exists in a sub domain, or if an override exists for that variant or dashboard in the same sub domain.

Once the override and the conditions and audiences are set, the content and configurations can be configured as needed. As standard to domain separation, the override is no longer affected by any changes done to the original variant or dashboard. The workspace, dashboard, or portal experience displays these overridden configurations if the user’s current domain session is within the affected domain or sub domains where this override was created. Audiences further determine what a user may or may not see.

In addition, a user can access the domain hierarchy to view existing overrides from higher domains. For example, Global <- Top <- Acme <- Current domain. If no overrides exist, the default variant or dashboard are displayed. The exception is if the default variant or dashboard is in a child domain or a peer domain.

If you select **Expand Domain Scope** while in Global, all variants and overrides in sub domains are shown as previously mentioned.
System administrators and ui_builder_admin can see what has been created in the ServiceNow® platform.

**Viewports and domain separation**

Viewports are variants that can be nested in page definition content. They can be created as a common configuration in Global, or can be overridden per sub domain.

**Declarative actions and domain separation**

Declarative Actions can be overridden per domain as well. A system administrator or ui_builder_admin should select the appropriate domain prior to creating a domain specific declarative action override.

**Themes and domain separation**

You can apply themes to a workspace or portal experience in UI Builder Experience Settings based on domain. However, prior to entering Experience Settings as a system administrator or ui_builder_admin, ensure that the appropriate domain is selected first.

Once the correct domain is selected, go to Experience Settings and select the theme that is to be applied. All sub domains have this theme applied unless specifically overridden.

**Related information**

- Domain separation for service providers

**Create an experience for UI Builder**

Learn how to create a new workspace or portal experience for Now® Experience UI Builder in the ServiceNow platform.

**Before you begin**

Role required: ui_builder_admin

**About this task**

This task shows you a basic example of how to create a new workspace or portal experience for UI Builder. This example creates an experience called My App to demonstrate the process.
Procedure

1. Navigate to **Now Experience Framework > UI Builder**.

2. From the UI Builder home page, select **+ Create experience in Platform**. You leave UI Builder and go to the ServiceNow platform to create the experience.

3. Select **New** to create a new UX Application record for your new experience.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Add a title for your new experience. For this example, it is <strong>MyApp</strong>.</td>
</tr>
<tr>
<td>Page</td>
<td>Search for the page you want to create the experience for. In this example, select <strong>UI Builder</strong>.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>App Shell UI</td>
<td>You must choose an app shell in order for the experience to work. For example, you could choose a workspace or portal app shell. The app shell is the wrapper of the page contents, which is similar to the functionality of a web page. The app shell can show things like the logo of your company, user preferences, the search icon. For more information, see App Shells for UI Builder.</td>
</tr>
<tr>
<td>URL path</td>
<td>Add a path for your experience. The path appends to the end of the URL parameter of your experience. For example, if you used MyApp as the title for your experience, you could type my/app for the path.</td>
</tr>
<tr>
<td>Admin panel</td>
<td>a. Choose a table to associate to your experience from the list of tables, such as UX App Configuration.</td>
</tr>
<tr>
<td></td>
<td>b. Search for a document, or create a new document. If you create a new document, click the magnifying glass icon, then select New. In the simple MyApp example, you would type My App in the Name field, then select Submit.</td>
</tr>
<tr>
<td></td>
<td>c. Select OK in the Select the document field.</td>
</tr>
</tbody>
</table>
4. Select **Submit** to submit your new experience record.

5. Go back to your UI Builder home page. Refresh the page. You see your new experience listed under *My Experiences*. In this simple example, you see My App listed.

**App Shells for UI Builder**

Understand what app shells are, what app shells are available for UI Builder, and why you would pick one over another.

**App shells**

An app shell is the wrapper of the contents of an experience page. An app shell contains functionality similar to a web page. For example, an app shell can show things in a header or footer of your experience, such as a logo for your company, user preferences, a search icon, configuration icon, user menu, and so on.

An app shell is required for UI Builder.

The app shell defines whether your experience has a workspace or portal design. A workspace is a graphical user interface that puts multiple tools on one page for handling requests from users. A portal is a page where users can add requests, such as order items, track their tickets, and so on.

You choose which app shell to apply to your experience when you create a new experience in the ServiceNow® platform. For more information on how to create a new experience, see *Create an experience for UI Builder*.

You can choose from a number of app shells when creating a new experience.
Agent Workspace and portal app shell descriptions

<table>
<thead>
<tr>
<th>App shell</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent Workspace app shell</td>
<td>The Agent Workspace app shell contains the headers and footers for an Agent Workspace. An Agent Workspace includes the tools that agents need to find, research, and resolve issues. The ServiceNow platform provides different Agent Workspaces for different issue types. For example, agents using ITSM Manager Workspace (ITSM) track and resolve IT issues. Agents using Customer Service Management Workspace (CSM) resolve external customer cases, such as sending a piece of computer hardware to fulfill a customer’s request.</td>
</tr>
</tbody>
</table>
Agent Workspace and portal app shell descriptions (continued)

<table>
<thead>
<tr>
<th>App shell</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Portal app shell</td>
<td>The portal app shell contains the header and footer for a portal experience. The portal app shell record is a reference implementation of how the menu, utilities, logo, and login are configured for a portal experience.</td>
</tr>
</tbody>
</table>

**Experience settings for UI Builder**

Learn about the User Interface (UI) Builder experience settings to build your own workspace and custom portal experiences.

By changing the settings of the workspace or portal experience that you're working in, you can affect how your users interact with the experience, how the experience looks, and how users navigate to and from the experience.

Before you can edit the experience settings, you need to be in the correct application scope. If you're in a different scope, the experience settings are read-only. To change your application scope, go to the main header, select the application picker (Global), and then select the application scope that you want. For more information about the application scope, see Security and roles.

**Workspace experience settings**

You can change these settings for your workspace experience:

- Modify or add some general settings for your experience. For example, you can change the title, description, and the path of the workspace or portal.

- Apply your organization's branding to all pages in your experience or to a page within your experience.

- Configure your side navigation settings to add pages to the side navigation in your experience.

- Turn on or turn off the notifications for your experience. You can change the global search settings for your experience. For example, you can choose to show or hide the search bar in your experience or change the search source that determines where the search results come from.
Workspace edit experience settings screen

<table>
<thead>
<tr>
<th>General</th>
<th>What are experience settings?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Branding and theming</td>
<td></td>
</tr>
<tr>
<td>Side navigation</td>
<td></td>
</tr>
<tr>
<td>Utilities</td>
<td></td>
</tr>
</tbody>
</table>

These settings allow you to customize your web experience to your needs. You can apply your organization's branding, pick a theme, configure navigation systems, or enable search functionality on any pages you create. Learn more on setting up your experience.

**Portal experience settings**

You can change these settings for your portal experience:

- Modify or add general settings for your experience. For example, you can change the title, the description, and the path of the workspace or portal.
- Apply your organization's branding to all of the pages in your experience or to a page within your experience.
- Set up the navigation and menu settings in the app shell of your portal experience. The app shell, which is the wrapper of the portal contents, is similar to a Google Chrome web page.
- Control whether the global search functionality is visible to the users of your portal experience. For example, you can enable or disable the global search for either your public or private pages.

To learn more about the workspace experience settings, see **UI Builder workspace experience settings**.

To learn more about the portal experience settings, see **UI Builder portal experience settings**.

**UI Builder workspace experience settings**

You can change the User Interface (UI) Builder workspace experience settings to fit your company goals.
By changing the experience settings, you can affect how your users interact with your workspace experience, how your workspace looks, and how users navigate to and around your workspace.

Before you can edit the experience settings, you need to be in the correct application scope. If you’re in a different scope, the experience settings are read-only. To change your application scope, go to the main header, select the application picker ( ), and then select the application scope that you want. For more information about the application scope, see Security and roles.

Learn how you can change these workspace experience settings:

• Modify or add the workspace general settings for your workspace experience. For example, you can change the title, the description, and the path of the workspace.

• Work with the workspace branding and theming settings to apply your organization’s branding to all pages in your workspace experience or to a specific page within your workspace experience.

• Change the workspace side navigation settings to add pages to the side navigation in your workspace experience.

• Modify the workspace utility settings to turn on or turn off notifications for your workspace experience. Also, you can change the global search settings for your workspace experience. For example, you can choose to show or hide the search bar in your workspace experience or change the search source that determines where the search results come from.

Experience settings for workspace

<table>
<thead>
<tr>
<th></th>
<th>What are experience settings?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>These settings allow you to customize your web experience to your needs. You can apply your organization’s branding, pick a theme, configure navigation systems, or enable search functionality on any pages you create. Learn more on setting up your experience.</td>
</tr>
</tbody>
</table>

Change the general settings in your workspace experience

Change the general settings for your workspace experience in User Interface (UI) Builder to fit the needs of your organization. For example, you can modify the title, description, and path for your workspace experience.
Before you begin
You need to be in the correct application scope to edit the experience settings. If you're in a different scope, the experience settings are read-only. To change your application scope, go to the main header, select the application picker ( ), and then select the application scope that you want. For more information about the application scope, see Security and roles.

Role required: ui_builder_admin

Procedure
1. Navigate to Now Experience Framework > UI Builder.
2. Open a workspace experience or create an experience by selecting Create experience in Platform.
   For more information, see Create an experience.
3. Open or create a page in your workspace experience.
4. To change the title, description, and path of your workspace experience, go to the menu and then click Edit experience settings.
   • The title lets everyone know what the workspace is called. Take care when choosing a name for your workspace.
   • The description lets your users know the details of the workspace.
   • The path must be unique. The path can include digits (0-9), letters (A-Z, a-z), and a few special characters ("-", ".", "_", or "~") with the words separated by a forward slash or hyphen.

UI Builder experience settings

General

Edit the name, description and URL of this experience. Learn more

Title *
Base agent workspace

Description

Path *
demo/baseaw

Advanced settings

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
5. To edit the workspace record on the Now Platform, navigate to **General > Advanced settings**.

6. Update the record in the platform when you are finished. When you go back to your workspace experience settings in UI Builder, you see the changes that you made to the general settings.

### Change the brand and theme settings in your workspace experience

Change the brand and theme settings for your workspace experience in UI Builder so that your users can see a consistent look and feel across all pages.

**Before you begin**

You need to be in the correct application scope to edit the experience settings. If you’re in a different scope, the experience settings are read-only. To change your application scope, go to the main header, select the application picker (Global), and then select the application scope that you want. For more information about the application scope, see **Security and roles**.

Role required: ui_builder_admin

**Procedure**

1. Navigate to **Now Experience Framework > UI Builder**.

2. Open a workspace experience to work in or create an experience by selecting **Create experience in Platform**.
   For more information, see **Create an experience**.

3. Open or create a page in your workspace experience.

4. From the list, select **Edit experience settings**.

5. Make changes to your brand and theme.
   For more information about themes in UI Builder, see **Work with themes**.

<table>
<thead>
<tr>
<th>Action</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply a theme</td>
<td>a. Select <strong>Branding and theming</strong>. If themes are available on your instance, select a theme from the list so that you can apply or change the theme for the workspace record on the platform, including</td>
</tr>
</tbody>
</table>
a. To edit the application record, navigate to Branding and theming > Advanced settings. You are now in the Now platform where you can edit the application record.

b. Update the record in the platform when you are finished.

c. To go back to your portal experience, select Open in UI Builder. Refresh UI Builder to see the changes that you made to the branding and theming.

Workspace platform opens in UI Builder.
<table>
<thead>
<tr>
<th>Action</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>a.</td>
<td>To create a theme record in the Now Platform, select <strong>Create a theme</strong>. Then, fill in the new record details as described in the following steps. If a theme that you want is not available, create a theme in the platform and then apply it to your workspace experience from within UI Builder. Refresh UI Builder to see your applied theme.</td>
</tr>
</tbody>
</table>

**New theme record**

Create a theme

b. In the **Name** field, enter a name for the theme
<table>
<thead>
<tr>
<th>Action</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>c.</td>
<td>In the <strong>Name</strong> field, enter a name for the theme.</td>
</tr>
<tr>
<td>d.</td>
<td>Add a description. The description helps to identify what the theme does.</td>
</tr>
<tr>
<td>e.</td>
<td>In the <strong>Extends</strong> field, search for an existing theme that extends to the theme that you are creating.</td>
</tr>
<tr>
<td>f.</td>
<td>Add an override theme, if needed.</td>
</tr>
<tr>
<td>g.</td>
<td>In the <strong>Theme</strong> field, add any of the default CSS styles. The CSS is formatted as JSON. For more information about creating a theme, see <a href="#">Create a custom theme</a>.</td>
</tr>
<tr>
<td>h.</td>
<td>Ensure that the Active check box is selected. If there are multiple themes, the theme that you apply takes precedence.</td>
</tr>
<tr>
<td>i.</td>
<td>Click <strong>Submit</strong>.</td>
</tr>
</tbody>
</table>

**Add pages to the side navigation in your workspace experience**

Add primary and secondary pages to your workspace side navigation in User Interface (UI) Builder so that your users can navigate to different pages within your workspace experience.

**Before you begin**

Role required: ui_builder_admin

**About this task**

You can add up to seven pages to your side navigation.

You need to be in the correct application scope to edit the experience settings. If you are in a different scope, the experience settings are read-only. To change your application scope, go to the main header, select the application picker (Global), and then select the application scope that you want. For more information about the application scope, see [Security and roles](#).
Procedure

1. Navigate to **Now Experience Framework > UI Builder**.

2. Open a workspace experience to work in or create an experience by selecting **Create experience in Platform**. For more information, see **Create an experience**.

3. Open or create a page in your workspace experience.

4. To change the settings for the entire workspace experience that you are working in, go to the menu and click **Edit experience settings**. You can only change the experience settings in the workspace record in the **Now Platform®**.

5. To add pages to the side navigation in your workspace experience, click **Side navigation**.

6. To go to the Now Platform® to modify the navigation settings in the application record, select **Advanced side navigation settings**.

   **Experience settings for the workspace side navigation**

   **Side navigation**

   Choose up to 7 pages to add to your side navigation. Each page can have an icon and reflect a certain order. Learn more

   ![Advanced side navigation settings](image)

   **a.** Search for the page that you want to add to your side navigation. You can add multiple pages to your workspace side navigation, but only the first seven icons show in the side navigation in the order that you select.

   **b.** If you have more than one page, define the order value of the page in the workspace side navigation.

7. Select **Submit**.

8. Go back to your workspace experience in UI Builder and preview your page to see the side navigation.

   You can preview your page in UI Builder by clicking **Open** in the right corner of the URL field.

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Set up notifications in Workspace utility settings

Set up the advanced notification and search settings in your workspace experience so that you can control how your users see notifications and search options.

Before you begin
Role required: ui_builder_admin

About this task
Set up the advanced notification and search settings in your workspace experience. You can turn notifications on or off to control how people see notifications in your workspace experience.

You can also control whether the search functionality is visible to the users of your workspace experience. If you choose to display a search option, you can define what search results are returned by choosing a source for the search.

Workplace experience settings for utility

Notifications

Control whether your end-user sees notifications as they interact with this workspace. Learn more

☐ Turn on notifications

☐ Advanced notification settings

Global search

You can show or hide the search functionality in this workspace. If you show search, you can define what search results appear to your end-user. Learn more

☐ Show global search

Search source

Agent Workspace Search Config

☐ Advance global search settings

Before you can edit the experience settings, you need to be in the correct application scope. If you’re in a different scope, the experience settings are read-only. To change your application scope, go to the main header, select the application picker (Global), and then select the application scope.
that you want. For more information about the application scope, see Security and roles.

Procedure

1. Navigate to **Now Experience Framework > UI Builder**.

2. Open a workspace experience to work in or create an experience by selecting **Create experience in Platform**.
   For more information, see **Create an experience**.

3. Open or create a page in your workspace experience.

4. To change the settings for the entire workspace experience that you are working in, click **Edit experience settings**.
   You can only change the experience settings in the workspace record in the Now Platform®.

<table>
<thead>
<tr>
<th>Action</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Edit notifications settings</strong></td>
<td>a. Click <strong>Utilities</strong>.</td>
</tr>
<tr>
<td></td>
<td>b. To turn notifications on or off for your experience, select <strong>Turn on notifications</strong>.</td>
</tr>
<tr>
<td></td>
<td>c. To go to the Now Platform® and edit the JSON values of the record, click <strong>Advanced notification settings</strong>.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Action</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Edit global search settings</strong></td>
<td>a. Click <strong>Utilities</strong>.</td>
</tr>
<tr>
<td></td>
<td>b. To show the search functionality on your workspace, click <strong>Show global search</strong>. To hide the search functionality on your workspace, deselect <strong>Show global search</strong>.</td>
</tr>
<tr>
<td></td>
<td>c. To go to the Now Platform® to make changes to the global search application record, click <strong>Advanced global search settings</strong>.</td>
</tr>
</tbody>
</table>

**UI Builder portal experience settings**

You can change your UI Builder custom portal experience settings to affect how your users interact with your portal experience.
Use experience settings to change the settings for the custom portal experience that you are working in. These settings affect how your users interact with your portal experience, how your portal looks, and how users navigate to and around your portal.

You need to be in the correct application scope to edit experience settings. If you are in a different scope, the experience settings are read-only. To change your application scope, in the main header, select the application picker, and then select the application scope that you want. For more information about the application scope, see Security and roles.

Learn how to change the following portal experience settings.

• Modify or add portal general settings for your portal experience. Change the title of your portal, the description, and the path of the portal.

• Work with portal branding and theming to apply an existing theme, or create a theme for your portal experience. You use a theme to add your organization's branding to all of the pages in your portal experience or to a specific page within your portal experience.

• Use portal navigation and menu settings to set up the navigation and menu settings in the app shell of your portal experience. The app shell is the wrapper of the portal contents. For example, the app shell can show things like the logo of your company, user preferences, the search icon, the configuration icon, and the user menu. For more information about app shells, see App shells for UI Builder.

• Modify the Portal search settings to show or hide the global search functionality on public or private pages in your portal experience.

---

**Change the general settings in your portal experience**

Change the general settings for your custom portal experience in UI Builder. For example, you can modify the title, description, and path for your portal experience.

**Before you begin**

You need to be in the correct application scope to edit the experience settings. If you're in a different scope, the experience settings are read-only. To change your application scope, go to the main header, select the application picker.
( Local ), and then select the application scope that you want. For more information about the application scope, see Security and roles.

Role required: ui_builder_admin

Procedure

1. Navigate to Now Experience Framework > UI Builder.

2. Open a portal experience to work in or create an experience by selecting Create experience in Platform.
   For more information, see Create an experience.

3. Open or create a page in your portal experience.

4. From the menu, click Edit experience settings to change certain settings for the portal experience you are working in.
   You can fully change all the experience settings in the portal record in the Now Platform®.

5. To change the title, description, and the path of the portal experience, click General.
   • The title lets everyone know what the workspace is called. Take care when choosing a name for your portal.
   • The description lets your users know the details of the portal.
   • The path must be unique. The path can include digits (0-9), letters (A-Z, a-z), and a few special characters ("-", ".", "+", "~"), with the words separated by a forward slash or hyphen.

   a. Add a title for your portal experience.
      The title of the portal is required. The title is an important part of the portal settings, as it tells everyone what the portal is. Take care to choose the proper name and naming convention when naming or changing the name of a portal.

   b. Optional: Add a description for your portal experience.
      The description helps the administrators and developers know the details of the portal.

   c. Set the path for your portal experience.
      The path is set based on the name of the page that you created in your portal experience. You can change this path any time. The path is required and must be unique. The path can include digits (0-9), letters (A-Z, a-z), and a few special characters ("-", ".", "+", "~"), with the words separated by a forward slash or hyphen.
6. To edit the portal record on the Now Platform, navigate to **General > Advanced settings**.

7. **Optional:** To change the title, description, and the path of the portal record on the Now Platform, click **Advanced settings**. You must have the proper role assigned to you before you can make these changes.

8. Update the record in the platform when you are finished.

**Results**
When you go back to your portal experience settings in UI Builder, you see the changes that you made to the general settings.

**Change the brand and theme settings in your portal experience**
Change the brand and theme settings for your custom portal experience in UI Builder so that your users can see a consistent look and feel across all pages.

**Before you begin**
You need to be in the correct application scope to edit the experience settings. If you're in a different scope, the experience settings are read-only. To change your application scope, go to the main header, select the application picker (Global), and then select the application scope that you want. For more information about the application scope, see **Security and roles**.

Role required: ui_builder_admin

**Procedure**

1. Navigate to **Now Experience Framework > UI Builder**.

2. Open a portal experience to work in or create an experience by selecting **Create experience in Platform**. For more information, see **Create an experience**.

3. Open or create a page in your portal experience.

4. To change certain settings for the portal experience that you are working in, from the menu, click **Edit experience settings**. You can fully change all the experience settings in the portal record in the Now Platform®.

5. Make changes to your brand and theme.
   For more information about themes in UI Builder, see **Work with themes**.
<table>
<thead>
<tr>
<th>Action</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply a theme</td>
<td>To apply a theme for your portal experience, select <strong>Branding and theming</strong>.</td>
</tr>
<tr>
<td></td>
<td><strong>Portal branding and theming</strong></td>
</tr>
<tr>
<td></td>
<td>Branding and theming</td>
</tr>
<tr>
<td></td>
<td>Select a theme for this experience. Themes apply branding to an experience to give it a consistent look and feel across all pages.</td>
</tr>
<tr>
<td></td>
<td>Learn more</td>
</tr>
<tr>
<td></td>
<td><em>Themes in:</em></td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="Create a theme" /></td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="Advanced settings" /></td>
</tr>
<tr>
<td></td>
<td>For more information about themes in UI Builder, see <a href="image">Work with themes</a>.</td>
</tr>
<tr>
<td>Edit or change a theme</td>
<td>a. To edit the application record, navigate to <strong>Branding and theming &gt; Advanced settings</strong>. You are now in the Now Platform where you can edit the application record.</td>
</tr>
<tr>
<td></td>
<td>b. Update the record in the platform when you are finished.</td>
</tr>
<tr>
<td></td>
<td>c. To go back to your portal experience select <strong>Open in UI Builder</strong>. Refresh UI Builder to see the changes that you made to the branding and theming.</td>
</tr>
<tr>
<td>Action</td>
<td>Procedure</td>
</tr>
<tr>
<td>--------</td>
<td>-----------</td>
</tr>
<tr>
<td><strong>Open a theme in UI Builder</strong></td>
<td><img src="image" alt="Image of UI Builder interface" /></td>
</tr>
<tr>
<td><strong>Create a theme</strong></td>
<td><img src="image" alt="Image of theme creation interface" /></td>
</tr>
</tbody>
</table>

d. To apply or change the theme for the portal record on the platform, including all pages in the portal experience, click **Advanced settings**.
e. Update the record in the platform when you are finished.
f. To go back to your portal experience in UI Builder, click **Open in UI Builder**.
g. To see the changes that you made to the branding and theme, refresh UI Builder.
a. To create a new theme record in the ServiceNow platform, select **Create a theme**. Fill in the new record details as described in the following steps. If a theme that you want is not available, create a theme in the platform and then apply it to your portal experience from within UI Builder. To see your applied theme, refresh UI Builder.
b. Fill in the new record details.
<table>
<thead>
<tr>
<th>Action</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>c.</td>
<td>In the <strong>Name</strong> field, enter a name for the theme.</td>
</tr>
<tr>
<td></td>
<td>d. Add a description.</td>
</tr>
<tr>
<td>Creating a theme record is useful if a theme that you want is not available. You create the theme in the platform, and then you can apply it to your portal experience from within UI Builder.</td>
<td></td>
</tr>
<tr>
<td>e.</td>
<td>In the <strong>Name</strong> field, enter a name for the theme.</td>
</tr>
<tr>
<td>f.</td>
<td>Add a description.</td>
</tr>
<tr>
<td>Action</td>
<td>Procedure</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>g.</td>
<td>In the <strong>Extends</strong> field, search for an existing theme that extends to the theme that you are creating.</td>
</tr>
<tr>
<td>h.</td>
<td>Add an override theme, if needed.</td>
</tr>
<tr>
<td>i.</td>
<td>In the <strong>Theme</strong> field, add any of the default CSS styles. The CSS is formatted as JSON. For more information about creating a theme, see <a href="#">Create a custom theme</a>.</td>
</tr>
<tr>
<td>j.</td>
<td>Ensure that the <strong>Active</strong> check box is selected. If there are multiple themes, the theme that you apply takes precedence.</td>
</tr>
<tr>
<td>k.</td>
<td>Click <strong>Submit</strong>.</td>
</tr>
</tbody>
</table>

**Change the navigation and menu settings in your portal experience**

Change the header, footer, and menu settings for your custom portal experience in UI Builder to fit the needs of your organization.

**Before you begin**

Role required: ui_builder_admin

**About this task**

Set up the navigation and menu settings in the app shell of your portal experience. The app shell is the wrapper of the portal contents. For example, the app shell can show things like the logo of your company, user preferences, the search icon, the configuration icon, and the user menu. You can configure this app shell in the Now Platform. For more information about app shells, see [App shells for UI Builder](#).

You need to be in the correct application scope to edit experience settings. If you are in a different scope, the experience settings are read-only. To change your application scope, in the main header, select the application picker ( ), and then select the application scope. For more information about the application scope, see [Security and roles](#).
Procedure

1. Navigate to **Now Experience Framework > UI Builder**.
2. Open a portal experience to work in or create a portal experience by selecting **Create experience in Platform**.
   For more information, see **Create an experience**.
3. Open or create a page in your portal experience.
4. To change certain settings for the portal experience that you are working in, from the menu, click **Edit experience settings**.
   You can fully change all the experience settings in the portal record in the **Now Platform®**.
5. To change the settings for the header, footer, and navigation menu for the app shell of the portal experience you are working in, click **Navigation and menus**.

### Portal experience settings for navigation

<table>
<thead>
<tr>
<th>Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
</tr>
<tr>
<td>Branding and theming</td>
</tr>
<tr>
<td>Navigation and menus</td>
</tr>
<tr>
<td>Search</td>
</tr>
</tbody>
</table>

**This typically displays a logo, an avatar and anything else you have to set as part of your header. Learn more**

**Advanced header settings**

---

**Menu**

You can choose to display a menu as part of this portal. A menu provides a way for users to navigate to key pages in your portal. **Learn more**

**Navigation Menu**

**Advanced navigation menu settings**

---

**Footer**

You can choose to display a footer as part of this portal. A footer provides an expanded view of any important links that you want to display. **Learn more**

**Primary footer**
<table>
<thead>
<tr>
<th>Action</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change the header</td>
<td><strong>a.</strong> To go to the Now Platform®, click <strong>Advanced header settings</strong>.</td>
</tr>
<tr>
<td></td>
<td>Here, you can change what information shows in the header of your portal experience. You can modify things like a company logo, user preferences, the search icon, the configuration icon, and so on.</td>
</tr>
<tr>
<td>Show the menu</td>
<td><strong>a.</strong> To add a menu to your portal experience, select <strong>Navigation Menu</strong>.</td>
</tr>
<tr>
<td></td>
<td><strong>b.</strong> To remove the navigation menu, deselect <strong>Navigation Menu</strong>.</td>
</tr>
<tr>
<td></td>
<td><strong>c.</strong> To go to the Now Platform, select <strong>Advanced navigation menu settings</strong>. Here, you can change what menu and menu items are displayed in the navigation menu. For more information, see <strong>App Shells for UI Builder</strong>.</td>
</tr>
<tr>
<td>Display a footer</td>
<td><strong>a.</strong> To add a primary footer to your portal experience, select <strong>Primary footer</strong>. The footer can show a footer logo, content, and other important links for your portal.</td>
</tr>
<tr>
<td></td>
<td><strong>b.</strong> To go to the Now Platform, click <strong>Advanced navigation menu settings</strong>.</td>
</tr>
<tr>
<td></td>
<td><strong>c.</strong> To add a second footer to your portal experience, select <strong>Secondary footer</strong>. A secondary footer includes additional text links, social media elements, and copyright information.</td>
</tr>
</tbody>
</table>

6. Click **Save all changes**.
Show or hide the search settings for your portal experience

Show or hide the global search functionality on the public or private pages for your custom portal experience in UI Builder so that your users can search on the portal.

Before you begin
Role required: ui_builder_admin

You can control whether the global search functionality is visible to users of your portal experience. You can enable or disable the global search for either public or private pages.

You need to be in the correct application scope to edit the experience settings. If you are in a different scope, the experience settings are read-only. To change your application scope, in the main header, select the application picker ( ), and then select the application scope. For more information about the application scope, see Security and roles.

Procedure

1. Navigate to Now Experience Framework > UI Builder.
2. Open a portal experience to work in or create a portal experience by selecting Create experience in Platform.
   For more information, see Create an experience.
3. Open or create a page in your portal experience.
4. From the menu, click Edit experience settings to change certain settings for the portal experience that you are working in.
   You can fully change all of the experience settings in the portal record in the Now Platform®.
5. To show or hide the global search settings for your portal experience, click Search.
Portal search

- General
- Branding and theming
- Navigation and menus

Search

Global search

You can show or hide the search functionality on any pages found in your portal. By default, all pages show search.

Learn more

Public pages

- Enable global search

Private pages

- Enable global search

Advanced settings

<table>
<thead>
<tr>
<th>Action</th>
<th>Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>To enable your search functionality on any public-facing pages in your portal experience.</td>
<td>Navigate to <strong>Public pages &gt; Enable global search.</strong></td>
</tr>
<tr>
<td>To enable your search functionality on the private pages in your portal experience.</td>
<td>Navigate <strong>Private pages &gt; Enable global search.</strong></td>
</tr>
</tbody>
</table>

6. To go to the Now Platform to make changes to the application record on the Now Platform®, click **Advanced settings.**

To enable or disable the search for a page, you must create a separate UX Page property. The UX Page property overrides the global search setting.

**Using UI Builder**

Get a full understanding of everything you can do when creating a page for your workspace or custom portal experience in UI Builder.

**UI Builder quick start**

Quick start for UI Builder

**Using UI Builder resources**

Find information on how to create a fully functioning page in UI Builder.

The following information will help you along your UI Builder journey.
• Work with pages.
• Create a variant of a page
• Work with components
• Work with layouts
• Work with themes
• Work with data resources
• Work with events

Work with pages
Learn what a page is in Now® Experience UI Builder. Understand the building blocks of a page, such as containers and components.

UI Builder quick start
UI Builder quick start video
Create pages in UI Builder as part of a workspace or custom portal experience. Pages consist of containers and components. You build a page using containers and components to guide a user through an experience. For example, you could build a page to manage travel requests for your employees. The page could have components that contain lists of all travel requests submitted and approved. You add buttons that let users add and submit travel requests. Then apply a company theme to your components. The way you build your page is limitless.

Components and containers
Components are the building blocks of your page. Add components to your page to build or customize your workspace or portal experience. For example, add a button component to your page that lets users submit requests.

You can add components to your page from the Toolbox or from the page itself. As you add components to your page, use the Content tree to easily navigate between components.

A container is a type of component that can contain other components. Containers have layout properties applied to them. Change the page layout and visual styling to make it your own experience.
Types of containers

<table>
<thead>
<tr>
<th>Container Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page</td>
<td>A container that spans a large area. It respects the rules of the app it exists in.</td>
</tr>
<tr>
<td>Container</td>
<td>Add components to a container to create the layout and organizational structure for your page. A page may have multiple containers. Containers can also include containers.</td>
</tr>
<tr>
<td>Modal</td>
<td>Use a modal to create a page type in UI Builder that renders on top of the page. Think of a modal as a pop-over screen that requires action.</td>
</tr>
<tr>
<td>Tabs</td>
<td>Use tabs as a collection of containers that only displays one container at a time.</td>
</tr>
</tbody>
</table>

See Work with components for more information.

**Structure of UI Builder**

UI Builder is separated into the following areas.

- Use the left vertical bar for access to the following in UI Builder.
  - Home screen
  - Page management panel
  - Components
  - Data resources
  - Client state
  - Client scripts
• Utilize the page management panel to create, edit, and manage your pages.

• Work in the main page area to build and modify your pages.
• Go to Page configuration to configure your components, add styling, and set events.

• Use the top bar to access the main menu and Help. Also see the URL, and open your page to preview it. You can also select a domain and set application scope. And finally, click the Save button often to save your page.
Left vertical bar

- Home screen: The landing page for UI Builder includes a list of all of your experiences and links to important documentation.

- Page management panel: Control and manage every aspect of your page. Create pages and variants, edit page settings, and use the content tree to easily manage your page construction.

- Components: See every component available in UI Builder that you can use to build your pages. You can drag components from here onto your page.

- Data resources: Bind data to your components using data resources to dynamically expose your data from tables, records, or other elements on your page. Data resources enable you to reuse your components. See Connect data to your components for more information.

- Client state: Set the parameters for the client state for your page. The client state parameters consist of a name, a type, and an initial value. You can see the preview of the client state parameters.

Page management panel

The page management panel lets you manage all aspects of your pages. You can do the following:

- Create a page.
- Open existing pages in your experience.
- Create a variant of a page.
- View existing variants of a page.
- Edit page settings
• Edit **Required parameters** for a page.
• Edit **Optional parameters** for a page.
• Open a page to preview it in a Web browser.
• If you have the proper role, open a page, variant, variant collection, page definition, or app config in Developer mode.
• Work in the **Content** tree, where you see all of your components in a tree, making it easy to navigate through your page.

**Create page**

Create a page to build a page experience from scratch or from a page template. Build a page one component at a time. If you use one of the pre-defined page templates, you start from a base structure and customize it to meet your needs.

Name your page. Set the path (or keep the default path that is automatically added based on your page name). A default path is added based on your page name. You can also create your own path, but the path must be unique. The **URL preview** shows what the path of your page will be.

The application scope protects applications by identifying and restricting access to application files and data. The application scope defaults to the scope that the user is currently in within the Now Platform®. For more information about application scope, see **Security and roles**.

Use a page template to create a page based on a pre-defined page template, and then customize the page to your needs. You can reference or copy a page template. For more information, see **Page templates**.
Create a page: Advanced settings

Add required parameters to your page. A required parameter is a piece of data that your page requires, such as a sysid, table, or query. Required parameters are useful for components, as they can bind to the value of the required parameter. For example, you can add a table parameter to the URL field, and then bind data to that table. When the table is referenced, it exposes the table data to any components on the page. Required parameters are visible in the URL when you add them to your page. In the following example, a required parameter called table was added. Notice it was appended to the URL.

Add optional parameters to your page. Optional parameters are optional pieces of data that you can add to the URL of your page. Unlike required parameters, optional parameters are always name and value pairs that work no matter what order they’re provided.
Select the **Variant** tab to set the audience and conditions settings for your page. When you create a page, UI Builder also creates a variant of the page for you by default. A page variant is a variation of your page at the same path that lets you target experiences for different audiences using user criteria. For example, a page for managers, and a variant of that page for the manager's direct reports. For more information about creating a variant, see [Create a variant of a page](#).

For more information about audiences, see [Understanding your audiences](#).

**Test values**
Add a test value to your page as a way to bring test data into the page for testing purposes. For example, if you add a table as a required parameter, you could add a test value of incident and bind a data resource to it to bring in test data on the incident for that table.
To get test values to show data, add a data resource, then configure the data resource to bind a record to the test value in the URL. For example, you could add `incident` as a test value. Then add a data resource named **Look Up Record**. In the **Table field**, dynamically bind the `incident` test value to a `context.props.table` table, as shown in the following image.

**View an existing page**
You can work in any page in your experience based on your role settings. To change to a different page, click the list next to page.

**Create a page variant**
A page variant is based on another page and exists at the same path as the page. A variant lets you target different audiences using user criteria. For example, you can create a homepage for agents, and a variant of the page for managers that exists at the same URL. You create a page variant by clicking the **Create** on the page management panel.
You set the audience for each page variant. The audience determines who uses the page variant. For example, if you create a travel request page, create a variant of that page for managers to manage the employee travel requests. You set the audience for the manager page for anyone in the manager role. Employees cannot view that variant. For more information about audiences, see Understanding your audiences.

Page variants are listed under **Variants** in the page management panel. You can change to any variant at any time.

See Create a variant of a page for more information.

**Edit page settings**
Change the settings of your page at any time from the **Menu**. Or, click the three-dot menu to the right of your page name and select **Edit page settings**. You can change the name and path of your page after you create a page.

**Edit required page parameters**
Edit required parameters to your page any time from the three-dot menu next to your page name. A required parameter is a piece of data that your page requires, such as a sysid, table, or query. Required parameters are useful for components, as they can bind to the value of the required parameter. For example, you can add a table parameter to the URL field, and then bind data to that table. When the table is referenced, it exposes the table data to any components on the page. Required parameters are visible in the URL when you add them to your page. In the following example, a required parameter called table was added. Notice it was appended to the URL.
Edit optional parameters
Edit optional parameters to your page any time from the three-dot menu next to your page name. Optional parameters are optional pieces of data that you can add to the URL of your page. Unlike required parameters, optional parameters are always name and value pairs that work no matter what order they’re provided.

Parameters

Optional parameter 1 name  
Enter a name for your optional parameter, like "selectedTabIndex"

Content tree

The content tree in the page management panel is important. It not only shows every container and component on your page, it lets you easily find your components and work with them. The content tree is especially important when you have multiple components on your page. You select the component in your content tree to highlight the component on the page, making it easier to build your page.

When you click a component in the content tree, it highlights the component on your page so you can configure them, add styling, events, data, and so on.

It is important to add a component label when you add a component to your page. The component label is used in the content tree to apply labels and IDs to each component in the content tree. You can identify the components much easier in the content tree when they are labeled. Use the pencil icon in the configuration panel to add or edit a component label.
Main page area
The main page area of UI Builder is where you perform the work when building your pages. It is the largest area of UI Builder. You add your containers and components here by either clicking the + button, or dragging components from the Component section to the page.

Configuration panel
Use the configuration panel to configure your components, add styling to your components, set up event handlers for components, and set the layout of a container.

- Select the Config tab to configure components. Each component has different configuration options that let you control the necessary parts of the component. For example, a button component is simple, and you can only configure how it looks, labels, and some properties. A list component is more complicated. You change dozens of list parameters.
Select the Styles tab in the configuration panel to add styling to a component. You can use standards-based Cascading Style Sheets (CSS) to change the visual style of a component. For example, add or change a background color, a border, or any other CSS style.
See Add styling to your component for more information.

• Select the Events tab to configure events in to add actions to your components, pages, data resources, and declarative actions. When you add components to your page, they are not configured to perform any action. For example, a button component is static and does not do anything until you bind an event action to it, such as deleting a record.

![Button 1](image)

See Work with events for more information.

• Layouts. Layouts govern what and how slots are available in a container on a page. A slot is a section of the page where components exist. When you create a page, you configure how you want the layout designed. Decide how many slots are included and how they are arranged. Once you add a component to a container, you can configure the layout systems in more advanced ways. For example, you can choose to have your content displayed as Flex or Grid. Flex lets you set the direction of the component layout, like up, down, or left right. You can also justify the content, align items, and set the height, width, margins, and padding of your content. CSS grid is the most powerful layout system. CSS Grid is built on top of a two-dimensional grid that gives you power over how you create your pages.
Open a page to preview it
Preview a page to see what the page looks like as a web page. Preview a page to test how your page looks and works as you build it. For example, say you add a Delete record button that opens a modal. The modal asks a user to confirm a deletion. You could open the page to ensure that the button and modal work the way you want them to.

Developer editing
From the Menu, edit the page as a developer on the platform. This option is dependent if you have the proper role as a developer and takes you out of UI Builder. This option lets you change the platform-level details of your page.

See Work with layouts for more information.
Create a page in UI Builder

Create a page in Now® Experience UI Builder for a portal, workspace, or custom application so that you can build a web experience for your users.

Before you begin
Role required: ui_builder_admin

About this task
A page has a collection of components that make up a workspace, portal, or custom application user interface (UI).
Procedure

1. Open UI Builder in any of the following ways.

<table>
<thead>
<tr>
<th>To open UI Builder from the following locations</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Now Experience Framework</td>
<td>a. Navigate to <strong>Now Experience Framework &gt; UI Builder.</strong></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>To open UI Builder from the following locations</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Admin panel of a specific experience</strong></td>
<td>b. Click an experience to work in, or create a new experience by selecting <strong>Create experience in Platform</strong>. For more information about creating an experience, see <strong>Create an experience</strong>.</td>
</tr>
</tbody>
</table>
| **From CSM Configurable Workspace**              | a. Navigate to **Experiences** and select your experience.  
|                                                  | b. Open the record in the **Admin panel**.  
|                                                  | c. Click **Open in UI Builder**.  
|                                                  | a. Navigate to **Workspace Experience > Workspaces > CSM Configurable Workspace**.  
|                                                  | b. Click the **User Menu** in CSM Configurable Workspace.  
|                                                  | c. Click **Edit page** to edit the CSM Configurable Workspace in UI Builder. |
To open UI Builder from the following locations

<table>
<thead>
<tr>
<th>From CSM Configurable Workspace</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. Navigate to <strong>Workspace Experience</strong>  &gt; <strong>Workspaces</strong> &gt; <strong>CSM Configurable Workspace</strong>.</td>
<td></td>
</tr>
<tr>
<td>b. Click the <strong>User Menu</strong> in CSM Configurable Workspace.</td>
<td></td>
</tr>
<tr>
<td>c. Click <strong>Configure Workspace</strong>.</td>
<td></td>
</tr>
<tr>
<td>d. Click <strong>Open in UI Builder</strong>.</td>
<td></td>
</tr>
</tbody>
</table>

2. Click an experience to open it, or create a new experience by selecting **Create experience in Platform**. For more information, see Create an experience.

3. Create a blank page, or use one of the included page templates to get you up and running quicker.

<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a blank page</td>
<td>a. Select <strong>Menu</strong> and select + <strong>Create page</strong>.</td>
</tr>
<tr>
<td></td>
<td>b. On the Create a page screen, in <strong>Name of page</strong>, enter the name of your new page.</td>
</tr>
<tr>
<td></td>
<td>c. In the <strong>Path</strong> field, specify a path for your page. A default path is added based on your page name. You can also create your own path. The path is required and must be unique. The path can include digits (0-9), letters (A-Z, a-z), and a few special characters (&quot;_&quot;, &quot;.&quot;, &quot;~&quot;, &quot;&quot;), with the words separated by a forward slash or hyphen. The <strong>URL preview</strong> shows what the path of your page will be.</td>
</tr>
</tbody>
</table>
To do this task | Do this
--- | ---
Create a page |
First, let’s set up this page.

Note: The application scope defaults to the scope that the user is currently in within the Now Platform®. For more information about the application scope, see Security and roles in UI Builder.

d. Select **Create** to create your page.

When creating a page, you can set advanced settings such as required and optional parameters, as well as variant settings.

a. Select **Add required parameters** to add any required parameters to your page URL.
   A required parameter is a piece of data that your page requires, such as a sys_id, table, or query. Required parameters are useful for components, because they can bind to the value of the required parameter.

For more information, see **Work with pages**.

b. Select **Add optional parameters** to add any optional pieces of data that you want to add to the URL of your page.
   Unlike required parameters, optional parameters are always name and value pairs that work no matter what order that they are provided in.

For more information, see **Work with pages**.
To do this task | Do this
---|---
c. Click the required or optional parameter in the URL, and in **Test values** type a value, such as `incident`.
You add a test value to your page to populate data into the page as a way to test it. For example, if you add a table as a required parameter, you could add a test value of `incident` to bring in test data on the incident for that table.

d. Select the **Variant** tab to set the audience and conditions settings for your page.
When you create a page, UI Builder also creates a variant of the page for you by default. A page variant is a variation of your page at the same path that lets you target experiences for different audiences using user criteria. For example, a page for managers, and a variant of that page for the manager's direct reports. For more information about creating a variant, see **Create a variant of a page**
To do this task | Do this
--- | ---
| Page | Variant |
Visibility |
Audiences |
You can target specific audiences with specific content. Audiences are based on user criteria or roles. |
Add audience |
Conditions |
You can adjust when the page will display. |
Set conditions |

For more information about audiences, see Understanding your audiences.

e. Click **Done**.

Use a page template if you want to create a page based on a pre-defined page template. You can customize the page to your needs. When using a template, either reference the template and its data, or copy the contents of the page template. When you reference a page template, your page automatically updates when you upgrade to a new release. If you copy the page template, your page will not update after upgrade. For more information, see Page templates.

a. Select **Menu** and select + Create page.

b. On the Create a page screen, in **Name of page**, enter the name of your new page.

c. In the **Path** field, specify a path for your page. A default path is added based on your page name. You can also create your own path. The path is required and must be unique. The path can include digits (0-9), letters (A-Z, a-z), and a few special characters ("_", ".", "~", ";"), with the words separated by a forward slash or hyphen.
<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a page</td>
<td>First, let's set up this page.</td>
</tr>
<tr>
<td>Note:</td>
<td>The application scope defaults to the scope that the user is currently in within the Now Platform®. For more information about application scope, see Security and roles in UI Builder.</td>
</tr>
<tr>
<td>d.</td>
<td>From the Page template list, select a template that you want to base your page on. With a page template, you can create a page that is based on a pre-defined page template and then you can customize the page to your needs.</td>
</tr>
<tr>
<td>e.</td>
<td>When you select a template, such as Agent assist, you must reference the template and its data, or just copy the contents of the page template. When you reference a page template, your page automatically updates when you upgrade to a new release. If you just copy the page template, your page does not update after you upgrade.</td>
</tr>
<tr>
<td>f.</td>
<td>Select Create to create your page.</td>
</tr>
<tr>
<td>Advanced settings</td>
<td>When creating a page, you can set advanced settings such as required and optional parameters, as well as variant settings.</td>
</tr>
<tr>
<td>To do this task</td>
<td>Do this</td>
</tr>
<tr>
<td>----------------</td>
<td>---------</td>
</tr>
<tr>
<td>a. Click <strong>Add required parameters</strong> to add any required parameters to your page URL. A required parameter is a piece of data that your page requires, such as a sys_id, table, or query. Required parameters are useful for components, as they can bind to the value of the required parameter.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Required parameters</th>
<th>+Add</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td></td>
</tr>
</tbody>
</table>

b. Click the required parameter in the URL, and in **Test values** type a value, such as *incident*. You add a test value to your page to populate data into the page as a way to test it. For example, if you add a table as a required parameter, you could add a test value of incident to bring in test data on the incident for that table.

For more information on test values, see **Test values**.

c. Click **Add optional parameters** to add any optional pieces of data to your variant that you want to add to the URL of your variant page. Unlike required parameters, optional parameters are always name and value pairs that work no matter what order they’re provided.
To do this task | Do this
---|---

**d.** Select the **Variant** tab to set the audience and conditions settings for your variant page. When you create a page, UI Builder also creates a variant of the page for you by default. A page variant is a variation of your page at the same path that lets you target experiences for different audiences using user criteria. For example, a page for managers, and a variant of that page for the manager's direct reports.

For more information about variants, see Create a page variant.

**e.** Ensure that the **Active** check box is selected to make this page active.

**f.** Select **Done**.

---

**Edit a page**

Edit a page to change the page settings, components, layouts, styles, and data resources.

**Before you begin**

Role required: ui_builder_admin
Procedure

1. Navigate to **Now Experience Framework > UI Builder**.

2. Open an experience to work in. Or, create a new experience by selecting **Create experience in Platform**. See **Create an experience** for more information.

3. Open the page you want to edit.

4. From the **Menu**, select **Edit page settings** to change the name and path of your page. This is useful if you decide the name or path should be changed after you create the page. If your page is in a different application scope, you can choose to edit it in the original scope.

5. **Optional**: You can also change the settings of your page by clicking the three-dot drop-down menu to the right of your page name in the page management panel and selecting **Edit page settings**.

6. From the **Menu**, select **Edit variant settings** to change the settings for any page variants that you have. You can change the name of the variant, change the page template, or change the application scope. You may need to change these settings because the variant was created with the wrong application scope. You may also want to change the page template if the wrong template was incorrectly chosen when the variant was created. Or, you may just want to change the name of the variant.
7. From the Menu, select **Edit experience settings** to change the settings for the entire workspace or portal experience you are working in. See **Experience settings** for more information on both workspace and portal experience settings.

8. Click **Menu > Developer** to edit page and variant settings, page definitions, and app configuration settings on the ServiceNow platform as a developer. This is useful when you need to make changes to the application record itself.

**Test values**

Add test values to your URL as a way to bring test data into a page.

Use test values to provide values for required and optional URL parameters as a way to test your page with actual data.

Test values are important because you can use them to simulate how your page behaves when the page gets its required or optional parameters from the URL. A UI Builder URL is not the same as the URL when you preview your page. So, UI Builder needs a way to simulate what happens to the page during different states of the preview URL.

For example, say that you are building a record page that displays a form for a single record. For the record page to load, the page must have a `<table>` and `<sysId>` in the URL so that it can get and display the proper record. In UI Builder, you must supply test values for the table and sysId so that you can see how the page will look when you preview the page.
To get test values to show data, add a data resource, then configure the data resource to bind a record to the test value in the URL. For example, you could add `incident` as a test value.

Then add a data resource named **Look Up Record**. In the **Table field**, dynamically bind the incident test value to a `context.props.table` table, as shown in the following image.
Page templates

Use page templates to reuse a page definition, such as record or list page, for pages in your workspace or portal.

Before you begin
Role required: ui_builder_admin

About this task
Use a page template to create a page based on a pre-defined page template. Customize the page to your needs. You can reference or copy a page template.

Procedure
1. Navigate to Now Experience Framework > UI Builder.
2. Open an experience to work in. Or, create a new experience by selecting Create experience in Platform. See Create an experience for more information.
3. Create a page using a page template.

<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. Click Menu and select + Create page.</td>
<td></td>
</tr>
<tr>
<td>b. On the Create a page screen, in Name of page, enter the name of your new page.</td>
<td></td>
</tr>
<tr>
<td>c. In the Path field, specify a the path for your page. UI Builder generates a default path based on your page name. UI Builder appends the path to the URL of the page, making your page a unique URL. You can change this path to anything you want. The path must be lowercase, with the words separated by a forward slash or hyphen.</td>
<td></td>
</tr>
<tr>
<td>d. From the Page template list, select a template that you want to base your page on. A page template lets you create a page based on a pre-defined page template, and then customize the page to your needs.</td>
<td></td>
</tr>
<tr>
<td>To do this task</td>
<td>Do the following</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------</td>
</tr>
<tr>
<td>e. When using a template, either reference the template and its data, or copy the contents of the page template. When you reference a page template, your page automatically updates when you upgrade to a new release. If you copy the page template, your page will not update after upgrade.</td>
<td></td>
</tr>
<tr>
<td>f. The <strong>Application scope</strong> defaults to Global. You can change the application scope after you create the page by using the scope picker to the right of the URL field. See <a href="#">Security and roles in UI Builder</a> for more information.</td>
<td></td>
</tr>
<tr>
<td>g. Optional: Click <strong>Add required parameters</strong> to add any required parameters to your page. A required parameter is a piece of data that your page requires, such as a sysid, table, or query. Required parameters are useful for components, as they can bind to the value of the required parameter. You can add test values for the required parameters to test your page.</td>
<td></td>
</tr>
<tr>
<td>h. Click <strong>Add optional parameters</strong> to add any optional pieces of data to your variant that you want to add to the URL of your variant page. Unlike required parameters, optional parameters are always name and value pairs that work no matter what order they're provided.</td>
<td></td>
</tr>
</tbody>
</table>
To do this task | Do the following
---|---

See [Work with pages](#) for more information.

i. **Optional**: Click the required or optional parameter in the URL, and in **Test values** type a value, such as `incident`. You add a test value to your page to populate data into the page as a way to test it. For example, if you add a table as a required parameter, you could add a test value of `incident` to bring in test data on the incident for that table.

For more information on test values, see **Test values**.

j. Click the **Variant** tab to set the audience and conditions settings for your variant page. When you create a page, UI Builder also creates a variant of the page for you by default. A page variant is a version of your page that lets you target experiences for different audiences using user criteria. For example, you can create a homepage for agents, and a variant for managers at the same URL.
To do this task | Do the following
--- | ---
Click Add audience to set the audiences for your variant to fit a specific role based on one or more criteria.
Select the Active check box to make this page active.
Click Done to create your page.

Create a variant of a page
A page variant in Now® Experience UI Builder is a variation of a page that exists at the same path that targets different audiences using user criteria.
Before you begin
Role required: ui_builder_admin

About this task
Learn how to create a page variant in UI Builder. A page variant is based on your page with content targeted for a different audience. For example, you can create a homepage for agents, and a variant of that page for managers of those agents. The variant exists at the same URL.

Procedure
1. Navigate to Now Experience Framework > UI Builder.
2. Open an experience to work in. Or, create a new experience by selecting Create experience in Platform. See Create an experience for more information.
3. Create or open a page.
4. To create a page variant, you can create it from scratch, or use a page template.

<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. Create a variant from scratch</td>
<td>a. Select Menu and select Create variant. Alternatively, you can click + Create next to Variants on the page as shown in the following image.</td>
</tr>
<tr>
<td>b. Create a variant from scratch</td>
<td>b. On the Create a variant screen, in Variant name, enter the name of your variant. The name can be similar to your page name, or what ever you want it to be.</td>
</tr>
<tr>
<td>c. Create a variant from scratch</td>
<td>c. Select Create to create your variant.</td>
</tr>
</tbody>
</table>

Note: The application scope defaults to the scope that the user is currently in within the Now Platform®. For more information about application scope, see Security and roles in UI Builder.
To do this task | Do the following
--- | ---
 | 
page shows different content based on the au­
dience assigned to the page.

For more information about audiences, see Un­
derstanding your audiences.

b. Select **Set conditions** to set criteria that deter­
mines when your page variant displays to users.

**Edit conditions**

By setting these conditions, you are adding additional criteria
to determine when this page will display

**Variant conditions**

e.g. `table=incident^sys_id!=-1` will only show this variant if the
table field is incident, and the sys_id field is not -1

**Order**

100

**Application scope**

Global

For more information on how to edit these con­
ditions, see **Edit conditions**.

c. Add an encoded query string in the **Variant
conditions** field. The query string can be some­
thing like

`table=incident^sys_id!=-1`

This query string only shows this variant if the ta­
ble field is incident, and the sys_id field is not -1.
For more information on encoded query strings, see **Encoded query strings**.

d. Add an order for the condition in the **Order**
field. You may want certain conditions to have
<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td>a higher priority than others. The lower the number, the higher the priority.</td>
<td>Use a page template if you want to create a variant based on a pre-defined page template. You can customize the variant to your needs. When using a template, either reference the template and its data, or copy the contents of the page template. When you reference a page template, your page automatically updates when you upgrade to a new release. If you copy the page template, your page will not update after upgrade.</td>
</tr>
<tr>
<td>e. Select <strong>Done</strong> to create your page.</td>
<td>Use a page template for more information, see Page templates.</td>
</tr>
</tbody>
</table>

For more information, see Page templates.

<p>| a. Select <strong>Menu</strong> and select <strong>Create variant</strong>. Alternatively, you can click + <strong>Create</strong> next to <strong>Variants</strong>. | <strong>a.</strong> Select <strong>Menu</strong> and select <strong>Create variant</strong>. Alternatively, you can click + <strong>Create</strong> next to <strong>Variants</strong>. |
| b. On the <strong>Create a variant</strong> screen, in <strong>Variant name</strong>, enter the name of your new variant. The name can be similar to your page name, or whatever you want it to be. | b. On the <strong>Create a variant</strong> screen, in <strong>Variant name</strong>, enter the name of your new variant. The name can be similar to your page name, or whatever you want it to be. |
| c. From the <strong>Page template</strong> list, select a template that you want to base your page on. A page template lets you create a variant based on a pre-defined page template, and then customize the page to your needs. | c. From the <strong>Page template</strong> list, select a template that you want to base your page on. A page template lets you create a variant based on a pre-defined page template, and then customize the page to your needs. |</p>
<table>
<thead>
<tr>
<th>To do this task</th>
<th>Do the following</th>
</tr>
</thead>
</table>

**d.** Choose to either reference the template and its data, or copy the contents of the page template. When you reference a page template, your variant automatically updates when you upgrade to a new release. If you copy the page template, your variant does not update after you upgrade.

**Note:** The application scope defaults to the scope that the user is currently within the Now Platform®. For more information about application scope, see Security and roles in UI Builder.

**e.** Select **Create** to create your variant.

**a.** Select **Add audience** to set the audience settings for your variant page to fit a specific role based on one or more criteria. The audience is important because it lets you assign your page variant to a specific role, group, and so on. The page shows different content based on the audience assigned to the page.

For more information about audiences, see Understanding your audiences.
To do this task | Do the following
---|---

**b.** Select **Set conditions** to set criteria that determines when your page variant displays to users.

**Edit conditions**

By setting these conditions, you are adding additional criteria to determine when this page will display

**Variant conditions**

* e.g. `table=incident^sys_id!=-1` will only show this variant if the `table` field is `incident`, and the `sys_id` field is not `-1`

**Order**

100

**Application scope**

Global

For more information on how to edit these conditions, see **Edit conditions**.

**c.** Add an encoded query string in the **Variant conditions** field. The query string can be something like

```
table=incident^sys_id!=-1
```

This query string only shows this variant if the `table` field is `incident`, and the `sys_id` field is not `-1`. For more information on encoded query strings, see **Encoded query strings**.

**d.** Add an order for the condition in the **Order** field. You may want certain conditions to have a higher priority than others. The lower the number, the higher the priority.

**e.** Select **Done**.

**Edit conditions**

A page variant has the same URL as a page. Edit the UI Builder page variant conditions to add additional criteria to determine when the page variant displays to users.
Before you begin
Role required: ui_builder_admin

About this task
You can use audience settings to set who sees a page variant. But you also edit the conditions to determine when a page variant is shown. The conditions are based on setting an order, and adding a query string that sets the criteria that must be met for the page variant to display. If you have multiple page variants that all have the same conditions, then the variants will go by the order setting. The following task shows how to set the variant conditions and order.

Procedure
1. Navigate to Now Experience Framework > UI Builder.
2. Open an experience to work in. Or, create a new experience by selecting Create experience in Platform. See Create an experience for more information.
3. Create or open a page variant.
4. In the page management panel, to the right of the variant, select Edit conditions to set criteria that determines when your page variant displays to users.

   Edit conditions

   By setting these conditions, you are adding additional criteria to determine when this page will display

   Variant conditions

   e.g. table=incident^sys_id!=-1 will only show this variant if the table field is incident, and the sys_id field is not -1

   Order: 100

   Application scope: Global

5. Add an encoded query string in the Variant conditions field. The query string can be something like table=incident^sys_id!=-1. This query string will only show this variant if the table field is incident, and the sys_id field is not -1. For more information on encoded query strings, see Encoded query strings.
6. Add an order for the condition in the **Order** field. You may want certain conditions to have a higher priority than others. The lower the number, the higher the priority.

7. Click **Save**.

**Work with components**

Learn what a component is in UI Builder. Also, see how components work within UI Builder.

Components are the base elements of your page. Components range from core elements like buttons and labels to more complex experience components like lists and forms.

You can add these components to your page to build or customize your workspace or portal experience. For example, adding an **Activity stream** component to your page that lets users see their travel request activity.

You can add components to your page in the following ways.
### Ways to add a component to a page

<table>
<thead>
<tr>
<th>Location</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>From the <strong>Components</strong> list in the far left column in UI Builder</td>
<td><img src="image1.png" alt="Image of UI Builder" /></td>
</tr>
<tr>
<td>Directly from a page in UI Builder</td>
<td><img src="image2.png" alt="Image of UI Builder" /></td>
</tr>
<tr>
<td>From the <strong>Content</strong> tree in UI Builder</td>
<td>Selecting a component in the content tree highlights the component.</td>
</tr>
</tbody>
</table>
### Ways to add a component to a page (continued)

<table>
<thead>
<tr>
<th>Location</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Selecting the component in the content tree also exposes the <strong>Config</strong> panel where you configure the component properties.</td>
</tr>
</tbody>
</table>

![Component Configuration Example](image-url)
Ways to add a component to a page (continued)

<table>
<thead>
<tr>
<th>Location</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>From the floating menu above the page in UI Builder</td>
<td><img src="image_url" alt="Image" /></td>
</tr>
</tbody>
</table>

**Containers**
A container is a type of component that you add to your page first. You then add other components to a container to build or customize your page. Think of a container as a defined part of the screen where you add components like lists and headings. You can have as many containers on a page as you want, with as many containers within containers, with as many components in the containers. As you add containers to your page, use the `Content` tree to easily navigate between the containers.
Configure components
There are three ways to configure a component in the configuration panel.

- Configure the component properties.
- Add CSS style overrides.
- Set up event handlers for the components.

Config tab. Component authors configure components to expose the appropriate properties required to set up their components. Component properties vary based on each component. Component configuration can be simple, as is the case with simple elements like buttons, headings, and labels. Experience components like lists and forms require significant configuration.

For more information about configuring components, see Now® Experience Components.

Styles tab. You can change the Cascading Style Sheets (CSS) styles for individual components. Change color borders, font size, and so on.

Events tab. Configure page-level and variant-level event mappings. Also add dispatched events and handled events for your variant. The more complex, experience components rely heavily on dynamic data that is provided by a data resource. Binding dynamic data to a component is an important feature.

You dynamically expose data from tables, records, or other elements on your page. Exposing data enables you to reuse your components. Also, you can point to the configuration fields to see icons data or scripting options for each field.

For each property in the component configuration panel, you can choose from the following options.

- Static: Use data from a list, or enter your own data. The data doesn't connect to an external data source.
- Dynamic data binding: A way to bind a component property to a data resource, page property, or client state.
- Script: Enter JavaScript code to populate a property value.
Component ID
Use the component ID when you add a script or bind data to the component as a way to reference the component. A component ID is automatically created and is based on the component label when you add a component to a page. You can change the component ID to anything you want, as long as it is unique. Click the pencil icon in the Config panel to see the component ID.

Component details

Component label
- Button 1

Component ID
- button_1

Apply Cancel

Component visibility
Select the eye icon in the Config panel to set component visibility. Component visibility is based on a property of the component itself, not who is viewing it. You could show or hide a component based on conditions. For example, hiding an image if it has a broken link.

Set the Test value to test what happens when the visibility is set to true, false, or none.

Test value
- None
- true
- false
- None

You can set the visibility based on dynamic data binding, or by editing a scripted property value.
Add and configure components

Learn how to add components to your page in Now® Experience UI Builder. A page is built by adding components.

Before you begin
Role required: ui_builder_admin

About this task
Learn how to add a component to a page. Once you add a component to a page, you must configure the component. For more information about configuring components, see Components.

Procedure
1. Open UI Builder in any of the following ways.

<table>
<thead>
<tr>
<th>To open UI Builder from the following locations</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>b. Click an experience to work in. Or, create a new experience by select-</td>
</tr>
</tbody>
</table>
To open UI Builder from the following locations | Do this
--- | ---
**Admin panel of a specific experience** | ing **Create experience in Platform**. See **Create an experience** for more information.

| | a. Navigate to **Experiences** and select your experience.
b. Open the record in the **Admin panel**.
c. Click **Open in UI Builder**.

| From CSM Configurable Workspace | a. Navigate to **Workspace Experience > Workspaces > CSM Configurable Workspace**.
b. Click the **User Menu** in CSM Configurable Workspace.
c. Click **Edit page** to edit the CSM Configurable Workspace in UI Builder.

| From CSM Configurable Workspace | a. Navigate to **Workspace Experience > Workspaces > CSM Configurable Workspace**.
b. Click the **User Menu** in CSM Configurable Workspace.
c. Click **Configure Workspace**.
To open UI Builder from the following locations

<table>
<thead>
<tr>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="UI Builder menu" /></td>
</tr>
</tbody>
</table>

d. Click **Open in UI Builder**.

2. Open or create a page. If you open an existing page, ensure you are in the same scope as the original page. If not, change the scope before you start editing the page. Application scoping protects applications by identifying and restricting access to application files and data. Administrators set the scope to specify what parts of an application are accessible to other applications. Application scope protects data and application files. See [Security and roles](#) for more information on application scope.

3. Change the layout of the page or containers as follows. See [Work with layouts](#) for more information.

<table>
<thead>
<tr>
<th>Are you working in an existing page, or a new page?</th>
<th>Then change the layout procedure as follows:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Working in an existing page</strong></td>
<td>Change the layout to add a component to a specific location. For example, say that the page already has a three-column container. Each column has a data visualization component. But, you must add another data visualization component. You can highlight the container and change the layout to four columns. Then, you can add the new data visualization component to the fourth column.</td>
</tr>
</tbody>
</table>
### Are you working in an existing page, or a new page?

<table>
<thead>
<tr>
<th>Then change the layout procedure as follows:</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Layout Procedure" /></td>
</tr>
</tbody>
</table>

**a.** Select the container on your page for which you want to change the layout. Or, you can select the container in the content tree to the left of the main page area. The content tree is often easier to use. When the page structure is in the structured tree, you can more easily find the container you want to change.

**b.** Select the **Layout** tab and select the layout you want to change it.
Are you working in an existing page, or a new page?

Then change the layout procedure as follows:

to. For example, say that the container has a three-column layout. You can change the layout to a four-column layout. Then, you can add a component in that fourth column in your container.

c. Select **Save** often as you work.

d. Select **Open** to see a preview of what it looks like as a web page.

Start by setting the layout you want for the page. For example, you can set your page to have two slots or columns. Then, you can add containers to each slot and change the layout for each container. You can also set the layout at the container level after you add containers to your page.

a. Select the **Body** of your page in the content tree to the left of the main page area. The content tree is a graphical look at how your page
Are you working in an existing page, or a new page?

Then change the layout procedure as follows:

construction. **Body** is the top page-level.

b. Select the **Layout** tab in the page configuration on the left. Layouts govern what slots are available on a page. Layouts also decide how to position slots, and what CSS rules apply to them. You can choose to have your page divided into columns, rows, or any custom design you want. You can also set the layout for a container.

c. Select **Save** often as you work.
Are you working in an existing page, or a new page?

<table>
<thead>
<tr>
<th>Then change the layout procedure as follows:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>d. Select <strong>Open</strong> to see a preview of what it looks like as a web page.</td>
</tr>
</tbody>
</table>

4. Add components to your page. UI Builder comes with a library of components to choose from. You add components as the building blocks of your page. For example, you can add a heading, data visualizations, and so on. The following table shows you the different ways you can add a component to a page.

<table>
<thead>
<tr>
<th>To add a component</th>
<th>Do the following</th>
</tr>
</thead>
<tbody>
<tr>
<td>From the component list</td>
<td>Select the <strong>Components</strong> icon in the left column of UI Builder. Drag a component from the component list onto your page.</td>
</tr>
<tr>
<td>Directly from your page</td>
<td>From your page, select + and choose a component from the list. You can search for a component or scroll through the list.</td>
</tr>
<tr>
<td>To add a component</td>
<td>Do the following</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Floating menu above page</td>
<td>Select the floating menu above your component.</td>
</tr>
<tr>
<td></td>
<td>• Select <strong>Add component</strong>.</td>
</tr>
<tr>
<td></td>
<td>• Choose a component from the component list.</td>
</tr>
<tr>
<td>To add a component</td>
<td>Do the following</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>You can add a component from the content tree in the following two ways.</td>
<td></td>
</tr>
<tr>
<td>• Select <strong>Add component</strong> beneath the container in the content tree. Then choose a component from the component list.</td>
<td></td>
</tr>
<tr>
<td>• Select <strong>Add</strong> to the right of the content tree, then select Component. Then choose a component from the component list.</td>
<td></td>
</tr>
</tbody>
</table>

5. Now configure the properties of the components that you just added. Configuring components means to customize them to your needs.

For more information about configuring components, see [Now® Experience Components](#).

a. Select the component you want to configure.

b. Select the **Config** tab from configuration in UI Builder.
c. Customize the component properties for the component. For example, you could add a name for a button component. Some components, like data visualizations, require a data source before you can configure the properties. Each component has different configuration properties based on the requirements and options for each component. For example, the Button component configuration is simple, while a List component requires more configuration.

For more information about configuring components, see Now® ExperienceComponents.

6. Optional: Select the Events tab to add an event handler to your component. You add an event handler to add actions to components on your page. For example, a button component is static and does not do anything until you bind an event action to it, such as saving a record. Some components do not have an event action applied to them, such as a heading component. But many components require you to map an event to your component to actually make it perform an action. See Bind an event to a component for more information on how to add event handlers to your component.

7. Optional: You can override any styles for a component by adding CSS styling under the Styles tab. For more information, see Add styling to a component.

Note: Style changes only affect a single component at a time. To change the visual style of all components in your experience, you must apply a theme to your experience. For more information, see Apply a theme to your experience.

8. Optional: Add additional containers to your page to display your components in an organized way. For example, one container could have a heading component. Another container below it could include a list component, a button component, and so on.

a. Select the + at the top of your container to add a container component before the existing component. Select the + at the bottom of a component to insert a container component after it.

b. Drag a container component from the Components list to your existing container on the page. Hold over the top line of the container to insert the new container before the existing container. Hold the container over the bottom line of the existing container to add it after.

9. Optional: Add more components to your page. Select the + on the top or bottom of components on the page. The + changes to + Add.
a. Select the + at the top of your component to add a component before the existing component. Select the + at the bottom of a component to insert a component after it.

b. Drag a component from the component list to insert the component before or after an existing component.

10. Optional: To make a modal appear when you select a component such as a button, you must add the modal to the component first. A modal is a confirmation pop-up that appears when you click the component. For example, if you add a button component that deletes a record, you add a modal to ask the user to confirm they want to delete the record. See Modals in Now Experience UI Builder for more information.

11. Select Save often on your page as you work.

12. Select Open to preview your page.

What to do next
You have added and configured components on your page. See work with data resources for more information. A data resource in UI Builder is the data that a page fetches to display content in components. Components use data resources as a way to reuse data and configurations across different experiences, and make the components dynamic on a page.

Add styling to a component
Set CSS styles for a component to change its default appearance.

Before you begin
Role required: ui_builder_admin

About this task
This task describes how to add styles to the wrapper containing your component, which is generally recommended instead of applying styles to the component directly. Make sure that the component whose styles you want to define is placed within a container component to put the component in a wrapper. Your component's wrapper is one level higher than the component in the Content hierarchy and is labeled by default as Main.
To add styling to an entire page, you can use standards-based CSS in the wrapper for the page. To add styling to your entire experience, you must Apply a theme to your experience.

Procedure

1. Navigate to Now Experience Framework > UI Builder.
2. Select an experience from the list, or select + Create experience in Platform to create a new experience and add it to the list. After selecting the experience you want to build, the UI Builder environment appears.
3. If you haven’t created a page for your experience, follow the steps to Create a page in UI Builder and Add and configure components.

   Note: Place your component in a container before adding styles to the component.

Then, in the Content panel, select the wrapper holding the component that you want to add styling to. Your component’s wrapper is one level higher than the component in the Content hierarchy and is labeled by default as Main. The Main Styles configuration panel appears.

4. In the CSS Styles window, enter standards-based CSS properties and values. The following CSS properties are the most commonly used to apply styles for components within containers:

   - background-color
   - background-image
   - border-style
   - border-width
   - border-color
   - border-radius
   - box-shadow
   - height
   - min-height
   - max-height
   - margin
   - overflow
   - padding
   - width
• min-width
• max-width
• z-index

**Important:** Some components contain built-in styling configurations that you cannot override with CSS in UI Builder. To override these style configurations, you must Apply a theme to your experience.

5. In the main header, select **Save** to save your changes.

---

**Work with viewports**

Viewports are specialized components that enable you to extend your experience without needing to own the parent page in Now® Experience UI Builder. You can work with viewports in three ways. You can add a viewport component or a viewport-enabled tab to a page, or add a viewport to the Contextual sidebar component.

Add a viewport component to your page to create separate content even if you do not own the content of the page itself. Create sub-pages, specify audiences, or customize content with different data, routes, and screens. For more information about viewport components, see **Viewport component**.

You can replace Tabs components with Viewport-enabled tabs as a way to display third-party custom data, assign audiences, and create variants. For more information about viewport-enabled tabs, see **Replace a tab with a viewport-enabled tab**.

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Adding viewports within a Contextual sidebar component is unique because you design the viewport in the Edit content mode of UI Builder and not on the page itself.

Edit tabs in a Contextual sidebar component as you would on a page using the Config panel. The main difference is, when you are in Edit content mode of a tab, the page management area changes to Tab management.

Select the menu icon to edit your tab settings, or change the required or optional parameters. For more information about using viewports in a Contextual sidebar component, see Add a viewport to a contextual sidebar component.

Add a Tabs component

Use the Tabs component to add tabs inline on your page. Each tab has its own container. Add components and tabs within a Tabs component on your page. You can move tabs in the Config pane to reorder them on the page.

Before you begin
Role required: admin

Procedure
1. Navigate to Now Experience Framework > UI Builder.
2. Open an experience to work in. Or, create a new experience by selecting Create experience in Platform. See Create an experience for UI Builder for more information.
3. Create or open a page.
4. Add a Tabs component to your page.

For more information on how to add a component to a page, see Add and configure components.
5. Click the **Edit settings** icon to add a tab label and icon to your Tabs component.

**Add label and icon to Tabs component**

*Edit settings for Tab 1*

<table>
<thead>
<tr>
<th>Tab label</th>
<th>Tab 1</th>
</tr>
</thead>
</table>

| Icon (Optional) | Add Item Above Fill |

6. Add a component to your tab, such as a list, button, or heading. The following example shows a Heading component, labeled **Tab 1 heading**.

7. You can also add additional tabs within a Tabs component. Click (+) in the Tabs configuration panel.
8. Add a label for your second tab. The following example shows Tab 2. Click the down arrow to choose an icon for the tab, then click Create when finished.

Add tab label and icon to Tab 2

Add a tab

Set this tab up with a label and icon. The icon is optional.

Tab label: 
Tab 2

Icon (Optional)

No icon

Search

- Add Item Above Fill
- Add Item Above Outline
- Add Item Below Fill
- Add Item Below Outline
- Add Item Left Fill
- Add Item Left Outline
- Add Item Right Fill
- Add Item Right Outline

9. Click the Config panel to configure the tabs.
a. From the Config panel, drag the tabs to reorder them. The tab reorder will then be reflected on the page.

Reorder tabs

<table>
<thead>
<tr>
<th>Tabs</th>
<th>Add</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tab 1</td>
<td></td>
</tr>
<tr>
<td>Tab 2</td>
<td></td>
</tr>
</tbody>
</table>

Default ⌁

- Span tab labels full width of tabs container ⌁
- Hide all tab labels ⌁

b. Click the Default down arrow to choose which tab opens by default when the page loads.

Set default tab

<table>
<thead>
<tr>
<th>Tabs</th>
<th>Add</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tab 1</td>
<td></td>
</tr>
<tr>
<td>Tab 2</td>
<td></td>
</tr>
</tbody>
</table>

Default ⌁

- Span tab labels full width of tabs container ⌁
- Hide all tab labels ⌁
c. Choose where the tab labels appear on the page, and whether to hide tab labels so you only see the icons.

Set tab labels full width, or hide tab labels.

- Span tab labels full width of tabs container
- Hide all tab labels

Results
Your page shows the two tabs you created. Click each tab to further configure them, add styling, or add an event handler. For more information on styling, see Add styling to a component. For more information on adding an event handler, see Events in UI Builder.

Replace a tab with a viewport-enabled tab
Convert a tab on a page to a viewport-enabled tab. Use viewport-enabled tabs to display third-party custom data, assign audiences, and create variants.

Before you begin
Role required: admin

About this task
Replace your tab or tabs with viewport-enabled tabs.

Note:
Replacing a tab with a viewport-enabled tab is a permanent and one-way process. You lose any existing tab content and must recreate the viewport tab content.
Procedure

1. Navigate to Now Experience Framework > UI Builder.
2. Open an experience to work in. Or, create a new experience by selecting Create experience in Platform. See Create an experience for UI Builder for more information.
3. Create or open a page with existing tabs, or add a Tabs component to your page.

   Tabs component
   
   Components
   
   ![Component image]
   
   For more information on how to add a Tabs component to a page, see Add a Tabs component.

4. Click the three-dot icon ( ⋮ ) and select Replace with viewport tabs.

Note:
Replacing a tab with a viewport-enabled tab is a permanent and one-way process. You lose any existing tab content and must recreate the viewport tab content.

**Viewport tab**

<table>
<thead>
<tr>
<th>Tabs 1</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ID: viewport_nrj</td>
<td></td>
</tr>
</tbody>
</table>

[Image of Viewport tab configuration]

5. Click **+ Add** in the Config panel to add a viewport-enabled tab.

For more information on viewport components, see [Viewport component](#).

6. From **Create a tab**, add the following information:

   a. In the **Name of tab** field, type a name for your viewport-enabled tab. For example, **Viewport tab**.

   b. Keep the default path, or change it to your preference.

   c. Choose a **page template** to start with a pre-defined template.

   d. Enter an order for the tab. The lower the number, the higher the priority.

   e. Choose an **icon** to represent the viewport-enabled tab.
f. Click **Create** to create the viewport-enabled tab.

g. In the Success! screen, click the **Variant** tab to add an audience for the viewport-enabled tab. Setting an audience lets you define who can see the viewport-enabled tab. For more information about audiences, see **Understanding your audiences**.
**Viewport-enabled tab audience**

**Success!**

Your tab has been created and we have created the first tab variant for you. Variants render different content for different users at the tab’s URL path.

Apply these additional settings now or later.

<table>
<thead>
<tr>
<th>Tab</th>
<th>Variant</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Visibility</th>
</tr>
</thead>
</table>

**Audiences**
You can target specific audiences with specific content. Audiences are based on user criteria or roles.

**Conditions**
You can adjust when this tab will display.

| order: 0 |

**Access controls**

---

**h. Click Done to finalize your viewport-enabled tab.**

**7. Add components to your tab.** You build your tab just as you build a page.

**8. Add an event handler to any components to make them perform actions, such as loading page content.** For more information about event handlers, see *Events in UI Builder*.

**9. In Data driven tabs, select a data source to bind data to your viewport-enabled tab using data resources to dynamically expose your data from tables and records.** You then bind these data properties to components in your tab. For more information about using data resources, see *Data resources in UI Builder*. 
Data driven tabs

Choose where the viewport-enabled tab labels appear on the page. Also decide whether to hide tab labels so you only see the icons.

Set tab labels full width, or hide tab labels.
- Span tab labels full width of tabs container
- Hide all tab labels

Add a viewport to a contextual sidebar component

Adding a viewport to your page lets you create separate content that is viewed when you click a tab on your page. You create tabs within a component. Then, within those tabs, you can create custom pages or even third-party custom user interfaces.

Before you begin
Role required: ui_builder_admin

About this task
Create a viewport by adding a tab to a contextual sidebar component on your page. Then create content for the tab.

Procedure
1. Navigate to Now Experience Framework > UI Builder.
2. Open an experience to work in. Or, create a new experience by selecting Create experience in Platform. See Create an experience for more information.
3. Create or open a page.
4. Add a contextual sidebar component to your page.
5. Select your component and click **Edit content** to add a tab.

6. Design your content for the tab. Notice you are designing it in the edit content mode of UI Builder and not on the page itself. You can build the tab just like you would build a page. You can add components, change layouts, and so on. This example shows adding a tab to a contextual sidebar component.

7. Optional: Click the **Config** panel to edit the tab you created. When you add viewport tabs to a component in UI Builder, the Config panel changes to show the Tabs configuration. When you click a tab in the Config panel, the tab information changes in the corresponding component on the page.
Viewport component

Add a viewport component to your page to create separate content even if you do not own the content of the page itself. Create sub-pages, specify audiences, or customize content with different data, routes, and screens.

Viewport components use sub-pages within the component. A sub-page is similar to a page variant, but is specific to the viewport component. Treat a sub-page like any other page. Add components, use different layouts, and apply custom data and events to the sub-page to make it specific to your needs.

Viewport components can be added anywhere on a page.

Next step

Work with viewport components

Work with viewport components

Add a viewport component to your page and create a sub-page to create separate content on the page.

Before you begin

Role required: admin

Procedure

1. Navigate to Now Experience Framework > UI Builder.

2. Open an experience to work in. Or, create a new experience by selecting Create experience in Platform. See Create an experience for more information.

3. Open or create a page. If you open an existing page, ensure you are in the same scope as the original page. If not, change the scope before you start editing the page. Application scoping protects applications by identifying and restricting access to application files and data. Administrators set the scope to specify what parts of an application are accessible to other applications. Application scope protects data and application files. See Security and roles in UI Builder for more information on application scope.

4. Add a viewport component. From your page, select + and choose the Viewport component from the list, or drag and drop it to your page. You can search for the component or scroll through the list.
5. Define the audience for the viewport. Click the edit icon to the right of the viewport component in the Variants panel, then click **Edit audiences**.

   a. Select an audience, for example, **Admin**.

   b. Add an order if you want the audience to have a higher priority. The lower the number, the higher the priority.
c. Click **Done** to add the audience.

**Audience info**

**Add audiences**

Select one or more audiences for this page. If an audience you need is not listed, you can choose the Open audiences in platform link to create one.

No audiences, yet.

6. Select the viewport component, then click **Edit content**.
7. Click Create a sub-page to create a sub-page for the viewport component.
a. Name of sub-page: Enter a name for your sub-page. This example uses **Sub page for viewport component**.
b. Path: The path defaults to the name you entered, with hyphens added between words.
c. Optional: Choose a page template if you want to pre-define the sub-page.
d. Icon: Choose an icon to represent the sub-page, or go without an icon.
e. Order: Choose an order for the sub-page. The lower the number, the higher it is prioritized.
f. Application scope. The sub-page is created in the default application scope.
g. Click **Create** to create your sub-page.

8. Build your sub-page. Add components, data resources, and events as needed. Save your sub-page. In the following example, a list component is added to the sub-page.
9. Click **Back to viewport-component** to get out of the sub-page and return to the viewport component. Notice the sub-page is grayed-out to show it is a sub-page.
10. To make the viewport component active, you must add an event handler that contains the route and component ID.

a. Locate the viewport ID and route. You can see the ID and route in the Config panel as shown in the following figure.

b. Select the viewport component in the Content tree, then click **Edit content**.

c. Click **Events**, then select **Add a new event handler**.
d. From **Page-level event handlers**, click **UXF Macroponent Viewport Load Requested**.

**Add event handler**

**Event handler preview:** On, Row Clicked → Select event handler below.

- Search

- **Inherited event handlers**
  - Add alert notifications
  - Remove alert notifications
  - Clear alert notifications
  - Set loading state
  - Modal closed
  - Add parameters to URL
  - Open or close modal dialog
  - Link to destination
  - USER_PREF_UPDATED
  - Modal opened
  - Screen Close Confirmed
  - Form Header Updated
  - Screen Status Changed
  - Session Logged out

- **Page-level event handlers**
  - Add alert notifications
  - Remove alert notifications
  - Clear alert notifications
  - Set loading state
  - Update client state parameter
  - UXF Macroponent Viewport Load Requested

**e.** In **viewportElementID**, replace null with the viewport ID that you located in a previous step. In this example, it is, "viewport_tsj".

**f.** In **route**, replace null with the viewport component route that you located earlier in a previous step. In this example, it is, "Sub page for viewport component".
g. Click Add.

11. The sub-page that you create in the viewport component shows in the Config pane of the page. You can edit sub-pages at any time. The following example shows the Sub page for viewport component example created in this task.
12. Select **Open** to preview your page.

**Results**

1. You added a viewport component to a page.
2. You defined the audience for your viewport component.
3. You then created a sub-page to the viewport component, and added a list component.
4. You added an event handler to make the viewport component active.
5. You saved your work and opened it to preview what it looks like.

**Modals in UI Builder**

Use modals in Now® Experience UI Builder with components to provide alerts or calls to action for a user. UI Builder comes with modals to save time and effort.

A modal window disables the main page in UI Builder. A modal keeps the page visible with the modal window displayed in front of it. Users must interact with the modal before they can return to the main UI Builder page.

UI Builder has preconfigured modals available. You can add a modal to your component. Then, configure the content of the modal, and how it displays on the screen. Add an event handler to the modal to perform an action when a user selects it. The action can alert a user about something, or ask a user to confirm an action. A modal is a way to ensure that a user knows what is happening. For example, a modal may ask a user to confirm a selection before continuing whatever action they are performing on the main page.
**Modal types**

Different types of modals are available in UI Builder, as shown in the following table.

<table>
<thead>
<tr>
<th>Modal type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alert</td>
<td>An Alert modal provides information relating to the component action. For example, when a user presses a delete button, you could have an alert pop-up that lets the user know they cannot undo a delete action.</td>
</tr>
<tr>
<td>Confirm</td>
<td>A Confirm modal asks a user to confirm the component action. For example, when a user presses a delete button, the user would have to confirm the deletion of data. You can choose the confirm options from the primary and secondary button label fields, such as Yes/Cancel.</td>
</tr>
<tr>
<td>Confirm and destroy</td>
<td>A Confirm and destroy modal is more directive, usually relating to deleting or erasing content. It lets the user know the seriousness of an action, and asks them whether they want to proceed with the action.</td>
</tr>
<tr>
<td>Custom</td>
<td>Custom modals address scenarios that are not handled using the standard modals. Custom modals can be thought of as a container component on a modal. You can add a custom layout, components, events, and data resources just like you do on a page. The custom modal uses layouts to let you fully design what information you want in the modal. Layouts also decide where the information sits within the modal screen. You can use Cascading Style Sheets (CSS) styling to change the visual look of the modal.</td>
</tr>
<tr>
<td>Modal type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>iframe</td>
<td>Use iframe to bring content into your modal from existing iframe content from a URL and data.</td>
</tr>
<tr>
<td>Modal viewport</td>
<td>Dynamically pass content into your viewport modal through an event binding using a client script.</td>
</tr>
</tbody>
</table>

**Event handlers and modals**

Exposé events to modals to handle call-to-action events. For example, a primary action, secondary action, and so on. You configure the data by adding an event handler and invoking a data resource. It is as simple as adding a new event handler for the component that has a modal. Or you can add an event handler to the modal itself. You select the event you want associated with the component or modal and add it. See Add modal to component for detailed instructions.

**Add modal to component**

Learn how to add a modal in Now® Experience UI Builder. A modal is a window that appears when you click a component. For example, add a button component that deletes a record. Then, add a modal to ask the user to confirm they want to delete the record.

**Before you begin**

Role required: ui_builder_admin

**About this task**

Add a modal to a component and add an event handler. This task uses a button component with a confirm modal as an example.

**Procedure**

1. Navigate to Now experience framework > UI Builder.
2. Open an experience to work in. Or, create a new experience by selecting Create experience in Platform. See Create an experience for more information.
3. Open or create a page.
4. Add a component to your page, such as a button component. See Add components to a page for more information.
5. Add a modal to the component. A modal is a screen that appears when you click a component such as a button.
a. Select the component to highlight it, either in the content tree, or on the page itself. When you highlight a component, you can configure it.

b. Select **+Add > Modal** in the content section.

c. Choose a modal type such as **Confirm**. You can choose from the six modal types.

<table>
<thead>
<tr>
<th>Modal type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alert</td>
<td>An alert modal provides information relating to the component action. For example, when a user presses a delete button, you could have an alert pop-up that lets the user know they cannot undo a delete action.</td>
</tr>
<tr>
<td>Confirm</td>
<td>A Confirm modal asks a user to confirm the component action. For example, when a user presses a delete button, the user would have to confirm the deletion of data. You can choose the confirm options from the primary and secondary button label fields, such as Yes/Cancel.</td>
</tr>
<tr>
<td>Modal type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Confirm and destroy</td>
<td>A Confirm and destroy modal is more directive, usually relating to deleting or erasing content. It lets the user know the seriousness of an action, and asks them whether they want to proceed with the action.</td>
</tr>
<tr>
<td>Custom</td>
<td>The custom modal uses layouts to let you fully design what information you want in the modal. Layouts also decide where the information sits within the modal screen. You can use Cascading Style Sheets (CSS) styling to change the visual look of the modal, such as background color.</td>
</tr>
<tr>
<td>iframe</td>
<td>Use iframe to bring content into your modal from existing iframe content from a URL and data.</td>
</tr>
<tr>
<td>Modal viewport</td>
<td>Dynamically pass content into your viewport modal through an event binding using a client script. See Bind an event to a component for more information on binding an event to a component.</td>
</tr>
</tbody>
</table>

d. Configure the modals as shown in the table. Configure each modal differently, depending on what the modal requires. Change what information goes into the modal, the size of the modal, and how it looks. You can add an event handler to the modal that performs the action for the modal, such as opening or closing the modal.

<table>
<thead>
<tr>
<th>Modal</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alert</td>
<td>• Add a header, which is the title of the modal.</td>
</tr>
<tr>
<td></td>
<td>• Write the content that you want displayed in the modal. The content tells the user what the alert is.</td>
</tr>
</tbody>
</table>
Modal Configuration

- Add text for the button label. It can be anything you want, such as **OK**, **Yes**, and so on.

- Choose the size of the modal on the screen. Select **Small**, **Medium**, **Large**, or **Fullscreen**.

- Enable or disable **Prevent Default Button Action**, depending on whether you want the modal to close based on the default action.

- Enable or disable **Defer modal content loading**. If you disable it, the modal loads with the page. If you enable it, the modal doesn’t load when the page loads.

- Select **Events > Add a new event handler** to add an event handler to the modal.

- Select an event handler to apply to the modal, then select **Add** to add it to the page. Choose from inherited or page-level event handlers. Event handlers perform an action such as open or close a modal. Depending on the modal type, you can refresh data for the App Shell data source, the
## Modal Configuration

- Add a header, which is the title of the modal.
- Write the content that you want displayed in the modal. The content tells the user what the alert is.
- Add text for the primary button. The primary button is the main action button for users, such as **Yes**, **Add**, and so on.
- Add text for the secondary button. The secondary button is usually the no choice for users, such as **Cancel**, **No**, and so on.
- Choose the size of the modal on the screen. Select **Small**, **Medium**, **Large**, or **Fullscreen**.
- Enable or disable **Prevent Default Primary Button Action**, depending on whether you want the modal to close based on the default action.
- Enable or disable **Prevent Default Secondary Button Action**, depending on whether you want...
## Modal Configuration

The modal to close based on the default action.

- Enable or disable **Defer modal content loading**. If you disable it, the modal loads with the page. If you enable it, the modal doesn't load when the page loads.

- Select **Events > Add a new event handler** to add an event handler to the modal.

- Select an event handler to apply to the modal. Choose from inherited or page-level event handlers. Event handlers perform an action such as open or close a modal. Depending on the modal type, you can refresh data for the App Shell data source, the user session for GraphQL, or a user session for Transform.

<table>
<thead>
<tr>
<th>Event handler preview: On Modal close</th>
<th>Select event handler below</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inherited event handlers</td>
<td></td>
</tr>
<tr>
<td>Links to destination</td>
<td></td>
</tr>
<tr>
<td>Add parameters to URL</td>
<td></td>
</tr>
<tr>
<td>Open or close modal dialog</td>
<td></td>
</tr>
<tr>
<td>Page-level event handlers</td>
<td></td>
</tr>
<tr>
<td>Add alert notifications</td>
<td></td>
</tr>
<tr>
<td>Remove alert notifications</td>
<td></td>
</tr>
<tr>
<td>Clear alert notifications</td>
<td></td>
</tr>
<tr>
<td>Set loading state</td>
<td></td>
</tr>
<tr>
<td>Update client state parameter</td>
<td></td>
</tr>
<tr>
<td>UX App Shell Data Source 1</td>
<td></td>
</tr>
<tr>
<td>Refresh</td>
<td></td>
</tr>
<tr>
<td>Canvas User Session Data Resource</td>
<td></td>
</tr>
<tr>
<td>Refresh</td>
<td></td>
</tr>
<tr>
<td>GraphQL</td>
<td></td>
</tr>
<tr>
<td>Refresh</td>
<td></td>
</tr>
<tr>
<td>Canvas User Session Data Resource</td>
<td></td>
</tr>
<tr>
<td>Refresh</td>
<td></td>
</tr>
</tbody>
</table>

## Confirm or destroy

- Add a header, which is the title of the modal.

- Write the content that you want displayed in the modal. The content tells the user what the alert is.
### Modal Configuration

- **Add text for the primary button.** Primary is the main action button for users, such as **Delete** or **Erase**.
- **Add text for the secondary button.** The secondary button is usually the no choice for users, such as **Cancel, No**, and so on.
- **Choose the size of the modal on the screen.** Select **Small, Medium, Large**, or **Fullscreen**.
- **Enable or disable Prevent Default Primary Button Action,** depending on whether you want the modal to close based on the default action.
- **Enable or disable Prevent Default Secondary Button Action,** depending on whether you want the modal to close based on the default action.
- **Enable or disable Defer modal content loading.** If you disable it, the modal loads with the page. If you enable it, the modal doesn't load when the page loads.
- **Select Events > Add a new event handler** to add an event handler to the modal.
- **Select an event handler to apply to the modal.** Choose from inherited or page-level event handlers. Event handlers perform an action such as open or close a modal. Depending on the modal type, you can refresh data for the App Shell data source, the

<table>
<thead>
<tr>
<th>Modal</th>
<th>Configuration</th>
</tr>
</thead>
</table>
| ![Modal](image) | • Add text for the primary button. Primary is the main action button for users, such as **Delete** or **Erase**.  
• Add text for the secondary button. The secondary button is usually the no choice for users, such as **Cancel, No**, and so on.  
• Choose the size of the modal on the screen. Select **Small, Medium, Large**, or **Fullscreen**.  
• Enable or disable **Prevent Default Primary Button Action,** depending on whether you want the modal to close based on the default action.  
• Enable or disable **Prevent Default Secondary Button Action,** depending on whether you want the modal to close based on the default action.  
• Enable or disable **Defer modal content loading.** If you disable it, the modal loads with the page. If you enable it, the modal doesn't load when the page loads.  
• Select **Events > Add a new event handler** to add an event handler to the modal.  
• Select an event handler to apply to the modal. Choose from inherited or page-level event handlers. Event handlers perform an action such as open or close a modal. Depending on the modal type, you can refresh data for the App Shell data source, the |
Modal Configuration

user session for GraphQL, or a user session for Transform.

- Choose a layout for your modal. You can select no columns, multiple columns, rows, and so on. The layout lets you add content in your modal however you want.

- Use CSS styling to change how your modal looks. You can apply any standard CSS styling to your modal, such as background color.

- Select **Events > Add a new event handler** to add an event handler to the modal.

- Select an event handler to apply to the modal. Choose from inherited or page-level event handlers. Event handlers perform an action such as open or close a modal. Depending on the modal type, you can refresh data for the App Shell data source, the
## Modal Configuration

- Add a header, which is the title of the modal.
- Add a source URL that points to your existing iframe content.
- Set the parameters and initial data that you want to iframe.
- Choose the size of the modal on the screen. Select **Small**, **Medium**, **Large**, or **Fullscreen**.
- Turn on **disable sandbox** to lift the following restrictions: allow-forms, allow-modals, allow-popups, allow-presentation, allow-same-origin, allow-scripts, and allow-downloads options. Turn off the **disable sandbox** to only lift the allow-scripts option.
- Enable or disable **Defer modal content loading**. If you disable it, the modal loads with the page. If you enable it, the modal doesn’t load when the page loads.
- Select **Events > Add a new event handler** to add an event handler to the modal.
Modal configuration

• Select an event handler to apply to the modal. Choose from inherited or page-level event handlers. Event handlers perform an action such as open or close a modal. Depending on the modal type, you can refresh data for the App Shell data source, the user session for GraphQL, or a user session for Transform.

Modal viewport

• Choose the size of the modal on the screen. Select Small, Medium, Large, or Fullscreen.

• Route any viewport contents to your modal by adding a script in the Viewport Contents field.

• Enable or disable Disable Dismiss. If enabled, the modal omits the dismiss icon. Pressing Escape doesn’t close the modal. This property only works for Bare Modals (select Bare as true in Event Mapping).

• Select Events > Add a new event handler to add an event handler to the modal.

• Select an event handler to apply to the modal. Choose from
Modal Configuration

inherited or page-level event handlers. Event handlers perform an action such as open or close a modal. Depending on the modal type, you can refresh data for the App Shell data source, the user session for GraphQL, or a user session for Transform.

6. When you finish configuring the modal, close it. Notice in the content tree that the modals you create sit above the body of your page structure.

Layouts in UI Builder

Layouts control what slots are available in a container on a page, and how these slots are organized. Learn how layouts are used in User Interface (UI) Builder.
Think about the layout of your page before you begin building it. Think about what you want to achieve and how complex the page layout must be. These questions determine what type of layout you will use.

Cascading Style Sheets (CSS) rules apply to layouts. Layouts can also specify empty spaces on a page that components cannot be inserted into. Changing the layout on an existing page will discard the component placements that were made to the page.

Layouts in UI Builder use standards-based CSS web layout technologies such as flexbox, CSS grid, and absolute positioning. So whatever you can do in CSS you can do here. Visit Mozilla to learn more about CSS.

The legacy UI Builder 12-column/row-column layout is also available, although you should use flexbox and CSS grid. For more information about using Flexbox and CSS Grid, see Change the layout of a page.

**Slots and containers in layouts**

Containers are a type of component that contains other components. Containers have layout properties applied to them.

Slots define where containers can be placed in a layout. Slots can hold a component, which could be a container. Slots support layout nesting, which is when one layout is inside another. In this example, your container has slot A and slot B. Slot B has another container inside with four more slots defined.
Types of layout systems

Flexbox is excellent for laying out items in a single dimension, such as a row of items flexibly spaced, or a column of unequally sized items vertically centered and spaced. Flexbox is used with a single slot that has multiple components within it. Components can be spaced in a row or column without needing a separate container with additional slots to maintain and style.

CSS Grid works well for larger layouts with a mix of columns and grids. CSS Grid lets you to position elements across these two dimensions. CSS Grid can also represent a literal grid with X number of columns, and Y number of rows.

A larger layout, such as one for the overall page, can easily incorporate things like a header, main content with two side bars, and a footer that is always at the bottom of your content. Instead of using multiple containers with different layout types applied, you can build a custom page layout that incorporates an overall grid page layout, with room for flex-based slots that will space out your content automatically as needed. You can add additional slots when necessary for additional content instead of containers.

The following table describes the Flexbox and CSS Grid layout systems.

<table>
<thead>
<tr>
<th>Layout system</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flexbox</td>
<td>Flexbox is a one-dimensional layout system in CSS. Flexbox is inherently flexible, which is useful for when you do not know the size of your content. You can change the direction of your content in the layout in the following ways:</td>
</tr>
<tr>
<td></td>
<td>• Row: Rows are organized left-to-right or right-to-left, depending on the direction of your browser's default language. Left/right is the case for an English browser.</td>
</tr>
<tr>
<td></td>
<td>• Row-reverse: Rows are organized in the reverse direction of your browser's default language, such as right-to-left or left-to-right.</td>
</tr>
<tr>
<td></td>
<td>• Column: Up/down or down/up</td>
</tr>
<tr>
<td></td>
<td>• Column-reverse: Down/up</td>
</tr>
<tr>
<td>Layout system</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Flexbox row and column directions</strong></td>
<td></td>
</tr>
<tr>
<td><img src="image" alt="Diagram" /></td>
<td></td>
</tr>
</tbody>
</table>

- **Justify content**: Defines the alignment along the main axis. Choices are as follows:
  - Flex-start: Items are at the start of the flex direction, similar to left-justified content. This is the default setting.
  - Flex-end: Items are at the end of the flex direction, similar to right-justified content.
  - Space-between: Items are distributed evenly.
  - Space-around: Items are distributed evenly with equal space around them.
  - Space-evenly: The spacing between items is equal.

- **Align items**: Defines how you want your flex content displayed along the cross axis. Choices are as follows:
  - Stretch: Stretch your content to fill the container. This is the default setting.
  - Flex-start: Place your content at the start of the cross axis.
  - Flex-end: Place your content at the end of the cross axis.
## Layout systems (continued)

<table>
<thead>
<tr>
<th>Layout system</th>
<th>Description</th>
</tr>
</thead>
</table>
|               | ◦ Center: Center your content in the cross axis.  
|               | ◦ Baseline: Align your content the same as your baseline alignment.  
|               | • Height: Set the height automatically or manually.  
|               | • Width: Set the height of your flexbox items automatically or manually.  
|               | • Margin: Set your minimal distance between flexbox items.  
|               | • Padding: Set the padding for each side of your flexbox items.  
|               | See [Create a Flexbox layout to build a page](#) for more information. |

**CSS Grid**  
CSS Grid is the most powerful layout system in CSS. CSS Grid is built on top of a two-dimensional grid. CSS Grid gives you control over how you create your pages. For example, you can have a grid with three columns and nine cells. Five slots are built on top of the grid. The header slot spans three columns and three cells. A footer slot also spans three columns and cells. And three slots exist in the middle columns, with the middle slot larger than the left and right slots.  
To find out more about CSS layouts within your UI Builder instance, you can find them in the [sys_uib_template] table.
## Change the layout of a page

Add and modify your layout design to change the way your page looks. Choose how components are displayed on a page through Cascading Style Sheets (CSS) web layout technologies such as Flexbox and CSS Grid.

### Before you begin
Role required: ui_builder_admin

### About this task
Layouts control what slots are available on a page, and where they are. CSS rules apply to them. You can change the layout as follows.

### Procedure
1. Navigate to **Now Experience Framework > UI Builder**.
2. Open an experience to work in or create a new experience by selecting **Create experience in Platform**.

---

### Layout systems (continued)

<table>
<thead>
<tr>
<th>Layout system</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSS Grid layout</td>
<td>[Image of CSS Grid layout] &lt;br&gt;See <a href="#">Create a CSS Grid layout to build a page</a> for more information.</td>
</tr>
<tr>
<td>Standard row/column</td>
<td>Standard row/column layout uses the legacy UI Builder 12-column grid and row system. Flexbox and CSS Grid provide you with more control over the page layout. You should use them instead of the legacy standard row/column layout.</td>
</tr>
</tbody>
</table>
See Create an experience for more information on creating experiences.

3. Create or open a page.
4. Select the Layout tab.
5. Depending on the type of page, do one of the following to select a container to update.

<table>
<thead>
<tr>
<th>Type of page</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Existing page</strong></td>
<td>Select the container on your page for which you want to change the layout.</td>
</tr>
<tr>
<td></td>
<td>Note: Alternately, you can select the container in the content tree to the left of the main page area. When the page structure is in the structured tree, you can more easily find the container you want to change. If you have many components and containers in the content tree, use the search bar to find a specific container.</td>
</tr>
<tr>
<td><strong>New page</strong></td>
<td>In the content tree to the left of the main page area, select the <strong>Body</strong> of your page. <strong>Body</strong> is the root element of the page.</td>
</tr>
</tbody>
</table>

If you are starting a new page, you can select the layout at the page level, and later at the container level. If you are not the owner of an existing page, be aware of the impact of changing the layout at the page level.

6. To change the layout of your container, do the following:
a. In the Page configuration pane, select the **Layout** tab to select the layout that you want to use.

You can change the current layout of an existing page into a new one. For example, in a three-column layout, you can click the four-column layout to change the layout.

You can also set the layout for a container. The following image shows the layout options for the UI Builder.

![Layout Options](image)

b. If you are creating a page, add components to the new areas in your layout.

c. Click **Save**.

d. To see a preview of the page, click **Open**.

**Example**

For changing an existing page, the following video shows how you can change the container layout from three to four columns, and then add a new component to the fourth column.
For setting the layout of a new page, the following video shows you how to set the layout that you want for the page. For example, you can set your page to have two slots or columns. Then, you can add containers to each slot and change the layout for each container. You can also set the layout at the container level after you add containers to your page.
7. Add components to slots.
   You build your page with containers components. See Work with components for more information.
   You can add components by using any of the following ways.

<table>
<thead>
<tr>
<th>Option</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component list</td>
<td>a. Select the components icon (מגש) in the left column of UI Builder.</td>
</tr>
<tr>
<td></td>
<td>b. Drag a component from the component list onto your page.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Option</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directly from your page</td>
<td>From your page, select the +Add icon (+Add) and choose a component from the list. You can search for a component or scroll through the list.</td>
</tr>
<tr>
<td>Floating menu above the page</td>
<td>Click the menu icon (Ⅲ) above your component, and choose <strong>Add component</strong>.</td>
</tr>
<tr>
<td>Option</td>
<td>Instruction</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Select + Add component, then select Component.</td>
</tr>
</tbody>
</table>

8. Modify the layout styling options in any of the following ways.
### Change the layout styling

<table>
<thead>
<tr>
<th>Option</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>a.</td>
<td>Click a container for which you want to change the layout. You can click the container in the content tree, or you can click the container on the page.</td>
</tr>
<tr>
<td>b.</td>
<td>From the floating menu above the page, click <strong>Layout</strong>.</td>
</tr>
<tr>
<td>c.</td>
<td>Choose to display your content as either Flex or Grid. For more information about flex or grid options, see Change the layout of a page.</td>
</tr>
</tbody>
</table>

### Choose Flex

<table>
<thead>
<tr>
<th>Display</th>
<th>Flex</th>
<th>Flex</th>
<th>Grid</th>
</tr>
</thead>
</table>

**a.** From the Display list, select **Flex**.

**b.** Select the following options based on how you want to display your container content on the page. The options can also be entered in the CSS styles box.

- **Flex direction:** Choose from **Row**, **Row-reverse**, **Column**, or **Column-reverse**.
- **Justify content:** Choose from any of the following:
  - **Flex-start:** Items are at the start of the flex direction, similar to left-justified content. This is the default setting.
  - **Flex-end:** Items are at the end of the flex direction, similar to right-justified content.
  - **Space-between:** Items are distributed evenly.
<table>
<thead>
<tr>
<th>Option</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Space-around</td>
<td>Items are distributed evenly with equal space around them.</td>
</tr>
<tr>
<td>Space-evenly</td>
<td>The spacing between items is equal.</td>
</tr>
<tr>
<td>Align items</td>
<td>Defines how you want your flex content displayed along the cross axis.</td>
</tr>
<tr>
<td></td>
<td>Choices are as follows:</td>
</tr>
<tr>
<td></td>
<td>◦ Stretch: Stretch your content to fill the container. This is the default</td>
</tr>
<tr>
<td></td>
<td>setting.</td>
</tr>
<tr>
<td></td>
<td>◦ Flex-start: Place your content at the start of the cross axis.</td>
</tr>
<tr>
<td></td>
<td>◦ Flex-end: Place your content at the end of the cross axis.</td>
</tr>
<tr>
<td></td>
<td>◦ Center: Center your content in the cross axis.</td>
</tr>
<tr>
<td></td>
<td>◦ Baseline: Align your content the same as your baseline alignment.</td>
</tr>
<tr>
<td>Height</td>
<td>Set the height automatically, or manually.</td>
</tr>
<tr>
<td>Width</td>
<td>Set the height of your flexbox items automatically or manually.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Option</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Margin: Set your minimal distance between flexbox items.</td>
<td></td>
</tr>
<tr>
<td>• Padding: Set the padding for each side of your flexbox items.</td>
<td></td>
</tr>
</tbody>
</table>

a. From the Display list, select **Grid**.

b. Fill in the following information:
<table>
<thead>
<tr>
<th>Option</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Height</td>
<td>Set the height automatically, or manually.</td>
</tr>
<tr>
<td>Width</td>
<td>Set the height of your grid items automatically or manually.</td>
</tr>
<tr>
<td>Margin</td>
<td>Set your minimal distance between grid items.</td>
</tr>
<tr>
<td>Padding</td>
<td>Set the padding for each side of your grid items.</td>
</tr>
</tbody>
</table>

Create a Flexbox layout to build a page

Create a Flexbox layout in Now Experience UI Builder to build powerful pages so that you can customize with cascading style sheets (CSS) and can improve your performance.

Before you begin
Role required: admin

About this task
You can easily build pages with the layouts that are included with UI Builder. If you plan to build complex pages, you can customize the layout of your page with CSS. Customizing the layout of your page lets you take full advantage of Flexbox so that you can achieve your overall page design. For more information, see Types of layout systems.

In the following procedure, you learn how to use Flexbox to modify your CSS to customize the layout of your page.

Procedure
1. Navigate to Now Experience Framework > UI Builder.
2. Open an experience to work in or create a new experience by selecting Create experience in Platform.
   For more information, see Create an experience for UI Builder.
3. Create a page in UI Builder or open a page.
4. Click the Layout tab and choose the two-column layout.
5. Click **Edit layout code**.

6. Expand the code editor so that you can easily view the CSS code.
7. Add another slot to the layout by copying the CSS code for an existing slot. Copying the code is easier than typing it into the layout.

Copy the CSS code for a slot

8. Below the code that you copied, place a comma and then paste the following CSS code:
   a. Change the "slotName" property to "Column 3",
   b. Change the "flex" property to "2",
   c. Add "margin-left": "1rem".
Changing the "flex" property increases the size of the column. By using "margin-left", you add space between the previous columns.
9. Collapse the expanded view, and then click **Apply**.

Apply a layout change

Update your page's layout

Changes made here will overwrite the existing configuration.

```json
{  "slotName": "Column 3",  "rules": {    "flex": "2",    "margin-left": "1rem"  },  "styles": {    "display": "flex",    "flex-direction": "column"  },  "root": null}
```

Results

The new Flexbox layout that you created shows the new slot that you added to the original two-column layout.
Click **Reset to original** to reset these layout changes back to the original at any time.

**Related information**

- Create a CSS Grid layout to build a page

**Create a CSS Grid layout to build a page**

Create a CSS Grid layout in Now® Experience UI Builder to build powerful pages so that you can customize with cascading style sheets (CSS) and can improve your performance.
Before you begin
Role required: admin

About this task
You can easily build pages with the layouts that are included with UI Builder. If you plan to build complex pages, you can customize the layout of your page by using CSS. Customizing the layout of your page lets you take full advantage of CSS Grid so that you can achieve your overall page design. For more information, see Types of layout systems.

In the following procedure, you learn how to use CSS Grid to modify your CSS to customize the layout of your page.

Procedure
1. Navigate to **Now Experience Framework > UI Builder**.
2. Open an experience to work in or create a new experience by selecting **Create experience in Platform**.
   See [Create an experience for UI Builder](#) for more information.
3. Create a page in UI Builder or open a page.
4. Click the **Layout** tab and choose the two-row by two-column layout.
5. Click **Edit layout code**.
6. Expand the code editor so that you can easily view the CSS code.

7. Add another slot to the layout by copying the CSS code for an existing slot. Copying the CSS code is easier than typing it into the layout.
Copy the CSS code for a slot

```
Copy the CSS code for a slot

8. Paste the CSS code below the code that you copied and change the
   slotName and grid-area to a unique value.
```
9. Modify the layout rules to include the new column and template area, as follows:

a. Delete the "Grid-template-columns": "1fr 1fr", line.

b. Add a second cell1 after cell1, and then add cell5 after cell4.

You can set many different configurations. In this example, there are the two resulting grids from two different grid template areas. The "grid-template-areas" property being modified is setting the CSS property of the same name. For more information, see MDN grid-templates-areas.
10. Collapse the expanded view, and then click **Apply**.

**Apply a layout change**
**Update your page's layout**

---

**Results**
The new CSS Grid layout shows that the new slot was added to the original two-row, two-column layout.
Click **Reset to original** to reset the layout changes back to the original at any time.

**Related information**

Create a Flexbox layout to build a page

**Themes in UI Builder**

Themes enable you to change the visual style of your app's experiences so that they express the look and feel of your brand.

**Note:** You can only apply themes to CSM Configurable Workspace and the Workspace and Portal experiences for App Engine Studio. To create a theme for a mobile app, see Mobile themes.
Getting started with themes
Before creating a theme to apply to your experience, create a theming plan for all of the experiences that your users interact with. Your theming plan can consider the following questions:

• What kinds of experiences do my users interact with?
• What similarities and differences should the visual styles of these experiences have?

For example, if you just want to create one theme that represents a singular branding look and feel throughout all of your experiences, then you might want to only create a main theme. However, if you want to represent multiple branding variations that are based on an overall brand style, then you can create a main theme and then extend that theme to create theme variations.

Creating your main theme
Your main theme represents the overall look and feel of your organization’s brand. When creating a main theme, consider the following:

• Which brand colors do you want to display to users for the following types of elements?
  1. Primary, or the most important, actions.
  2. Secondary, or complementary, actions.

After choosing your brand colors, you can then assign color values, as RGB triplets, to the following Now® Experience CSS custom properties for theming.

Now® Experience CSS custom properties for theming

<table>
<thead>
<tr>
<th>CSS custom property</th>
<th>RGB color value to assign</th>
</tr>
</thead>
<tbody>
<tr>
<td>--now-color_brand--primary</td>
<td>Your primary brand color.</td>
</tr>
<tr>
<td></td>
<td>Note: A saturated, darker color works best, because this color frequently appears on a light background. Dark backgrounds are not currently supported.</td>
</tr>
<tr>
<td>--now-color--primary-1</td>
<td>Same as previous. Your primary brand color.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Now® Experience CSS custom properties for theming (continued)

<table>
<thead>
<tr>
<th>CSS custom property</th>
<th>RGB color value to assign</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Note: A saturated, darker color works best, because this color frequently appears on a light background. Dark backgrounds are not currently supported.</td>
</tr>
<tr>
<td>--now-color--primary-0</td>
<td>Slightly lighter variation of your primary brand color.</td>
</tr>
<tr>
<td>--now-color--primary-2</td>
<td>Slightly darker variation of your primary brand color.</td>
</tr>
<tr>
<td>--now-color--primary-3</td>
<td>Darkest variation of your primary brand color.</td>
</tr>
<tr>
<td>--now-color_brand--secondary</td>
<td>Your secondary brand color.</td>
</tr>
<tr>
<td>--now-color--secondary-1</td>
<td>Same as previous. Your secondary brand color.</td>
</tr>
<tr>
<td>--now-color--secondary-0</td>
<td>Slightly lighter variation of your secondary brand color.</td>
</tr>
<tr>
<td>--now-color--secondary-2</td>
<td>Slightly darker variation of your secondary brand color.</td>
</tr>
<tr>
<td>--now-color--secondary-3</td>
<td>Darkest variation of your secondary brand color.</td>
</tr>
</tbody>
</table>

You can then use these CSS custom properties in a Theme [sys_ux_theme] record to Create a custom theme.

Note: Your instance doesn’t apply your main theme to all of your experiences automatically. To learn more about how to apply your main theme to an experience, see Apply a theme to your experience.

The following images demonstrate where Now® Experience CSS custom properties appear in a Workspace and Portal. Refer to these images when determining which CSS custom properties to use in your theme.
Where CSS custom properties appear in a Workspace experience

Workspace

Note: To change the banner image, you must do the following:

1. Upload a new image at System UI > Images.
2. From the Image [db_image] record, copy the value of the Name field.
3. In UI Builder, select the Hero Banner's Row container.
4. In CSS styles, set the value of the background-image property to the URL for the copied image's name. For example, if your image's name is my_image.png, set the value of background-image to url(my_image.png).

Creating variations on your main theme
You can create variations on your main theme depending on the type of experience that you're building for your users. When creating a variation on a main theme to apply to an experience, consider the following questions:

- What roles do your users have in this experience?
- What tasks are your users focusing on the most in this experience?
- How might your users be feeling when working in this experience?

Asking your team these questions can help you better understand your approach for creating a variation on your main theme. For example, assume you’re creating a theme variation for fulfillers who have strict SLAs for completing actions in a Workspace. Then, you should consider calling more attention to primary and secondary actions in the Workspace by transforming all button text to uppercase.

The following is an example of how to style a main theme and two variations on a main theme:

---

**Main Workspace theme**

- Primary brand color: blue
- Secondary brand color: purple

**Workspace Theme A**
- Inherits: Primary brand color: blue
- Overrides: Secondary brand color: green
- Adds: More rounding to button borders

**Workspace Theme B**
- Inherits: Primary brand color: blue
- Secondary brand color: purple
- Adds: Thicker border to menus
- All uppercase text to button labels

---

**Note:** Your instance doesn't apply your main theme and theme variations to experiences automatically. To step through an example of creating a main theme and a variation on a main theme, see Extend a theme.
**Theme variables in CSS editor**

When you define theme variables in the [sys_ux_theme] table, you can use them directly in the UI Builder Styles CSS editor to add styling to your pages.

UI Builder will intelligently predict or suggest theme variables that you defined in your theme, as well as CSS properties and values to you as you type your CSS configuration.

Use the open icon () icon to expand the CSS styles editor in a modal. The expanded modal provides a search option and a larger space to work in so you can fully view the variables. The expanded editor also shows two tabs. The Editor tab shows a full view of the CSS code. The Variables tab shows the variables defined by the user’s theme.

### Edit styles

**CSS styles**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Theme Preview template

Create a new page and apply the Theme Preview page template to see how an applied theme looks on a page.
The theme preview page shows how common components will look with the theme applied to them. For example, a theme preview page will look similar to the following, depending on what theme you have defined.

Create a custom theme

Create a custom theme so that you can override default component styles. You can define your own style properties in a theme record.
Before you begin
Role required: ui_builder_admin or admin

About this task
Themes enable you to change the visual style of your app’s experiences so that they express the look and feel of your brand. Before creating a own custom theme for only one experience, consider creating a theming plan for your app. For more information, see Themes in UI Builder.

Note: You can apply themes to CSM Configurable Workspace and only to experiences created in App Engine Studio.

Procedure
1. To apply a theme, do one of the following:
   - Follow the steps to Apply a theme to your experience:
     a. Navigate to Menu > Edit experience settings > Branding and theming.
     b. On the Brand and theming settings screen under Themes, select Create a theme.
   - Navigate to Now Experience Framework > Themes.
The Themes list view appears.
2. To create a new theme record in the Theme [sys_ux_theme] table, click New.
3. On the form, fill in the fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name that describes your theme. This name appears in the list of selectable themes in the UI Builder Experience Settings menu.</td>
</tr>
<tr>
<td>Description</td>
<td>Detailed description for your theme.</td>
</tr>
<tr>
<td>Extends</td>
<td>Another theme record that inherits from your styling configurations. To inherit default ServiceNow fonts and component spacing, select the lookup using list icon and then select the Default theme from the list. For more information on how to extend a theme from your main custom theme, see Extend a theme.</td>
</tr>
<tr>
<td>Application</td>
<td>Application that contains your experience.</td>
</tr>
<tr>
<td>Active</td>
<td>Leave selected.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>Domain</td>
<td>Domain that is associated with your create. For more information, see domain-separated themes.</td>
</tr>
<tr>
<td>Override</td>
<td>This field should be left empty.</td>
</tr>
</tbody>
</table>
| Theme   | Enter Now® Experience CSS custom properties and values for your theme, formatted as a JSON object.  
When configuring your theme in this field, consider the following questions:  
- Which brand colors do you want to display to users for the following types of elements?  
  a. Primary, or the most important, actions.  
  b. Secondary, or complementary, actions.  
After choosing your brand colors, you can then assign color values, as RGB triplets, to the following Now® Experience CSS custom properties for theming. |

**Now® Experience CSS custom properties for theming**

<table>
<thead>
<tr>
<th>CSS custom property</th>
<th>RGB color value to assign</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>--now-color_brand--primary</code></td>
<td>Your primary brand color.</td>
</tr>
<tr>
<td><strong>Note:</strong> A saturated, darker color works best, because this color frequently appears on a light background. Dark backgrounds are not currently supported.</td>
<td></td>
</tr>
<tr>
<td><code>--now-color--primary-1</code></td>
<td>Same as previous. Your primary brand color.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><strong>CSS custom property</strong></td>
<td><strong>RGB color value to assign</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> A saturated, darker color works best, because this color frequently appears on a light background. Dark backgrounds are not currently supported.</td>
</tr>
<tr>
<td>--now-color--primary-0</td>
<td>Slightly lighter variation of your primary brand color.</td>
</tr>
<tr>
<td>--now-color--primary-2</td>
<td>Slightly darker variation of your primary brand color.</td>
</tr>
<tr>
<td>--now-color--primary-3</td>
<td>Darkest variation of your primary brand color.</td>
</tr>
<tr>
<td>--now-color_brand--secondary</td>
<td>Your secondary brand color.</td>
</tr>
<tr>
<td>--now-color--secondary-1</td>
<td>Same as previous. Your secondary brand color.</td>
</tr>
<tr>
<td>--now-color--secondary-0</td>
<td>Slightly lighter variation of your secondary brand color.</td>
</tr>
<tr>
<td>--now-color--secondary-2</td>
<td>Slightly darker variation of your secondary brand color.</td>
</tr>
<tr>
<td>--now-color--secondary-3</td>
<td>Darkest variation of your secondary brand color.</td>
</tr>
</tbody>
</table>

The following images demonstrate where Now® Experience CSS custom properties appear in a Workspace and Portal. Refer to these images when determining which CSS custom properties to use in your theme.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>

Where CSS custom properties appear in a Workspace experience

**Workspace**

![Workspace Image]

Where CSS custom properties appear in a Portal experience

**Portal**

![Portal Image]

**Note:** To change the banner image, you must do the following:

a. Upload a new image at **System UI > Images**.

b. From the Image [db_image] record, copy the value of the **Name** field.

c. In UI Builder, select the Hero Banner's Row container.

d. In CSS styles, set the value of the **background-image** property to the URL for the copied image's name.

For example, if your image's name is **my_image.png**, set the value of **background-image** to **url(my_image.png)**.

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The following is an example theme configuration snippet.

```json
{
    "--now-color_brand--primary": "23,64,139",
    "--now-color--primary-1": "23,64,139",
    "--now-color--primary-0": "220,226,238",
    "--now-color--primary-2": "18,48,105",
    "--now-color--primary-3": "12,32,70",
    "--now-color_brand--secondary": "201,8,42",
    "--now-color--secondary-1": "201,8,42",
    "--now-color--secondary-0": "247,218,223",
    "--now-color--secondary-2": "151,6,32",
    "--now-color--secondary-3": "151,6,32"
}
```

**Note:** Hex color values are not supported in a Theme [sys_ux_theme] record. All color values must be RGB triplets.

4. To finish creating your theme, select **Submit**.

**What to do next**
Apply your theme to an experience so that your custom styles get applied to all of the visual elements in your app. For more information, see **Apply a theme to your experience**.

**Extend a theme**
Create a variation on your main theme by extending it to another theme.

**Before you begin**
Role required: ui_builder_admin or admin

**About this task**
Extending a Theme [sys_ux_theme] record to another Theme record enables you to create a variation on your main theme for a specific experience. After extending a theme, you can then override inherited styles from the main theme by setting more specific styles in the extended theme variation. For example, you can create your organization's main branding theme. You can then extend that theme to a Workspace theme with some customized styles specific to the Workspace.

The following procedure walks you through an example of how to create a main theme and then extend that theme and customize its style configurations for a
Workspace. For more information on how to create and extend themes using Now® Experience CSS custom properties, see Themes in UI Builder.

**Procedure**

1. Do one of the following:
   - Follow the steps to Apply a theme to your experience:
     a. Navigate to **Menu > Edit experience settings > Branding and theming.**
     b. On the Brand and theming settings screen under Themes, select **Create a theme.**
   - Navigate to **Now Experience Framework > Themes.** The Themes list view appears.

2. Click **New** to create a new Theme [sys_ux_theme] record.

3. On the form, fill in the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter the name of your theme extension, for example, <em>My Main Theme</em></td>
</tr>
<tr>
<td>Description</td>
<td>Describe the theme extension.</td>
</tr>
<tr>
<td>Extends</td>
<td>Select the lookup using list icon (🔍) and then select the <strong>Default</strong> record from the list.</td>
</tr>
</tbody>
</table>
   | Theme   | Enter the following:

```plaintext
{
    "--now-color_brand--primary": "23,64,139",
    "--now-color--primary-1": "23,64,139",
    "--now-color--primary-0": "220,226,238",
    "--now-color--primary-2": "18,48,105",
    "--now-color--primary-3": "12,32,70",
    "--now-color_brand--secondary": "201,8,42",
    "--now-color--secondary-1": "201,8,42",
    "--now-color--secondary-0": "247,218,223",
    "--now-color--secondary-2": "151,6,32",
    "--now-color--secondary-3": "151,6,32"
}
```
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong></td>
<td>Hex color values are not supported in a Theme [sys_ux_theme] record. All color values must be RGB triplets.</td>
</tr>
<tr>
<td>Active</td>
<td>Select to make this theme extension active.</td>
</tr>
<tr>
<td>Override</td>
<td>Leave this field empty.</td>
</tr>
</tbody>
</table>

4. Leave all other fields as they are, and then select **Submit** to finish creating the main theme. The Themes list view appears again.

5. In the context header, select **New** to create another Theme record.

6. In the form, fill in the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter <strong>My Workspace Theme</strong></td>
</tr>
<tr>
<td>Description</td>
<td>Describe the theme extension.</td>
</tr>
<tr>
<td>Extends</td>
<td>Select the lookup using list icon (🔍) and then select the <strong>My Main Theme</strong> record from the list.</td>
</tr>
<tr>
<td>Theme</td>
<td>Enter the following code:</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;--now-actionable--text-transform&quot;: &quot;uppercase&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;--now-actionable--border-radius&quot;: &quot;30px&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;--now-actionable--border-width&quot;: &quot;5px&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;--now-window--border-radius&quot;: &quot;30px&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;--now-window--border-width&quot;: &quot;5px&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

**Note:** Hex color values are not supported in a Theme [sys_ux_theme] record. All color values must be RGB triplets.
7. Leave all other fields as they are, and then select **Submit** to finish creating the Workspace theme.
   Your main theme now extends to your Workspace theme. Styles set in your main theme are inherited by the Workspace theme unless a more specific style in the Workspace theme overrides the inherited style.

**What to do next**
Now that you’ve created a main theme and an extended theme variation, try applying a theme to your experience.

**Apply a theme to your experience**
Apply a theme to change the visual style of your experience and provide a consistent look and feel across all pages.

**Before you begin**
Role required: ui_builder_admin or admin

**About this task**
Themes enable you to override default component styles, such as fonts and colors. You can create a consistent look and feel across your entire experience.

With Theme [sys_ux_theme] records, you can set style properties that are passed down to all of the visual elements in your experience, without needing to rebuild components from scratch. For more information on creating your own theme, see [Create a custom theme](#).

When you define theme variables in the [sys_ux_theme] table, you can use them directly in the UI Builder Styles CSS editor to add styling to your pages. For more information about working with theme variables in the CSS styles editor, see [Themes in UI Builder](#).

**Procedure**

1. Navigate to **Now Experience Framework > UI Builder**.

2. To select an experience that you want to build, do one of the following:
   - Select an experience from the list.
   - Select + **Create experience in Platform** to create a new experience, and then add it to the list.
   The UI Builder environment appears.

3. Make sure that your selected application scope is the same scope that your experience is associated with.
   To change your application scope, do the following:
a. In the main header, select the application picker ( ),
b. Select the appropriate application scope.

4. In the UI Builder main header, select **Menu > Edit experience settings**. The Experience Settings screen appears.

5. From the settings menu, select **Branding and theming**.

6. From the Themes list, select the theme that you want to apply to all pages in your experience.
   The following themes are available for you to select:
   - Default: The default ServiceNow theme.
   - Base Agent Workspace theme: The default theme for a Workspace experience.
   - If you want to create a new theme, select **Create a theme**. See **Create a custom theme**.

7. After selecting a theme to apply, select **Save all changes**.

8. **Optional**: To view your theming changes, select **Open** in the URL header.

## Data resources in UI Builder

You can bind data to your page by using data resources to dynamically expose your data from the tables and records on your page. You then bind these data properties to the components on your page so that you can reuse your components in User Interface (UI) Builder.
Learning about data resources

A data resource is the data that a page fetches to display the content in the components. The components use these data resources as a way to reuse the data and configurations across different experiences. Data resources make the components dynamic on a page, which means that you don’t have to create a component for every page.

You can bind the configuration properties for components, other data resources, client scripts, client state, and events to these data resources.

Set conditions for a filter in your data resource. For more information, see Connect data to your components.

How data resources work

The data resources fetch the data from Glide, GraphQL, as well as the REST APIs. The data resources then update the data that it fetches and transforms the data for use in a component on a page.

Both inherited data resource instances and local data resource instances are used in components. The inherited data resources are automatically brought in through your page. The local data resource instances enable you to add and configure additional data resources. You can also specify when the resources get evaluated and supply the input values so that the data resources know how to get the right data. For more information about inherited and local data resources, see Inherited versus local data resources.

Local data resources

You can choose local data resources, such as the server data, operations, transforms, or client data, like the gForm API, to bring your own data to your page.

The data resources are organized by the application that owns them. They are then further organized by the data resource type like Server data or Transform. For example, the Global application has several data resources but the Customer Service Management (CSM) Workspace application has only a few data resources.
You see different data resources depending on the application that you are in. If you select the Global application, you see different data resources under Server data than if you had selected the CSM Configurable Workspace application and Server data.

The following table shows the data resources that are available for the Global application.

### Global data resources

<table>
<thead>
<tr>
<th>Data resource</th>
<th>Definition</th>
</tr>
</thead>
</table>
| Server data   | Server data resources that grab data from a server. They include:  
  - Aggregation Query  
  - Build route map  
  - Canvas_User_Session_DB_GQL  
  - Dynamic Routing  
  - EVAM Data Resource  
  - Fetch EVAM Data  
  - Fetch EVAM Metadata |
<table>
<thead>
<tr>
<th>Data resource</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Filtered Incidents</td>
</tr>
<tr>
<td></td>
<td>• GlideRecord Collection Query</td>
</tr>
<tr>
<td></td>
<td>• GlideRecord Query</td>
</tr>
<tr>
<td></td>
<td>• Look Up Properties</td>
</tr>
<tr>
<td></td>
<td>• Look Up Records</td>
</tr>
<tr>
<td></td>
<td>• Look Up Users</td>
</tr>
<tr>
<td></td>
<td>• Look Up User Preferences</td>
</tr>
<tr>
<td></td>
<td>• Search EVAM Data Resource</td>
</tr>
<tr>
<td></td>
<td>• Table route map</td>
</tr>
</tbody>
</table>

**Operations**

Operations data resources that let you perform actions. They include:

- Create Record
- Delete Multiple Records
- Delete Record
- Update Multiple Records
- Update Record

**Transform**

Transform data resources that let you perform transforms. They include:

- Bind data to component prop
- Canvas_User_Session_DB_Transform
- Data Driven Items
- Data Driven Tabs Transform
- Get Incident Caller Greetings
- Now DateTime
- Substitute Query Variables

**Client data**

Client data resources that include:

- Service Catalog Glide Form
Using data binding

You can use data binding to bind properties to static components to make a dynamic page. When you first add a component to a page, it is static, which means that it doesn't perform any action. You add data resources to fetch data from the back end of your instance to connect a component to data. For example, a Content tree component doesn't show any data until you bind it to a data resource that pulls in the content. You use field parameters to get data resource properties from the URL, like a table name or sys_id.

You can bind data to a component in the following ways:

Context binding. Use required or optional URL parameters to bind context property fields in the URL. For example, you bind the table name from the URL into your component by using @context syntax, such as @context.props.table.

For example, let's say that your page has a required field on it called table. The URL that resolves to your page would then be something like /demo/page/<table-name>. The <table-name> could be incident. Data could also come from the parent data resources or be local page properties that don't map to anything. You bind component properties, other data resource properties, or event payload properties to the page property with a @context.props.table context binding. If you are using a context binding, make sure that you have supplied a test value on the URL or a static value for that property in the body configuration of your page.

Data resource binding. Add data resources to fetch data from the back end of your instance, such as Client state, GraphQL, or a REST API. Data resources expose properties that you can bind to components and elements on a page.

For example, let's say that you have a Lookup Record data resource. In a button component, you could use a data expression in the label prop like @data.lookup_record_1.result.number.displayValue.

Component binding. Use component binding to bind one component to another. Let's say that you have a List Menu component on the page. It exposes the currently selected list to other components on the page. Other components could get the data by binding to it with an expression such as @elements.list_menu_1.selectedListId.
Client state parameter binding. Use `@state` syntax to bind a state property to a client state parameter.

**Types of data sources available in UI Builder**
You can use the following data resource types that are shown in the table.

<table>
<thead>
<tr>
<th>Data resource type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GraphQL</td>
<td>GraphQL queries and mutations that are executed.</td>
</tr>
<tr>
<td>Transform</td>
<td>Script that transforms the input data into another format.</td>
</tr>
<tr>
<td>Client state</td>
<td>Client-side data resources that include the client information, domain-specific states or logic, user preferences, and so on.</td>
</tr>
<tr>
<td>Composite</td>
<td>Single reusable data resource that contains multiple data resources.</td>
</tr>
<tr>
<td>REST</td>
<td>Data resources that are made through REST API requests.</td>
</tr>
</tbody>
</table>

**Inherited versus local data resources**
Inherited data resources come from the parent page or pages that surround your current page. For example, there may be data resources coming from the application shell of your UX application or data resources from the page surrounding a tab set. Inherited data resources are available for you to use and expose to your page by binding to them in the body properties for your page and mapping them to local page (context) properties. You do not need to add your own data resource if the data you need is already being supplied through inheritance.

Local data resource instances are data resources that you add to the page. Think of local data resources as specific data resources that you add yourself. You expose local data resources to a component on a page.

For example, if you are creating a travel request page, you can expose employee records to a list component. Employees can then create travel requests that are associated with their own employee data.
You can add a data resource from an application or you can create a local data resource in ServiceNow platform Lists and Forms. You then use UI Builder to expose that data for the components on your page.

Once the data is available to your page, bind the data to your component so that the component can use the data. For example, you can create a data resource for a set of records. You then expose that set of records in UI Builder. Finally, you bind that set of records to a component. You then configure the component to perform an action on that data, such as save a new record.

**Data resource properties**

Data resource properties can be thought of as how the data resource fetches the data. The properties of a data resource get exposed when you add the data resource to UI Builder. The properties define everything about the data resource. For example, the properties of a data resource could be the tables, conditions, how to order them, and so on.

**Data resource scripts**

When you add a data resource to your page, you see a preview of the outputs of that data resource if all the required properties have been configured. The output preview is a good way to see how the data gets retrieved so that you can quickly determine if the data you need is present.

**Next steps**

Connect data to your components

Define and bind client scripts to components

Create and bind a client state parameter to a component

**Connect data to your components**

Add data resources so that you can dynamically expose your data from tables, records, or other elements on your page in User Interface (UI) Builder. You can bind that exposed data to components on your page and later reuse those components elsewhere.

**Before you begin**

Role required: ui_builder_admin

**Procedure**

1. Navigate to Now Experience Framework > UI Builder.
2. Open an experience to work in or create a new experience by selecting Create experience in Platform.
For more information, see Create an experience for UI Builder.

3. Create or open a page.

4. Add a component to your page, such as a button.
   You need a component on your page before you can bind a data resource to it.

5. From the lower-left pane, select Data.
   In the Data resource instances section, you can add and configure data resource instances for your page. If you have any inherited data resources, you see them listed in the inherited data resource pane but they are read-only. You do not configure inherited data in UI Builder.

6. To add a data resource to your page, select + Add.

7. Choose an application to select a data resource from.
   The Global application contains the most common data resources. The data resources relate to Server data, Operations, and Transform. You can add a data resource such as Look Up Records, Table route map, GlideRecord Query, and so on.

8. Select a data resource to add to your page.
   For example, under Server data, you could select Lookup Record. Select Add to add it to your page.

9. Click the Configurations tab to configure the data resource and then do the following actions:

   a. From the When to evaluate this data resource list, select when you want to evoke the data resource.

   b. To set the conditions for a filter for your data resource, click Edit incident filter.
For more information about using the condition builder, see Condition builder.

10. To set up an event handler for your component, click + Add a new event handler.
   The following example shows how to add an event handler for a button component.

11. From the Event handler preview section, select an event handler to bind to your component.
   For more information about event handlers, see Configure an event handler.
For more information about binding an event to a component, see Bind an event to a component.

12. **Configure the event handler and click Add.**
You configure each event handler differently, depending on the action of the event. For example, if you add an event handler for a button component, you can choose what that button action performs.
For more information about binding an event to a component, see Bind an event to a component.

**Define and bind client scripts to components**
Add and edit client scripts in User Interface (UI) Builder so that you can update the client state through events. You can bind these scripts to any component by using an event handler.

**Before you begin**
Role required: ui_builder_admin

**About this task**
You can create JavaScript client scripts in UI Builder by using the Now Code editor. Then, you can add the client script as an event handler to update the client state, emit a handled event of your page, or execute a data resource
operation. For example, you could write a script to increment a date or counter, and bind the script to a component event, like a button click. For more information about the Now Code Editor, see Now Code Editor.

With these scripts, you can do the following actions:

• Get available data, manipulate the data, and store it in the client state.
• Access data resource results.
• Execute data resource operations.
• Dispatch events.

Procedure

1. Navigate to Now experience framework > UI Builder.

2. Open an experience to work in or create a new experience by clicking Create experience in Platform.
   For more information, see Create an experience for UI Builder.

3. Open or create a page.
   For more information, see Work with pages.

4. Optional: If you do not have any components on your page, add a component to your page.
   For example, you can add a Heading component. For more information, see Add and configure components.

5. In the lower-left panel, click the Client Scripts icon, and then click the Add.
6. Write your script to perform an action.
   For example, you could write a simple date script that you bind to a component.

7. Name your script.
   A descriptive name helps you know what the script does. It also makes it easier to find the script later when you add it to an event handler. The following example shows a simple date client script.

8. Optional: Add a **Script include** or **Associated components**, which will show up in the **imports** parameter of your client script function.

9. Select the component on your page that you want to bind the script to and then click **Events**.

10. Click **+Add a new event handler**, select the script that you created under Scripts, and then click **Add**.
    The following example shows a date client script.
11. Click **Save**.

12. To preview your scripted component, click **Open**.

Create and bind a client state parameter to a component

Create and bind a client state parameter value to a component in User Interface (UI) Builder. By adding custom values to your components, these components can then be automatically updated through a script.

**Before you begin**
Role required: ui_builder_admin

**About this task**
You can add two components to a page and then connect them by using client state parameters and scripts. Start by creating a client state parameter and then binding the client state parameter to one of the components, such as a label. Next, you create a client script and bind it to a second component, such as a button, using an event handler. When you create the event handler, bind the client state parameter value to it to connect the two components. If you click one component, it changes the state of the other component. Client
states are useful because you can add custom values to your components that can be automatically updated through a script. Think of a client state as a bucket for storing information that is specific to the page.

For example, you can add a button and label component to your page. The button changes the value of the label, such as changing the text color.

**Procedure**

1. Navigate to **Now Experience Framework > UI Builder**.

2. Open an experience to work in or create a new experience by selecting **Create experience in Platform**.
   For more information, see **Create an experience**.

3. Create or open a page.
   For more information, see **Work with pages**.

4. Add two components to your page.
   For more information, see **Add and configure components**.

5. To define a client state parameter that has a value that is associated with it, click the Client state icon in the lower-left panel, and click +Add.

6. Enter a name for the client state, a type, and an initial value.
   The client state supports Strings, Numbers, Boolean, and JSON. For example, you could enter the name as `dynamic_label_value`, the type as `String`, and the initial value as `Initial Value`. 
7. Bind the value of the client state parameter to your component.
   a. Select the component you want to bind the client state parameter to.
   b. Click the Config panel and in the Value field, start typing @state and auto-complete lists the available client state parameters.
   c. Select the client state parameter you created from the list.
   d. In the ongoing dynamic_label_value example, you are using in this task, in the Value field, you would enter @state, and select @state.dynamic_label_value from the list.

8. Click Save, and select Open.

9. Bind the client state value to one component and create a script to connect the second component to the first.
   a. In the left pane, click the script icon.
   b. Click the +Add icon.
   c. Enter a script name that describes the task. For example, you could enter Update Label Value because that is what you want the script to do.
   d. In the Now Code Editor, add your script. Let’s say that you choose an API to call, and the parameters for the API, such as a state and a value. For example, you could call the api.setState, and include the
dynamic_label_value as the first parameter, and a NEW VALUE as the second parameter.

```javascript
/**
 * @param {params} params
 * @param {api} params.api
 * @param {any} params.event
 * @param {any} params.imports
 */
function handler({api, event, helpers, imports}) {
  api.setState('dynamic_label_value', 'NEW VALUE')
}
```

10. Add an event handler to your second component to call the new client script that you created.

   a. Select the component on your page that you want to bind the script to.

   b. Click the Events tab.

   c. Click + Add a new event handler.

   d. Under the Scripts section of event handlers, select the script that you created, and click the + Add icon to add it as an event handler for the component.

11. Save your page.
12. To preview your page and test the components to ensure they are connected, select Open.
When you select one component, it should change the state of the second component. For example, click the Button component to change the Label value component text from Initial Value to NEW VALUE.

Results
What you did in this procedure:

• Added two components to your page.
• Defined a client state parameter that had an initial value.
• Bound the client state parameter to the first component.
• Created a client script that updates a value that is defined by the client state parameter.
• Created an event handler for the second component to call the new client script that you just created.
• Selected the second component so that it changed the state of the first component.

Entity View Action Mapper for UI Builder
With User Interface (UI) Builder, you can use the Entity View Action Mapper (EVAM) application to standardize how the data sources in your components are displayed in your cards and lists.

EVAM Overview
Entity View Action Mapper (EVAM) is an application that standardizes how different data sources are displayed in cards and lists. UI Builder uses EVAM data sources to show information in a component as a card grid view or as a list of information. You can add a toggle to your component to let users switch between card grid and list views.

EVAM consists of the following components:

• Entity (data source). Associated data that you intend to display, such as a community post or a user.
• View. How a card displays data and actions.
• Actions. Action that it performs on the card. For example, you can activate a user into your system.
• Map. A process that maps the data source to generic fields that are displayed on the card. You can also associate actions that trigger from the card view.
**EVAM data sources**
You add EVAM data sources in UI Builder and bind them to a component.

### EVAM data resources

<table>
<thead>
<tr>
<th>Data resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamic Routing</td>
</tr>
<tr>
<td>EVAM Data Resource</td>
</tr>
<tr>
<td>Fetch EVAM Data</td>
</tr>
<tr>
<td>Fetch EVAM Metadata</td>
</tr>
<tr>
<td>Filtered Incidents</td>
</tr>
<tr>
<td>GlideRecord Collection Query</td>
</tr>
<tr>
<td>GlideRecord Query</td>
</tr>
<tr>
<td>Look Up Properties</td>
</tr>
<tr>
<td>Look Up Record</td>
</tr>
<tr>
<td>Look Up Records</td>
</tr>
<tr>
<td>Look Up User</td>
</tr>
<tr>
<td>Look Up User Preferences</td>
</tr>
<tr>
<td>Search EVAM Data Resource</td>
</tr>
<tr>
<td>Table route map</td>
</tr>
</tbody>
</table>

#### EVAM data sources

<table>
<thead>
<tr>
<th>EVAM data source</th>
<th>Description</th>
</tr>
</thead>
</table>
| EVAM Data Resource | Add an EVAM definition and other information about the data source.  
- **Type:** Composite.  
- **When to evaluate:** In the When to evaluate this data resource list, select **Immediately** to have the EVAM data resource instance evaluated on page load or select **Only when invoked** to use an event handler to evaluate the EVAM data resource.  
- **EVAM definition:** In the **EVAM Definition** field, add the EVAM definition record that is associated with the data resource.  
- **Page cursor:** In the **Page Cursor** field, add a page cursor for the EVAM definition. |
### EVAM data sources (continued)

<table>
<thead>
<tr>
<th>EVAM data source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Page size: In the <strong>Page Size</strong> field, add a size for the EVAM pagination.</td>
</tr>
<tr>
<td></td>
<td>• Page number: In the <strong>Page Number</strong> field, add the page number for the EVAM pagination.</td>
</tr>
<tr>
<td></td>
<td>• EVAM Filter Preference: In the <strong>EVAM Filter Preference</strong> field, enter the sys_ids for the EVAM data filter.</td>
</tr>
</tbody>
</table>

To add an event handler for when that data fetch is initiated, succeeded, or failed, click **Events**.

In the Now Code Editor section, which is next to the Configuration pane, preview the EVAM definition script.

### EVAM Data Resource configuration

![EVAM Data Resource configuration](image)

**Fetch EVAM Data**

To add an EVAM definition and other information about the data source, select **Configuration**.

• **Type**: GRAPHQL.

• **When to evaluate**: To have the EVAM data resource instance evaluated on a page load, select **Immediately**. o use an event
EVAM data sources (continued)

<table>
<thead>
<tr>
<th>EVAM data source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>handler to evaluate the EVAM data resource, select <strong>Only when invoked</strong>.</td>
</tr>
<tr>
<td></td>
<td>• EVAM definition: In the <strong>EVAM Definition</strong> field, enter the EVAM definition record that is associated with the data resource.</td>
</tr>
<tr>
<td></td>
<td>• Page cursor: In the <strong>Page Cursor</strong> field, enter a page cursor for the EVAM definition.</td>
</tr>
<tr>
<td></td>
<td>• Page size: In the <strong>Page Size</strong> field, enter a size for the EVAM pagination.</td>
</tr>
<tr>
<td></td>
<td>• Page number: In the <strong>Page Number</strong> field, enter the page number for the EVAM pagination.</td>
</tr>
<tr>
<td></td>
<td>• EVAM Filter Preference: In the <strong>EVAM Filter Preference</strong> field, enter the sys_ids for the EVAM data filter.</td>
</tr>
</tbody>
</table>

To add an event handler for when that data fetch is initiated, succeeded, or failed, select **Events**.

Preview the EVAM definition script in the **Now Code Editor** to the right of the Configuration pane.
EVAM data sources (continued)

<table>
<thead>
<tr>
<th>EVAM data source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fetch EVAM Metadata</td>
<td>To select when to evaluate the EVAM data resource and add an EVAM definition, select <strong>Configuration</strong>.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Type</strong>: GRAPHQL.</td>
</tr>
<tr>
<td></td>
<td>- <strong>When to evaluate</strong>: To have the EVAM data resource instance evaluated on a page load, select <strong>Immediately</strong>. To use an event handler to evaluate the EVAM data resource, select <strong>Only when invoked</strong>.</td>
</tr>
<tr>
<td></td>
<td>- <strong>EVAM definition</strong>: In the <strong>EVAM Definition</strong> field, enter the EVAM definition record that is associated with the data resource.</td>
</tr>
<tr>
<td></td>
<td>To add an event handler for when that data fetch is initiated, succeeded, or failed, select <strong>Events</strong>.</td>
</tr>
<tr>
<td>EVAM data source</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Preview the EVAM definition script in the <strong>Now Code Editor</strong> to the right of the Configuration pane.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search EVAM Data Resource</th>
<th>To add an EVAM definition, and other information about the data source, select <strong>Configuration</strong>.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• <strong>Type:</strong> Composite. <strong>When to evaluate:</strong> To have the EVAM data resource instance evaluated on a page load, select <strong>Immediately</strong>. To use an event handler to evaluate the EVAM data resource, select <strong>Only when invoked</strong>.</td>
</tr>
<tr>
<td></td>
<td>• <strong>EVAM Config ID:</strong> In the <strong>EVAM Config ID</strong> field, add the <strong>EVAM Config ID</strong> that is associated with the data record.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Search Context Config ID:</strong> In the <strong>Search Context Config ID</strong> field, add</td>
</tr>
<tr>
<td>EVAM data source</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>the Search Context Config ID that you are searching for.</td>
</tr>
<tr>
<td></td>
<td>• Search Term: In the <strong>Search Term</strong> field, add the search terms that you are searching for.</td>
</tr>
<tr>
<td></td>
<td>• Facet Filters: In the <strong>Facet filters</strong> field, add any facet filters to help define your search.</td>
</tr>
<tr>
<td></td>
<td>• Search Filters: In the <strong>Search Filters</strong> field, add any search filters to help define your search.</td>
</tr>
<tr>
<td></td>
<td>• Disable Spell Checking: In the <strong>Disable Spell Checking</strong> field, select if you desire spell checking to be on or off during your search.</td>
</tr>
<tr>
<td></td>
<td>• Pagination Token: In the <strong>Pagination Token</strong> field, enter a pagination token if needed.</td>
</tr>
</tbody>
</table>

To add an event handler for when that data fetch is initiated, succeeded, or failed, select **Events**.

Preview the EVAM definition script in the **Now Code Editor** to the right of the Configuration pane.
## Add an Entity View Action Mapper data resource to a page

Add an Entity View Action Mapper (EVAM) data resource to your page in Now® Experience UI Builder so that you can standardize how the data sources in your components are displayed in your cards and lists.

### Before you begin

Role required: ui_builder_admin

### Procedure

1. Navigate to **Now experience framework** > **UI Builder**.
2. Open an experience to work in or create a new experience by selecting **Create experience in Platform**. For more information, see **Create an experience**.
3. Open or create a page. For more information, see **Work with pages**.
4. Optional: If you do not have any components on your page, add a **Data set** component to your page. For more information, see **Add and configure components**.

### EVAM data sources (continued)

<table>
<thead>
<tr>
<th>EVAM data source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add an Entity View Action Mapper data resource to a page</td>
<td>Add an Entity View Action Mapper (EVAM) data resource to your page in Now® Experience UI Builder so that you can standardize how the data sources in your components are displayed in your cards and lists.</td>
</tr>
<tr>
<td><strong>Before you begin</strong></td>
<td>Role required: ui_builder_admin</td>
</tr>
<tr>
<td><strong>Procedure</strong></td>
<td></td>
</tr>
<tr>
<td>1. Navigate to <strong>Now experience framework</strong> &gt; <strong>UI Builder</strong>.</td>
<td></td>
</tr>
<tr>
<td>2. Open an experience to work in or create a new experience by selecting <strong>Create experience in Platform</strong>. For more information, see <strong>Create an experience</strong>.</td>
<td></td>
</tr>
<tr>
<td>3. Open or create a page. For more information, see <strong>Work with pages</strong>.</td>
<td></td>
</tr>
<tr>
<td>4. Optional: If you do not have any components on your page, add a <strong>Data set</strong> component to your page. For more information, see <strong>Add and configure components</strong>.</td>
<td></td>
</tr>
</tbody>
</table>
5. In the left panel, click the data icon and then click + Add.

6. Select an application that has the data resources that you want to choose from.
   The EVAM data resources are in the Global application.

7. Under Data Resources, select the EVAM data resource that you want to add and repeat this step for all the EVAM data resources for your page.
8. Configure each EVAM data resource.

<table>
<thead>
<tr>
<th>EVAM data source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVAM Data Resource</td>
<td>Add an EVAM definition and other information about the data source.</td>
</tr>
<tr>
<td></td>
<td>• Type: Composite.</td>
</tr>
<tr>
<td></td>
<td>• When to evaluate: In the When to evaluate this data resource list, select <strong>Immediately</strong> to have the EVAM data resource instance evaluated on page load or select <strong>Only when invoked</strong> to use an event handler to evaluate the EVAM data resource.</td>
</tr>
<tr>
<td></td>
<td>• EVAM definition: In the <strong>EVAM Definition</strong> field, add the EVAM definition record that is associated with the data resource.</td>
</tr>
<tr>
<td></td>
<td>• Page cursor: In the <strong>Page Cursor</strong> field, add a page cursor for the EVAM definition.</td>
</tr>
<tr>
<td></td>
<td>• Page size: In the <strong>Page Size</strong> field, add a size for the EVAM pagination.</td>
</tr>
<tr>
<td></td>
<td>• Page number: In the <strong>Page Number</strong> field, add the page number for the EVAM pagination.</td>
</tr>
<tr>
<td></td>
<td>• EVAM Filter Preference: In the <strong>EVAM Filter Preference</strong> field, add the sys_ids for the EVAM data filter.</td>
</tr>
</tbody>
</table>

To add an event handler for when that data fetch is initiated, succeeded, or failed, click **Events**. In the Now Code Editor section, which is next to the Configuration pane, preview the EVAM definition script.
<table>
<thead>
<tr>
<th>EVAM data source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Fetch EVAM Data</strong></td>
</tr>
<tr>
<td></td>
<td>To add an EVAM definition and other information about the data source, select <strong>Configuration</strong>.</td>
</tr>
<tr>
<td></td>
<td>• Type: GRAPHQL.</td>
</tr>
<tr>
<td></td>
<td>• When to evaluate: To have the EVAM data resource instance evaluated on a page load, select <strong>Immediately</strong>. To use an event handler to evaluate the EVAM data resource, select <strong>Only when invoked</strong>.</td>
</tr>
<tr>
<td></td>
<td>• EVAM definition: In the <strong>EVAM Definition</strong> field, add the EVAM definition record that is associated with the data resource.</td>
</tr>
<tr>
<td></td>
<td>• Page cursor: In the <strong>Page Cursor</strong> field, add a page cursor for the EVAM definition.</td>
</tr>
<tr>
<td></td>
<td>• Page size: In the <strong>Page Size</strong> field, add a size for the EVAM pagination.</td>
</tr>
<tr>
<td>EVAM data source</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>• Page number: In the <strong>Page Number</strong> field, add the page number for the EVAM pagination.</td>
</tr>
<tr>
<td></td>
<td>• EVAM Filter Preference: In the <strong>EVAM Filter Preference</strong> field, add the sys_ids for the EVAM data filter.</td>
</tr>
</tbody>
</table>

To add an event handler for when that data fetch is initiated, succeeded, or failed, select **Events**.

Preview the EVAM definition script in the **Now Code Editor** to the right of the Configuration pane.

---

**Fetch EVAM Metadata**

To select when to evaluate the EVAM data resource and add an EVAM definition, select **Configuration**.

• **Type**: GRAPHQL.

• **When to evaluate**: To have the EVAM data resource instance evaluated on a page load, select **Immediately**. To use an event...
<table>
<thead>
<tr>
<th>EVAM data source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>handler to evaluate the EVAM data resource, select <strong>Only when invoked</strong>.</td>
<td></td>
</tr>
<tr>
<td><strong>•</strong> EVAM definition: In the <strong>EVAM Definition</strong> field, add the EVAM definition record that is associated with the data resource.</td>
<td></td>
</tr>
<tr>
<td>To add an event handler for when that data fetch is initiated, succeeded, or failed, select <strong>Events</strong>.</td>
<td></td>
</tr>
<tr>
<td>Preview the EVAM definition script in the <strong>Now Code Editor</strong> to the right of the Configuration pane.</td>
<td></td>
</tr>
</tbody>
</table>

Search EVAM Data Resource

To add an EVAM definition and other information about the data source, select **Configuration**.

- **Type**: Composite.
- **When to evaluate**: To have the EVAM data resource instance evaluated on a page load, select **Immediately**. To use an event handler to evaluate the EVAM...
<table>
<thead>
<tr>
<th>EVAM data source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>data resource, select <strong>Only when invoked.</strong></td>
</tr>
<tr>
<td></td>
<td>• EVAM Config ID: In the <strong>EVAM Config ID</strong> field, add the <strong>EVAM Config ID</strong> that is associated with the data record.</td>
</tr>
<tr>
<td></td>
<td>• Search Context Config ID: In the <strong>Search Context Config ID</strong> field, add the Search Context Config ID that you are searching for.</td>
</tr>
<tr>
<td></td>
<td>• Search Term: In the <strong>Search Term</strong> field, add the search terms that you are searching for.</td>
</tr>
<tr>
<td></td>
<td>• Facet Filters: In the <strong>Facet filters</strong> field, add any facet filters to help define your search.</td>
</tr>
<tr>
<td></td>
<td>• Search Filters: In the <strong>Search Filters</strong> field, add any search filters to help define your search.</td>
</tr>
<tr>
<td></td>
<td>• Disable Spell Checking: In the <strong>Disable Spell Checking</strong> field, select if you desire spell checking to be on or off during your search.</td>
</tr>
<tr>
<td></td>
<td>• Pagination Token: In the <strong>Pagination Token</strong> field, add a pagination token if needed.</td>
</tr>
</tbody>
</table>

To add an event handler for when that data fetch is initiated, succeeded, or failed, select **Events**.

Preview the EVAM definition script in the **Now Code Editor** to the right of the Configuration pane.
9. Configure your data set component:
   
   a. Bind an EVAM data resource to your component.
   
   b. Set the Show grid/list toggle to on to let users choose between the grid or list view of the EVAM information on the page.
   
   c. Set the other configuration settings for the cards on the page.

10. Click Save.

11. To preview your component with the EVAM data resources, click Open.
Finished state of the EVAM

What to do next
For more information, see the following EVAM topics:

- Entity View Action Mapper
- Getting started: Create a multi-data source list display in Entity View Action Mapper
- Create an EVAM action definition
- Create an EVAM definition

Events in UI Builder
Learn how to work with events so that you can add actions to components, pages, data resources, and declarative actions in User Interface (UI) Builder.

Event mapping
An event mapping is the process that enables you to map an event’s payload or contextual values to the object or handler that acts on that event.

For more information about event mapping, see Event mapping.

Events
Use an event to add actions to your components, pages, and data resources.
• A component event is an action that you set up for a component. You set up an event handler to configure that component action. For example, you can add a button component to your page. Then, you can add an event handler to apply an action for that button, such as going to a web page.

• Page events perform actions for the entire page. You can configure the following page events:
  ◦ Page event mappings. Add, remove, or clear alert notifications on your page.
  ◦ Variant event mappings. Add, remove, or clear alert notifications on your page variant.
  ◦ Dispatched events. Create dispatched events for your page to create relayed event mappings that model events after a parent event handler. Select a target parent event handler to model the payload fields after it.
  ◦ Handled events. A handled event is an event that is exposed and available for use by other users. After you create a handled event, it is available under Page event mappings for other users to use. You can also set up payload fields that you create manually or choose a template for your handled event, such as an open or close a modal dialog.

• Data resources events map data resources to notify information about when data is fetched.

Event handlers

Use an event handler to configure an action for your page or the components on the page. For example, you can map an event to your page to add an alert notification when the page successfully loads. The alert notification shows as a modal on the page that asks a user to verify that the user wants to delete the record.

When you add an event handler to a page or component, you can choose different types of event handlers. For example, a Button component can have the following types of event handlers:

• Inherited event handlers. An inherited event handler is exposed from the page that you are working in. If you are in the parent page, an inherited event handler could be exposed from the app shell. The following table lists the different types of inherited event handlers that you can use and what you can do with them.
Inherited event handlers

<table>
<thead>
<tr>
<th>Event handler</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link to destination</td>
<td>Navigate to a destination.</td>
</tr>
<tr>
<td>◦ App routes: Link to another page within an app, like a home screen.</td>
<td></td>
</tr>
<tr>
<td>◦ External URL: Link to a website or any external URL.</td>
<td></td>
</tr>
<tr>
<td>Add parameters to URL</td>
<td>Add additional parameters to a URL.</td>
</tr>
<tr>
<td>Open or close modal dialog</td>
<td>After you create a modal, use Open or close modal dialog to trigger the modal.</td>
</tr>
</tbody>
</table>

Page-level event handlers. This type of event handler is common to all pages, and you would use this handler type when you want to add or clear page-level alert notifications. The following table lists the different types of page-level event handlers that you can use and what you can do with them.

<table>
<thead>
<tr>
<th>Event handler</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add alert notifications</td>
<td>Add a code snippet to send an alert notification. For example:</td>
</tr>
<tr>
<td></td>
<td>[{&quot;content&quot;: &quot;hello world&quot;}]</td>
</tr>
<tr>
<td>Remove alert notification</td>
<td>Add code to call alert notification IDs that you want to dismiss. For example, click a button to remove a page load alert notification.</td>
</tr>
<tr>
<td>Clear alert notification</td>
<td>Add code to call all alert notification IDs that you want to dismiss. For example, click a button to remove all alert notifications.</td>
</tr>
<tr>
<td>Set loading state</td>
<td>Toggle loading on or off. For example, you can toggle loading on to load the page when you click a button or toggle loading off to not load the page when a button is clicked.</td>
</tr>
</tbody>
</table>
### Event handler | Description
--- | ---
Update client state parameter | Declaratively set the client state parameter. Let's say that you have a client state parameter that you set up with a value. You can configure the **Update client state parameter** event handler to update the client state parameter with a new value. For example, you have a client state parameter that is called **Greeting** that is set up with **Hello** as the initial value. You can add an **Update client state parameter** event handler select the **Greeting** client state parameter, and then enter a new value like **Goodbye**. When you click the button, **Goodbye** replaces **Hello** on the page.

- Data resource handlers. This type of event handler triggers the fetching or writing of data to the server. You can refresh the app shell data source data on your page by clicking a button. For example, with a data resource handler, you can do the following actions:
  - Bind data to the description of an incident record.
  - Change the value of the incident description.
  - Add a button component to your page.
  - Label the button as **Refresh incident**.
  - Add a **Look Up Record** event handler for the button.
  - Save your page.
  - If the description of the incident record changes, click **Refresh** to update the description on your page.

- Client scripts. Scripts that execute when an event is triggered on a component. You create these scripts in the Client scripts section in UI Builder. For more information, see **Define client scripts**.

#### Binding events to components

Bind event handlers to the events on a component. When you add components to your page, these components are not configured to perform any action on your page. For example, a button component is static and does not do
anything until you bind an event action to it, such as deleting a record. Some components have several events where event handlers can be assigned. For example, on the list component, you can assign a navigation handler to the Row clicked event. You can also assign an open modal to the Data fetch failed event. For more information, see Bind event to a component.

**Binding events to pages**

Bind a page-level event to execute event handlers on the page. For example, use page-level events for page notifications, page load, or when a page property changes. The assignment of the event handler to the page-level event is similar to assigning handlers to events from components.

You can bind event mappings using the following methods:

- Page event mappings. Add, remove, or clear alert notifications on your page.
- Variant event mappings. Add, remove, or clear alert notifications on your page variant.
- Dispatched events. Create dispatched events for your page to create relayed event mappings that model events after a parent event handler. Select a target parent event handler to model the payload fields after it.
- Handled events. Add a handled event for an event that is exposed and available for use by other users.

For more information, see Bind event to page.

**Binding events to data resources**

Bind event handlers to individual data resources on your page. For example, when a data resource successfully fetches new data, it executes an event handler, like navigation, to take a user to the next step in a flow. When a data resource successfully adds a record to a table, it shows a success modal that uses the show modal event handler. For more information, see Bind event to data resources.

**Binding events to declarative actions**

Bind data elements to add event actions to a declarative action definition in Actions & Components in the Now Platform®. For example, you could bind a data element to add an event action to complete work on a table.

If you add a component to your page that has a declarative action, you must bind it to a handled event. The handled event creates an action that is performed when a user selects the component. By selecting Configure declarative action event mapping, you add a new event handler to define what the declarative action does on the page.
Event mapping

An event mapping in User Interface (UI) Builder is the process that enables you to map an event's payload or contextual values to the object or handler that acts on that event. The four event types are component, page, data resource, and declarative action.

Event mapping is an important process within UI Builder. When you build pages with components, you need those components to perform actions for users. For example, if you add a button component to the page, a button-clicked event must be mapped to an event handler. The event handler performs a button-clicked action when it is selected by a user. An example is when you add a data resource, such as a form, and have an event handler notify the user when the form successfully loads.

Event types

The event types that are available are based on the component. For example, declarative action events are available for specific components, such as the Action bar or List components.

You choose a type of event based on what action you want to perform on your page. For example, if you want to bind an action to a component, such as a button loading a web page, you would use a component event. If you want an event to apply to your whole page, such as adding an alert notification to a page, you use a page event. The following table describes each event type
that is available in UI Builder and provides some examples on how you can use the events.

### Event types and descriptions

<table>
<thead>
<tr>
<th>Event type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component events</td>
<td>Action that you set up for a component. You set up an event handler to configure that component action. The following example shows how to add an event handler to apply an action for a button, such as going to a web page. For more information on binding events to components, see <a href="#">Bind an event to a component</a>.</td>
</tr>
<tr>
<td>Page events</td>
<td>Page event that performs an action for the entire page. You can configure the following page events:</td>
</tr>
<tr>
<td></td>
<td>• Page event mappings.</td>
</tr>
<tr>
<td></td>
<td>◦ These event mappings are saved on the page definition record, which can be found in the [sys_ux_macroponent] table.</td>
</tr>
<tr>
<td></td>
<td>◦ The source events for these event mappings are Page ready and Page property. They are defined for your page.</td>
</tr>
<tr>
<td></td>
<td>◦ The available handlers for these event mappings are as follows:</td>
</tr>
<tr>
<td></td>
<td>■ User session events</td>
</tr>
<tr>
<td></td>
<td>■ Page-level events</td>
</tr>
<tr>
<td></td>
<td>■ UXR App Shell Data Source</td>
</tr>
<tr>
<td></td>
<td>■ Client scripts that are defined on the page</td>
</tr>
<tr>
<td></td>
<td>■ Operations from local and inherited data resources</td>
</tr>
<tr>
<td></td>
<td>• Variant event mappings.</td>
</tr>
<tr>
<td>Event type</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>◦ These event mappings are saved on the variant record, which you can find in the [sys_ux_screen] table.</td>
<td></td>
</tr>
<tr>
<td>◦ The source events for these event mappings are any dispatched events that are defined for your page.</td>
<td></td>
</tr>
<tr>
<td>◦ The available handlers for these event mappings are canvas-level events, the UXR App Shell Data Source, and operations from the inherited data resources.</td>
<td></td>
</tr>
<tr>
<td>◦ Variant event mappings are used as relays so that the events that are dispatched from components on your page can be relayed up to user session event handlers.</td>
<td></td>
</tr>
<tr>
<td>◦ Mappings are created automatically on a page save when you have mapped a component's dispatched event to a user session handler. For example, by mapping a <strong>Button clicked</strong> event to the canvas-level <strong>Link to destination</strong> event, the event creates a dispatched relay event and a variant event mapping when the page is saved.</td>
<td></td>
</tr>
</tbody>
</table>

• Dispatched events.

| ◦ These events are saved in the [sys_ux_event] table. The page definition record contains references to the [sys_ux_event] record. |
## Event types and descriptions (continued)

<table>
<thead>
<tr>
<th>Event type</th>
<th>Description</th>
</tr>
</thead>
</table>
| ◦ These events serve as source events for variant event mappings.  
◦ You can select **+Add** to create new dispatched events for your page. Configure the event label to auto-populate the event name and payload fields for your new `sys_ux_event`.  
◦ These events are used to create relay event mappings. You can select a **Target parent event handler** to model the payload fields after the selected event. You can also manually create payload fields for your dispatched event. |  
• Handled events.  
◦ These events are saved in the `[sys_ux_event]` table. The page definition record contains references to the `[sys_ux_event]` record.  
◦ These events serve as source events for page event mappings.  
◦ You can select **+Add** to create new dispatched events for your page. Handled events are not modeled after parent event handlers. Payload fields for handled events are created manually. To use an existing handler's payload fields as a template, select a template and edit the fields as necessary. |
## Event types and descriptions (continued)

<table>
<thead>
<tr>
<th>Event type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data resource events</td>
<td>Events that are mapped to data resources to provide notifications about when data is fetched.</td>
</tr>
<tr>
<td></td>
<td>• Data Fetch Initiated. When a data resource event is triggered, the event handler executes the data fetch process.</td>
</tr>
<tr>
<td></td>
<td>• Data Fetch Succeeded. When a data resource event is triggered, the event handler executes the process</td>
</tr>
</tbody>
</table>

For more information on binding an event to a page, see [Bind event to page](#).
<table>
<thead>
<tr>
<th>Event type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Fetch Initiated</td>
<td>to notify a user when the data fetch completed successfully.</td>
</tr>
<tr>
<td>Data Fetch Succeeded</td>
<td>• Data Fetch Failed. When a data resource event is triggered, the event handler executes the process to notify a user if the data fetch was unsuccessful.</td>
</tr>
</tbody>
</table>

For more information on binding an event to a data resource, see Bind event to data resource.

### Declarative action events

Bind data elements within UI Builder to add event actions to a declarative action.

You configure a declarative action event mapping in the Now Platform® declarative action assignment table. For an example, navigate to Filter navigator > sys_declarative_action_assignment.list and then search for and open an existing declarative action.

In UI Builder, you bind an event to the declarative action. For more information on how to use declarative event actions, see Bind event to data resource.
### Event types and descriptions (continued)

<table>
<thead>
<tr>
<th>Event type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>action events, see Bind event to declarative action.</td>
<td>action events, see Bind event to declarative action.</td>
</tr>
</tbody>
</table>
a. Select the component to highlight it, either in the content tree or on the page itself.

b. From the Content section, click +Add.

c. Select Modal, and choose a modal type, such as Confirm.

d. Configure the modal. You can change the text in the modal, the names of the buttons in the modal, and the size of the modal screen. You can also set actions for the modal. When you finish configuring the modal, close it. Notice the modal that you created sits above the body of your page structure in the Content tree.

6. To add an event handler to your component’s event, go to the Configuration pane and select Events. An event handler lets you assign an event to a component. For example, if you add a button component to your page, you want it to perform an action when a user clicks it.

7. To start the process of assigning the event to the component, choose an event that is associated with a component, and then click Add a new event handler. For example, a button component only has the Button clicked event that is associated with it. Other components can have many events.
8. From the Event handler preview screen, select an action that you want assigned to the component. For example, to add an event handler for the button component, you could link it to another destination.

9. Click the Open or close modal dialog event handler.
   a. Click the Confirm modal dialog that you created earlier.
   b. Select the Open modal dialog selector so that the modal opens when a user clicks the button component.
   c. To add the modal event handler to your component, click Add.

10. Click Save.

### Bind an event to a component

Bind data elements within User Interface (UI) Builder so that you can add event actions to your components. Each component has specific events that it binds to. For example, a button component has only a button-clicked event, while other components can have multiple events that are associated with them.

#### Before you begin
Role required: ui_builder_admin

#### About this task
Some components do not have an event action that is applied to them. An example is the heading component. Many components require that you map an event to your component to make it perform an action, such as loading data.

To add actions to components, pages, and data resources on your page, you can add an event handler. A button component that you add to a page is static. By binding an event action to the button, you can link it to a web page.

#### Procedure
1. Navigate to Now Experience Framework > UI Builder.
2. Open an experience to work in or create a new experience by selecting Create experience in Platform.
For more information, see Create an experience.

3. Create or open a page.

4. Add a component to your page, such as a button.

5. To add an event handler to your component, select the Events tab.
   For more information on how to add event handlers to your component, see Work with events.

a. To start the process of setting up an event handler for your component, click + Add a new event handler.
   The following example shows how to add an event handler for a button component.

   ![Events tab preview]

   b. From the Event handler preview section, select an event handler to bind to your component.

   ![Event handler preview]
c. Configure the event handler and click **Add** when finished.
   You configure each event handler differently, depending on the action of the event. For example, if you add an event handler for a button component, you can choose what that button action performs.

6. **Optional:** If you want a modal to pop up for your event, add the modal to the component before you bind your event to the component.
   A modal is a confirmation pop-up that appears when you click the component. For example, if you add a button component that deletes a record, you add a modal to ask the user to confirm that the user wants to delete the record. For more information, see **Add modal to component**.

a. Select the component to highlight it, either in the content tree or on the page.

b. From the content section, click **+ Add**.

c. Click **Modal**, and choose a modal type, such as Confirm and destroy.

![Modal types](image)

d. Configure the modal.
   You can change the text in the modal, the names of the buttons in the modal, and the size of the modal screen. You can also set actions for the modal. When you finish configuring the modal, close it. Notice that the modal you created sits above the body of your page structure in the content tree.

7. **Bind an event to your component.**
   You bind an event to the button to trigger an action. Select the button component to highlight it again, either in the content tree, or on the page.

8. In the Configuration pane, click **Events**.

9. To start the process of binding an event to the button component, select **Add a new event handler**.
The button component only has the button-clicked event associated with it. Other components can have more than one event.

10. Select an action that you want assigned to the component. For example, to add an event handler for the button component, you could link it to another destination and add alert notifications. In this example, you can choose **Open or close modal dialog** to make the button open the modal that you created earlier.

![Modal Dialog Configuration](image)

11. In this button scenario, select the Confirm and destroy modal dialog event handler that you created earlier.

   a. Turn on the **Open modal dialog** selector so that the modal opens when a user clicks the button component.

   b. To add the modal event handler to your component, click **Add**.

12. Click **Save**.

13. To preview your work, click **Open**.

![Modal Dialog Preview](image)

14. To trigger the confirmation modal that you created, click the button on the page.

15. Click **Save**.

16. To preview your page and test the component event, click **Open**.

![Modal Dialog Preview](image)

**Link an event to another page**

Add a link to the destination event handler within User Interface (UI) Builder so that an event action can open another page. You can also configure the event handler to follow the App Route to the desired page.
Before you begin
You must have a workspace page that contains a component that is intended
to open another page when a user clicks it. The dashboard overview is an
example of such a component.
Role required: ui_builder_admin

About this task
To configure an event action to open another page, you must know what page
you want to open, the required and optional parameters are for that page,
and what payload values to set on the event handler to pass the required
parameters to the destination page.

Tip: You may be able to find examples of both the components that you
want to link from and the destination pages that you want to link to in the
Base Agent Workspace Experience. This Now® Experience is provided in
the base system. If you create a page from a page template, you should
only copy the contents of the template. Do not reference it. For more
information about the difference between copying and referencing a
page template, see Page templates.

Procedure
1. Open your experience in Now® Experience UI Builder.

2. Optional: If the destination page doesn’t exist in your experience, create one.
For information about creating pages, see Create a page in UI Builder. Make
sure that you set the required and optional parameters for the page so that
you can use it as a destination. If a particular component in the page is a
destination, you need to include that component. You also need to configure
the properties on the component to consume the page parameters with
@context.props.<parameter-name> values.

You might consider creating the page from a page template. The Base
Agent Workspace Experience has several page templates that are already
configured to be destinations for other components. If you create a
destination page from a template, the components are already configured
with the correct properties. Any necessary state parameters or client scripts
are also copied over. You have to add the page parameters. You can copy
these parameters from the UX App Routes related list on the Agent app
config [sys_ux_app_config] record of the experience that contains the page
templates.

To make sure that the pages that you are creating work reliably as
destinations in your experience, your experience must have the same app
shell UI as the experience with the page templates.

3. Switch to the page that you want to link to the destination page.
4. Navigate to the relevant component and select it.

5. Click the Events tab.

6. Expand an event action that should open another component and click + Add a new event handler.

7. In the Inherited event handlers section, Click **Link to destination**.

   Event handler preview: On, Report Visualization Clicked → Link to destination (UXR Blank AppShell) → Enter values below...

   ![Event handler preview image]

   - **Search**
   - **Inherited event handlers**
     - **Link to destination**
     - Add parameters to URL
     - Open or close modal dialog
   - **Page-level event handlers**
     - Add alert notifications
     - Remove alert notifications
     - Clear alert notifications
     - Set loading state
     - Update client state parameter
   - **UXR App Shell Data Source 1**
     - Refresh
   - **Canvas User Session Data Resource GraphQL**

   ![Select destination dialog]

   **Select destination**
   - Remove previous URL from browser history
   - Load in background (advanced)
   - Title (advanced)
   - Multi-instance field (advanced)
   - Sub-navigation route (advanced)

8. Click **Select destination**.

9. Expand **App routes** and select the route to the page in the experience that you want to link to.

   Fields appear for each of the parameters on the destination page that the route leads to. Required parameters are marked with an asterisk (*).
10. Complete each required parameter field and the applicable optional parameter fields with an appropriate `@payload.*` value.
If the developers of your component included default payload values in your event, you can select one through autocompletion. As shown in the following example, the payload value may not match the parameter name.
If no default values are provided, or you can't determine which values are correct for some fields, refer to the configuration and API documentation for the component in the ServiceNow® Developer Site. If you still can't find the necessary @payload.* values, contact Customer Service and Support.

Tip: If you create your linking component by creating a page from a Base Agent Workspace page template, the component contains Link to destination Relay event handlers. These event handlers do not work. However, they contain the applicable @payload.* values for the parameters.

Example: Configuring the event handlers for an Analytics Q&A component

Let's say that you want to take a new Now® Experience and add a page with an Analytics Q&A component. First, you create the page from the Analytics Center page template that is provided in the Base Agent Workspace experience. Next, you create a target page for the first of the three events in Analytics Q&A and then you configure an event handler for that event.

By navigating to Now Experiences Framework > Experiences, you see the Test experience UX application. Because it uses the same Agent Workspace App.
Shell UI as the Base Agent Workspace, you can use the page templates from the Base Agent Workspace.

You next select the Test workspace admin panel, find an UX App Configuration record with no UX app routes or pages, and then click **Open**.

As the example shows, in the UI Builder, you have created a page named Analytics Center that is based on the Analytics Center page template from the Base Agent Workspace. Next, you select the option to copy only the contents of the page template.
You select the Analytics Q&A 1 component and open the Events tab. From here, you can open the Link to destination Relay event handler for the Report Visualization Clicked event. When a question in Analytics Q&A returns a report, you can trigger this event by clicking a value in the report. When you click a value, you also see a list of the records that contribute to this value. In the Route field, you see that the destination is expected to be a page that is based on the Simple List page template. You also see the parameters of the page that the @payload.* values correspond to, and that the Title field can be populated with @payload.listTitle.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>@payload.* value</th>
</tr>
</thead>
<tbody>
<tr>
<td>table (required)</td>
<td>@payload.table</td>
</tr>
<tr>
<td>listTitle</td>
<td>@payload.listTitle</td>
</tr>
<tr>
<td>query</td>
<td>@payload.query</td>
</tr>
<tr>
<td>disableInlineEditing</td>
<td>none</td>
</tr>
</tbody>
</table>
Next, you navigate to **Menu > Create page** and create a page that is based on the Simple List template. Let's say that you name the page as Record list. You then follow a similar process as when you created the Analytics Center page. This time in the last steps of the process, you would add **table** as a required parameter and **listTitle**, **query**, and **disableInlineEditing** as optional parameters.
Because this page already contains a List component, when you open the Config tab for this component, you see that the parameters are already passed in the `@context.props.*` values.
Now, you return to the Analytics Center page. In the Report Visualization Clicked event, you add a new event handler. Next, you select the Record list page that you created and add the `@payload.*` values in the `table`, `listTitle`, and `query` fields, following the information that you got from the Link to destination Relay event handler. Predictive typing helps you to fill in these fields.
After you click **OK** and add `@payload.listTitle` as the **Title**, the event handler is done. You now delete the Link to the destination Relay event handler for this event.

The following example shows an Analytics Center page. On this page, you can enter a query for Incidents by Priority and get a report as a result. Also, by clicking a column, you trigger a Report Visualization Clicked event. The event handler enables you to see a simple list of the incidents in the report.
Before you begin
Role required: ui_builder_admin

About this task
You can bind an event to a page by using the following types of events:
• Page event mappings. Add, remove, or clear alert notifications on your page.
• Variant event mappings. Add, remove, or clear alert notifications on your page variant.
• Dispatched events. Create dispatched events for your page to create relayed event mappings that model events after a parent event handler. Select a target parent event handler to model the payload fields after it.

• Handled events. A handled event is an event that is exposed and available for use by other users. After you create a handled event, it is available under **Page event mappings** for other users to use. You can also set up payload fields that you create manually or choose a template for your handled event, such as an open or close a modal dialog.

**Procedure**

1. Navigate to **Now Experience Framework > UI Builder**.
2. Open an experience to work in or create a new experience by selecting **Create experience in Platform**. See **Create an experience** for more information.

3. Open or create a page.

4. Highlight the page body in the content tree. The body is the top-level line of the content tree. When you highlight the whole page, you can add page-level events.

5. In the configuration panel, click the **Events** tab.

6. Add an event handler in one of the following ways:

<table>
<thead>
<tr>
<th>To add this event handler</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Page ready</strong> source event:</td>
<td><strong>Page ready</strong> source event:</td>
</tr>
<tr>
<td>a. Click <strong>Add a new event handler</strong>.</td>
<td><strong>Page ready</strong> source event:</td>
</tr>
<tr>
<td>b. To add, remove, or clear alert notifications from the page, choose an event handler from <strong>Page-level event handlers</strong>. You can add, re-</td>
<td><strong>Page ready</strong> source event:</td>
</tr>
</tbody>
</table>
### To add this event handler

<table>
<thead>
<tr>
<th>Do this</th>
<th>move, or clear alert notifications on your page.</th>
</tr>
</thead>
<tbody>
<tr>
<td>c.</td>
<td>To add the event handler to your page, click <strong>Add</strong>.</td>
</tr>
</tbody>
</table>

**Page property changed** source event:

| a. | Click **Add a new event handler**. |
| b. | Choose an event handler from **Page-level event handlers**. You can add, remove, or clear alert notifications on your page. You can also choose **Set loading state** to show the word **Loading** on your page so users can see that the page is loading. You can also select **Update client state parameter** to change the client state name. |
| c. | To add the event handler to your page, click **Add**. |

### Variant event mappings

<table>
<thead>
<tr>
<th>Variant event mappings</th>
<th>You must have a variant page to perform a variant event mapping. The Variant event mappings that are available depend on the variant page. If the variant page has a list, you add event handlers for things like Row clicked, Nav item, and so on. When the variant has a Viewport, you may have different event handlers to set up.</th>
</tr>
</thead>
<tbody>
<tr>
<td>a.</td>
<td>Select <strong>Variant event mappings</strong>.</td>
</tr>
<tr>
<td>b.</td>
<td>Click <strong>Add a new event handler</strong>. For example, you can add a new event handler for <strong>Row clicked</strong>.</td>
</tr>
</tbody>
</table>
| c. | Select an event handler from the list of available Inherited event han-
To add this event handler | Do this
--- | ---
| d. To add the event handler to your variant, click **Add**.

- These event mappings are in the variant record that are found in the `sys_ux_screen` table.
- The source events for these event mappings are any dispatched events that are defined for your page.
- The available handlers for these event mappings are canvas-level events, the UXR App Shell Data Source, and operations from inherited data resources.
- These event mappings are used as relays so that events that are dispatched from components on your page are relayed up to the user session handlers.
- These mappings are automatically created on a page save when you have mapped a component’s dispatched event to another event. For example, when you map a **Button clicked** event to the **Link to destination** event, the event cre-
<table>
<thead>
<tr>
<th>To add this event handler</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ates a dispatched relay event and a variant event mapping after you save the page.</td>
</tr>
</tbody>
</table>

a. Click **Dispatched events**.
b. Click **Add**.
c. In the Create an event screen, do the following actions:
   - Type a label for your event.
   - Select a target parent event handler.
d. To add the event handler to your dispatched events, click **Add**.

e. The new event handler that you created appears in Dispatched events. The following example
<table>
<thead>
<tr>
<th>To add this event handler</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>shows how to add a Link to destination event handler.</td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="Dispatched events" /></td>
</tr>
</tbody>
</table>

- These events are in the [sys_ux_event] table. The page definition record contains references to the [sys_ux_event] record.
- These events serve as source events for variant event mappings.
- To create new dispatched events for your page, click +Add. Configure the event label to auto-populate the event name and payload fields for your new sys_ux_event.
- These events are often used to create relay event mappings. You can select a Target parent event handler to model the payload fields after the selected event. You can also manually create payload fields for your dispatched event.

<table>
<thead>
<tr>
<th>Handled events</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="image" alt="Dispatched events" /></td>
</tr>
</tbody>
</table>

- These events are saved in the [sys_ux_event] table. The page definition record contains references to the [sys_ux_event] record.
- These events serve as source events for Page event mappings.
- To create new dispatched events for your page, click +Add. These events are not modeled after parent event handlers. Payload fields for handled events are created.
To add this event handler

<table>
<thead>
<tr>
<th>To add this event handler</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>manually. To use an existing handler's payload fields as a template, select a template and edit the fields as necessary.</td>
<td></td>
</tr>
</tbody>
</table>

7. Click **Save**.

8. To preview your page and test the data resource event that you set up, click **Open**.

---

**Bind an event to a data resource**

Assign event handlers within User Interface (UI) Builder to individual data resources on your page. When a data resource successfully fetches new data, it executes an event handler to take a user to the next step in a flow.

**Before you begin**

Role required: ui_builder_admin

**About this task**

By binding an event to a data resource, you can perform data-related actions on your page. For example, you can add a button to reload data to a page if it doesn't load the first time. You can also set up a data resource event to reload data behind the scenes of your page.

**Procedure**

1. Navigate to **Now experience framework > UI Builder**.
2. Open an experience to work in or create a new experience by selecting **Create experience in Platform**.
   For more information, see **Create an experience**.
3. Open or create a page.
   For more information, see **Work with pages**.
4. **Optional**: If you do not have any components on your page, add one to your page.
   For more information, see **Add and configure components**.
5. To add a data resource, go to the left panel, click the data icon, and then click **+Add**.
6. To add an event handler to your data resource, click the **Events** tab in the configuration of the data resource. You can add event handlers to your data resource to handle actions for Operation Initiated, Operation Succeeded, and Operation Failed.

<table>
<thead>
<tr>
<th>Data resource instances</th>
<th>🗔️ Add</th>
<th>Configuration</th>
<th>📦 Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inherited data resource instances</td>
<td>🗔️</td>
<td>🌍</td>
<td>📦</td>
</tr>
<tr>
<td>🕰️ UXR App Shell Data Source 1</td>
<td>🕒</td>
<td></td>
<td>📦</td>
</tr>
<tr>
<td>🕒 Chrome State Manager</td>
<td>🕒</td>
<td></td>
<td>📦</td>
</tr>
<tr>
<td>🕒 Dynamic Routing</td>
<td>🕒</td>
<td></td>
<td>📦</td>
</tr>
<tr>
<td>Local data resource instances</td>
<td>🗔️</td>
<td>🌍</td>
<td>📦</td>
</tr>
<tr>
<td>🕗 Delete Record 1</td>
<td>🕗</td>
<td></td>
<td>📦</td>
</tr>
</tbody>
</table>
a. To start the process of setting up an event handler for your data resource, click + Add a new event handler.

b. From Event handler preview, select an event handler that you want to bind to your data resource.

c. Configure the event handler and click Add.

7. Click Save.

8. To preview your page and test the data resource event that you set up, click Open.

Bind an event to a declarative action

Bind data elements within User Interface (UI) Builder so that you can add event actions to a declarative action.

Before you begin
Role required: ui_builder_admin

About this task
Bind a handled event to a component so that an action is performed when a user selects the component.

Procedure
1. Navigate to Now Experience Framework > UI Builder.

2. Open an experience to work in or create a new experience by selecting Create experience in Platform.
   For more information, see Create an experience.

3. Open or create a page.

4. Add a component to your page that can have a declarative action, such as an action bar or related list.
   For more information, see Add and configure components.

5. To create a declarative action definition in a table in the Now Platform®, navigate to Workspace Experience > Actions & Components.
   Choose a table where you want the declarative action to be available in. For example, you could create a Complete my work action in an incident table or you could use an existing declarative action definition record.
6. Select the **Active** check box, and then save or update the record.

7. Return to the UI Builder.

8. To invoke a handled event for the declarative action, go to the Config panel and click **Configure declarative action event mapping**.

9. Choose the declarative action that you created earlier. To continue with the example in step 5, the declarative action could be something like **Complete my work**.

10. To define what the declarative action does on your page, click **+ Add a new event handler**.
a. Give the event handler a name. The name should describe what action you want the event handler to perform.

b. Optional: Provide a description of the event handler.

c. Choose the handled event that you want to invoke.

d. Optional: Add payload values for your event handler.

e. Click **Save**.

11. Click **Done**.

12. Click **Save**, and then click **Open**.

13. Test the declarative action on your page by clicking **Complete my work** to see if it works.

---

**Now Code Editor**

Now Code Editor is a rich-text editor like interface that supports Cascading Style Sheets (CSS), Hypertext Markup Language (HTML), JavaScript, Extensible Markup Language (XML), and JavaScript Object Notation (JSON). Use Now Code Editor to modify UI configuration, data resource configuration, styles, events, client-side and server-side scripts in Now® Experience UI Builder components.

Now Code Editor supports the following features:

- Basic editing
- Debugging
- Command palette
- Code formatting
- Syntax checking and highlighting
- Auto-suggestions
- Script macros for common code
# Basic editing

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| Format code 📝  | Applies proper indentation to the script. Keyboard shortcut:  
  • Windows: Shift+Alt+F  
  • Mac: Shift+Option+F |
| Highlight syntax | Highlights the syntax of the code.                                                                                                                                                                   |
| Check syntax ✂️  | Checks for formatting errors and highlights syntax errors.  
  • Windows: Shift+Alt+C  
  • Mac: Shift+Option+C |
| Show suggestions | Displays a list of valid elements at the insertion point such as:  
  • Class names  
  • Function names  
  • Object names  
  • Variable names  
  Select and click an entry to add it to the script. Keyboard shortcut:  
  • Windows: Control+Space  
  • Mac: Control+Space  
  You can also enable or disable **Syntax highlighting** from the **Settings** menu. |
| Toggle comments ☐ | Comments one or more lines of code with two consecutive forward-slashes //. Keyboard shortcut:  
  • Windows: Control+/  
  • Mac: Command+/ |
### Action and Description

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show minimap</td>
<td>Displays the minimap of the code snippet. You can display or hide the minimap option, from the Settings menu.</td>
</tr>
<tr>
<td>Enable word wrap</td>
<td>Enables word wrap function in the editor area. You can toggle the Enable word wrap option from the Settings menu.</td>
</tr>
<tr>
<td>Show command palette</td>
<td>Displays a list of available commands for the common operations. You can execute editor commands, find and replace text, fold and unfold code blocks, toggle comments and many more tasks using the same interactive window. Keyboard shortcut • Windows: F1 • Mac: F1</td>
</tr>
<tr>
<td>Expand editor or collapse editor</td>
<td>Expands or collapses the editor. Keyboard shortcut • Windows: Control+M • Mac: Control+M</td>
</tr>
</tbody>
</table>

### Debugging

To launch Script Debugger, click the Script Debugger icon in the toolbar.

ℹ️ **Note:** You can add a breakpoint, conditional breakpoint, or logpoint, only when debugging is enabled and selected language is JavaScript.

<table>
<thead>
<tr>
<th>Task</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add breakpoint</td>
<td>Right-click beside a line number in the ruler area and select <strong>Add breakpoint</strong>.</td>
</tr>
</tbody>
</table>
(continued)

<table>
<thead>
<tr>
<th>Task</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add conditional breakpoint</td>
<td>1. Right-click beside a line number in the ruler area and select <strong>Add conditional breakpoint</strong>.</td>
</tr>
<tr>
<td></td>
<td>2. Enter a break condition in the editor.</td>
</tr>
<tr>
<td>Add logpoint</td>
<td>Right-click beside a line number in the ruler area and select <strong>Add logpoint</strong>.</td>
</tr>
<tr>
<td>Compare text in Diff mode</td>
<td>Use the side-by-side view icon ☐ and inline view icon ☐ to toggle between views.</td>
</tr>
</tbody>
</table>

**Code editor macros**

for

- **Description:** Inserts a standard for loop with an example array.
- **Output:**

```javascript
for (var i=0; i< myArray.length; i++) {
  //myArray[i];
}
```

method

- **Description:** Inserts an empty JavaScript function template.
- **Output:**

```javascript
/*
 * Description:
 * Parameters:
 * Returns:
 */
: function() {
},
```

**info**
• Description: Inserts a GlideSystem information message.
• Output:

```javascript
gs.addInfoMessage(gs.getMessage(""));
```

doc

• Description: Inserts a comment block for describing a function or parameters.
• Output:

```javascript
/**
 * Description:
 * Parameters:
 * Returns:
 */
```

vargror

• Description: Inserts a GlideRecord query for two values with an OR condition.
• Output:

```javascript
var gr = new GlideRecord('');
var qc = gr.addQuery('field', 'value1');
qc.addOrCondition('field', 'value2');
gr.query();
while (gr.next()) {
}
```

vargr

• Description: Inserts a standard GlideRecord query for a single value.
• Output:
Web services

HTTP-based web services allow diverse applications to talk to each other. ServiceNow supports both inbound (provider) and outbound (consumer) web services.

Direct inbound web services

Inbound web services allow you to access and modify ServiceNow data using a client application.

- REST API
- Scripted REST APIs
- SOAP web service
- JSONv2 Web Service
- RSS feed generator

Exporting and converting table records into complex data types

- Exporting and converting records into complex data types

Other inbound web services

- Scripted SOAP web services

Outbound web services

Outbound web services allow you to send SOAP and REST messages to external web service providers.

- Outbound SOAP web service
- Outbound REST web service
REST API

REST (REpresentational State Transfer) is a simple stateless architecture that provides standards between computer systems on the web, making it easier for them to communicate with each other.

The Now Platform provides various REST APIs, which are active by default. These APIs provide the ability to interact with various ServiceNow functionality within your application. Such functionality includes the ability to perform create, read, update, and delete (CRUD) operations on existing tables (Table API), insert data into, retrieve information from, and run transforms against a MetricBase database (MetricBase Time Series API, and many others.

For a list of available REST APIs, see REST API reference.

Note: You can view inbound API transactions in the Transaction logs. Use a link like the one below to view the transactions for the current day:

https://<instancename>.service-now.com/syslog_transaction_list.do?
sysparm_query=sys_created_onONToday%40javascript
%3Ags.beginningOfToday()%40javascript%3Ags.endOfToday()%5Etype%3Drest

REST URI format and available parameters

ServiceNow REST APIs follow standard REST API protocol. They also provide "custom" URI and query parameters to ensure backwards compatibility and provide additional functionality such as paginating long lists of results. The following sections describe the functionality behind these custom parameters, which are all optional.

REST API versioning

ServiceNow REST API URIs may include a version number, such as /api/now/v1/table/{tableName}. Version numbers identify the endpoint version that a URI accesses. By specifying a version number in your URIs, you can ensure that future updates to the REST API do not negatively impact your integration.

URIs that do not specify a version number in the URI, such as /api/now/table/{tableName}, use the latest REST endpoint for your instance version.

Supported REST API headers

The ServiceNow REST API supports various headers. Some headers are mandatory for specific endpoints and HTTP methods.
Supported HTTP request methods

- GET
- DELETE
- HEAD
- PATCH
- POST
- PUT

**Note:** HEAD methods can be used in place of GET methods to return a response without a response body.

Data format headers

REST APIs require the `Accept` and `Content-Type` request headers for proper data formatting for requests that contain a request body or response body. POST, PUT, PATCH, and DELETE operations require you to provide both headers. GET and HEAD operations require only the `Accept` header. Failing to provide the required headers results in a 400 Bad Request error.

For most ServiceNow REST APIs these request headers support the following values:

- `Accept`: application/json, application/xml
- `Content-Type`: application/json, application/xml

For the list of specific values supported by each endpoint, refer to the REST API reference.

Other headers

All requests may contain an authentication header that specifies the user credentials to use for authentication.

You can also override HTTP methods, such as GET or POST, by setting the `X-http-method-override` header.

Custom query parameters

The ServiceNow REST APIs use the following query parameters across many of the available APIs, providing consistent behavior across the APIs. Use these parameters to paginate large record sets, filter results, and restrict the number of records returned in a single query.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. For requests that exceed this number of records, use the <code>sysparm_offset</code> parameter to paginate record retrieval. This limit is applied before ACL evaluation. If no records return, including records you have access to, rearrange the record order so records you have access to return first. <strong>Note:</strong> Unusually large <code>sysparm_limit</code> values can impact system performance. Data type: Number Default: 1000</td>
</tr>
<tr>
<td>sysparm_fields</td>
<td>Comma-separated list of fields to return in the response. Data type: String</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, <code>sysparm_offset</code> is set to &quot;0&quot;. To simply page through all available records, use <code>sysparm_offset=sysparm_offset+sysparm_limit</code>, until you reach the end of all records. Do not pass a negative number in the <code>sysparm_offset</code> parameter. Data type: Number Default: 0</td>
</tr>
</tbody>
</table>
| sysparm_query  | Encoded query used to filter the result set. Syntax: `sysparm_query=<col_name><operator><value>`.  
  - `<col_name>`: Name of the table column to filter against.  
  - `<operator>`: Supports the following values:  
    - `=`: Exactly matches `<value>`.  
    - `!=`: Does not match `<value>`.  
    - `^`: Logically AND multiple query statements.  
    - `^OR`: Logically OR multiple query statements. |
LIKE: `<col_name>` contains the specified string. Only works for `<col_name>` fields whose data type is string.

STARTSWITH: `<col_name>` starts with the specified string. Only works for `<col_name>` fields whose data type is string.

ENDSWITH: `<col_name>` ends with the specified string. Only works for `<col_name>` fields whose data type is string.

`<value>`: Value to match against.

All parameters are case-sensitive. Queries can contain more than one entry, such as:
```
sysparm_query=<col_name><operator><value>[<operator><col_name><operator><value>]
```
For example:
```
(sysparm_query=caller_id=javascript:gs.getUserID()^active=true)
```

Encoded queries also supports order by functionality. To sort responses based on certain fields, use the ORDERBY and ORDERBYDESC clauses in `sysparm_query`.

Syntax:
- `ORDERBY<col_name>`
- `ORDERBYDESC<col_name>`

For example:
```
sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory
```
This query filters all active records and orders the results in ascending order by number, and then in descending order by category.

If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property `glide.invalid_query.returns_no_rows`. Set this property to true to return no rows on an invalid query.

**Note:** The `glide.invalid_query.returns_no_rows` property controls the behavior of all queries across the instance, such as in lists, scripts (`GlideRecord.query()`), and web service APIs.
sysparm_view

UI view for which to render the data. Determines the fields returned in the response.

Valid values:
- desktop
- mobile
- both

If you also specify the sysparm_fields parameter, it takes precedence.

Data type: String

Dot-walking in REST API requests

You can use dot-walking when specifying the sysparm_query or sysparm_fields parameters in requests to REST APIs that support those parameters.

ℹ️ Note: The Import Set API does not support dot-walking.

Dot-walking in sysparm_query

You can filter queries using related record values by dot-walking in the sysparm_query parameter. For example, you can retrieve all incident records where the incident **Company** has a specific **Stock symbol** value.

https://<instance>.service-now.com/api/now/table/incident?
sysparm_query=company.stock_symbol=NYX

Dot-walking in sysparm_fields

You can view field values from multiple tables by dot-walking in the sysparm_fields parameter. For example, you can retrieve the **Name**, **Sys_id**, and **Department** of each user that has certain roles, as well as the role **Name**.

The request runs on the User Roles [sys_user_has_role] table which defines a many-to-many relationship between users and roles. The response includes field values from the User [sys_user] and Roles [sys_user_role] tables.

https://<instance>.service-now.com/api/now/table/sys_user_has_role?
sysparm_fields=role%2Crole%2Crole.name%2Cuser%2Cuser.name%2Cuser.sys_id%2Cuser.department%2Csysparm_query=role%3D3d3d43716d0f6002003a2d47bce1050e0d%5EORrole%3Dac73b52d0f6002003a2d47bce1050e0e&sysparm_display_value=true
REST API HTTP response codes

Calls made to REST endpoints return HTTP response codes. You can use these response codes to ensure that the REST API executed properly. If it did not, the endpoint returns an error response code. Use the information in the error response to troubleshoot issues with your call format. For a list of standard response codes that an endpoint may return, see REST API HTTP response codes. For the list of response codes returned by a specific ServiceNow REST API, see the REST API reference.

REST API security

By default, ServiceNow REST APIs use basic authentication or OAuth to authorize user access to REST APIs/endpoints. You can also configure your instance to use multi-factor authentication to access REST APIs. There is no support for inbound mutual authentication.

The user ID that you specify in a REST endpoint call is subject to access control in the same way as an interactive user. Each request requires the proper authentication information, such as user name and password. Ensure that each endpoint request includes an Authorization header with sufficient credentials to access the endpoint.

ServiceNow REST APIs also support cookies that enable binding to the existing session.

REST API roles

In addition to user authentication, each REST endpoint can have different requirements for the roles required to access the endpoint. Some require the admin role and others require API specific roles. Role requirements are specified in the access control list (ACL) associated with the REST API/endpoint. For specifics on the valid roles for each REST API/endpoint, refer to the REST API reference or locate the associated ACL for the API/endpoint within an instance through Security System > Access Control (ACL).
REST API ACLs

REST API ACLs define criteria, such as the roles needed and conditions that a user must meet to access a ServiceNow REST API or endpoint. A single ACL may be defined for an entire REST API, such as the Table API and Attachment API ACLs, or for an individual endpoint, such as the clotho_rest_put ACL that only applies to MetricBase PUT methods.

The following ServiceNow REST API ACLs are available in the base system but are deactivated by default. All other ServiceNow REST API ACLs are active by default.

- Table API
- Aggregate API
- Import Set API
- Attachment API

For additional information on ACLs, see Access control list rules.

**Important:** You should never modify the names of REST API ACLs.

REST API table access

By default, all tables, including base system tables, global tables, and scoped tables are accessible through web services. You must fulfill any web service security requirements, such as basic authentication and ACLs to access tables through web services.

To allow access to tables without any authentication or authorization, add the table name to the Public Pages [sys_public] table with a status of Active. The REST interface still enforces any defined ACLs on associated tables. If ACL enforcement is not the desired behavior, you must deactivate the ACLs on the tables, which is not typically suggested.

You can also control direct web service access to tables using the Allow access to this table via web services check box on the table application access settings. You must select this check box to enable web service interaction with the table.

**Note:** The application access fields controlling CRUD operations, such as Can read or Can create do not apply to web service requests.

Multi-factor authentication for inbound REST

When multi-factor authentication is enabled for a user account, you must submit an MFA token with basic auth credentials when making REST requests as that user.
To send an MFA token with a REST request, append the token to the end of the user’s password in the basic auth username:password string, such as `joe.employee:password62161147`. Encode the full string including the MFA token using base64 encoding, then send the encoded string in the Authorization header.

**Multi-factor authentication REST responses**

The response to an MFA authentication request varies depending on the validity of the provided credentials and MFA token.

<table>
<thead>
<tr>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic auth credentials and MFA token are valid</td>
<td>User is authenticated and the session created. The request is processed normally.</td>
</tr>
<tr>
<td>Basic auth credentials are valid but MFA token is invalid or missing</td>
<td>Response returns error 401. The response includes the X-MFA_TOKEN header with the value <code>invalid</code>.</td>
</tr>
<tr>
<td>Basic auth credentials are invalid</td>
<td>Response returns error 401. The X-MFA_TOKEN header is not included in the response.</td>
</tr>
</tbody>
</table>

**REST API CORS support**

The REST API supports cross-origin resource sharing (CORS) security. CORS support allows you to define which domains can access each REST API. By defining a CORS rule for a domain, you can allow cross-origin requests from that domain. Cross-origin requests cannot be made from domains without a CORS rule.

⚠️ **Note:** CORS support applies only to REST APIs, including scripted REST web services. Other web service APIs, such as the SOAP API, do not support CORS.

You can configure CORS to allow access to only certain APIs/endpoints, HTTP methods, and headers from other domains. For example, you can limit requests to the Table API from a specific domain to allow only GET operations.

To view the CORS rules defined on your instance, navigate to System Web Services > CORS Rules.

You can disable CORS support for an instance by setting the `glide.rest.cors.enabled` property to `false`. When `false`, no CORS evaluation is performed on incoming REST requests. This property is `true` by default.
REST API Explorer

The REST API Explorer is a ServiceNow tool that uses information from your instance to provide a list of endpoints, methods, and variables that you can use to build and send REST requests.

After you build the request, the REST API Explorer provides code samples in multiple programming languages that you can use to initiate the request, and detailed request and response information.

To access the REST API Explorer, in your instance, navigate to System Web Services > REST API Explorer. You must have the rest_api_explorer role to access the REST API Explorer. For additional information, see Use the REST API Explorer.

⚠️ Warning: The REST API Explorer interacts with data on the current instance. Use caution when working with requests that create, edit, or delete data on a production instance.

Automated Test Framework support

The Automated Test Framework (ATF) supports inbound REST test steps. You can create automated tests for custom inbound REST APIs that you create. Creating tests for your custom REST APIs simplifies upgrade testing, and makes it possible to verify that modifications to a REST API are backward compatible.

Example REST client applications

Several example REST client applications and source code are available to demonstrate integrations using REST web services. The example REST client applications demonstrate how to use the ServiceNow REST API with an external application, such as a NodeJS or iOS application.

ℹ️ Important: These applications are provided only as a demonstration of the REST functionality and are not officially supported.

The examples are open source and available to the community. You can use these example applications to help familiarize yourself with the REST functionality, or use them as a starting point to create your own REST client applications.

Users with the rest_api_explorer role can access the list of available client applications by navigating to System Web Services > REST > Example Client Apps.

When viewing the list of applications, click an application to view the source code and additional documentation hosted on GitHub.
Developer training
On the ServiceNow® Developer Site, you can get training for Inbound REST integrations and Outbound REST integrations.

Additional information
The remainder of the REST API section contains "how to" topics that describe specific implementations using the ServiceNow REST API and provides reference information that describes various data elements used by the ServiceNow REST API.

Related information
REST API FAQ

Use the REST API Explorer
In this tutorial you will use the REST API Explorer to test the ServiceNow REST APIs.
The REST API Explorer allows you to discover ServiceNow REST APIs, quickly construct and execute requests, and view responses from ServiceNow REST APIs within your browser. Before beginning, ensure that your user account has the rest_api_explorer and web_service_admin roles. These roles are required to complete the example procedures.

Access the REST API Explorer
View available REST API resources using the REST API Explorer.

Before you begin
Role required: web_service_admin, rest_api_explorer, or admin

About this task
You can browse available APIs, API versions, and methods for each API.

Procedure
Navigate to **System Web Services > REST API Explorer.**
Retrieve existing incidents

Use a GET request to view existing incident records.

Before you begin
Role required: admin, web_service_admin, or rest_api_explorer

About this task
Use the REST API Explorer to send the following request:

GET https://instance.service-now.com/api/now/v1/table/incident

Procedure
1. In the top-left of the REST API Explorer, select Table API and version v1.
2. Click Retrieve records from a table (GET). For information, see Table API - GET table
3. In the Path Parameters section, select the Incident (incident) table.
4. Scroll to the bottom of the page and click Send.

The response includes incident records from the instance. The REST API Explorer limits queries to 10 records at a time. Only the first 10 incident records appear. The response also includes a Link header that provides the URL to query the next 10 incident records.
The response also indicates the **Status code** and **Execution time** (in milliseconds) of the request.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status code</td>
<td>200 OK</td>
</tr>
<tr>
<td>Execution time</td>
<td>85 ms</td>
</tr>
</tbody>
</table>

**Response Body**

```
{
    "result": [
        {
            "parent": "",
            "made_sla": "true",
            "closed_by": "",
            "closed_time": "",
            "upon_reject": "cancel",
            "sys_updated_on": "2013-06-02 16:48:36",
            "child_incidents": "1",
            "hold_reason": "",
            "approval_history": "",
            "skills": "",
            "number": "386018885",
            "resolved_by": "",
            "sys_updated_by": "admin",
        }
    ]
}
```

**Create an incident record**

Use a POST request to create a new record.

**Before you begin**

Role required: admin, web_service_admin, or rest_api_explorer

**About this task**

Use the REST API Explorer to send the following request:
**POST** https://instance.service-now.com/api/now/v1/table/incident

**Procedure**
1. In the top-left of the REST API Explorer, click **Create a record (POST)**.
2. In the Path Parameters section, select the **Incident (incident)** table.
3. In the Request Body section, click **Add a field**.
4. Select a field and specify a value for that field.
5. **Optional:** Click the plus sign (+) and specify any additional field to assign a value to.
   The request body updates automatically based on your entries, such as
   `{ "short_description":"Test incident creation through REST", "comments":"These are my comments" }
6. After constructing the request, click **Send**.
   The response includes a Location header that specifies where the incident was created and how to retrieve the incident.

   ☑ **Tip:** Record this header to use in the next part of this guide.
The response also indicates the **Status code** and **Execution time** (in milliseconds) of the request.

### Response

<table>
<thead>
<tr>
<th>Status code</th>
<th>201 Created</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution time (ms)</td>
<td>1298</td>
</tr>
</tbody>
</table>

### Headers

- Cache-control: no-cache, no-store, must-revalidate, max-age=1
- Content-encoding: gzip
- Content-type: application/json; charset=UTF-8
- Date: Thu, 11 Sep 2018 19:00:45 GMT
- Expires: 0
- Location: https://api.now/v1/table/incident/f08f88bac134e23c0f06c33226144b0d4
- Pragma: no-store, no-cache
- Server: ServiceNow
- Strict-transport-security: max-age=63072000; includeSubDomains
- Transfer-encoding: chunked
- X-la-logged-in: true
- X-transaction-id: 93b8887ac13a8

### Response Body

```json
{  "result": {  "parent": "",  "made_sla": "true",  "caused_by": "",  "watch_list": "",  "upon_reject": "cancel",  "sys_updated_on": "2018-09-11 19:00:45",  "child_incidents": ""0",  "held_reason": "",  "approval_history": """,  "skills": "",  "number": "INC00100009",  "resolved_by": "",  "sys_updated_by": "terry.barracoughsnc",  "opened_by": ""}}
```

## Read the inserted incident

Use the Location header from the POST response to run a GET request.

### Before you begin

**Role required:** admin, web_service_admin, or rest_api_explorer

**About this task**

Use the REST API Explorer to send the following request:

```
GET https://instance.service-now.com/api/now/v1/table/incident/{sys_id}
```
Procedure

1. In the top-left of the REST API Explorer, click **Retrieve a record (GET)**.
2. In the Path Parameters section, select the incident table.
3. In the **sys_id** field, enter the sys_id of the record you created.
   The record sys_id appears as a 32-character string at the end of the POST response Location header.
4. Click **Send**.
   The response body contains a text representation of the record. You can control the format of the response, such as JSON or XML, using the **Response Format** field.
   The response also indicates the **Status code** and **Execution time** (in milliseconds) of the request.
Update the incident

Update the incident record using either a PUT or PATCH function.

**Before you begin**
Role required: admin, web_service_admin, or rest_api_explorer

**About this task**
Use the REST API Explorer to send the following request:

PUT https://instance.service-now.com/api/now/v1/table/incident/{sys_id}?sysparm_exclude_ref_link=true

**Procedure**

1. In the top-left of the REST API Explorer, click **Modify a record (PUT)** or **Update a record (PATCH)**.
2. In the Path Parameters section, select the **Incident (incident)** table.
3. In the **sys_id** field, enter the sys_id of the record you created.
4. In the Request Body section, click **Add a field**.
5. Select the **Short description** field and specify a new value.
6. Click **Send**.
   - The response indicates the **Status code** and **Execution time** (in milliseconds) of the request.
7. Verify that the Response Body contains the updated **short_description** value.

**Delete the incident**

Delete the incident using a DELETE request.

**Before you begin**

Role required: admin, web_service_admin, or rest_api_explorer

**About this task**

Use the REST API Explorer to send the following request:

```
DELETE https://instance.service-now.com/api/now/v1/table/incident/{sys_id}
```

**Procedure**

1. In the top-left of the REST API Explorer, click **Delete a record (DELETE)**.
2. In the Path Parameters section, select the **Incident (incident)** table.
3. In the **sys_id** field, enter the **sys_id** of the record you created.

4. Click **Send**.

   The response indicates the **Status code** and **Execution time** (in milliseconds) of the request.

![Response Table](image)

**Response**

<table>
<thead>
<tr>
<th>Status code</th>
<th>204 No Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution time (ms)</td>
<td>2699</td>
</tr>
<tr>
<td>Headers</td>
<td></td>
</tr>
<tr>
<td>Content-encoding</td>
<td>gzip</td>
</tr>
<tr>
<td>Date</td>
<td>Tue, 11 Sep 2018 21:10:59 GMT</td>
</tr>
<tr>
<td>Server</td>
<td>ServiceNow</td>
</tr>
<tr>
<td>Strict-transport-security</td>
<td>max-age=600; includeSubDomains</td>
</tr>
<tr>
<td>X-is-logged-in</td>
<td>true</td>
</tr>
<tr>
<td>X-transaction-id</td>
<td>2a96ab6c1328</td>
</tr>
</tbody>
</table>

5. Verify that the response **Status code** is **204**.

**Explore the REST API for a table**

You can explore the REST API for a table directly from that table. Explore the API using the REST API Explorer to quickly construct and test REST requests for that table.

**Before you begin**

Role required: itil, personalize_dictionary, and rest_api_explorer

**Procedure**

1. Navigate to any form or list.
2. In a form, right-click the form header and select **Configure > Table**.
3. In a list, perform the appropriate action for the list version.
• For List v2, right-click any column heading and select **Configure > Table**.

• For List v3, open the list title menu and select **Configure**, and then click **Table**.

**4. Click the Explore REST API related link.**

The REST API Explorer opens, displaying the Table API for the selected table. If the table does not allow web service interaction, a warning appears instead.

**What to do next**

Use the REST API Explorer to construct and test REST requests for the table.

**Inbound REST API rate limiting**

To prevent excessive inbound REST API requests, set rules that limit the number of inbound REST API requests processed per hour. You can create rules to limit requests for specific users, users with specific roles, or all users.

⚠️ **Note:** As requests reach an instance, each node maintains a rate limit count per user. Every 30 seconds, the count is committed to the database. As a result, a rate limit rule may not take effect for up to 30 seconds.

**Rate limiting priority**

If an inbound REST API request matches multiple rate limit rules for the same resource, rate limiting priority is enforced as follows:

• Rules set for **Single user** override rules for **All users** and rules for **Users with role**.

• Rules set for **Users with role** override rules for **All users**.

In this example, there are four rate limit rules for the same REST API resource: GET /now/v2/table/incident:

<table>
<thead>
<tr>
<th>Name</th>
<th>REST API resource</th>
<th>Requests</th>
<th>Type</th>
<th>Role</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limit Incidents</td>
<td>GET /now/v2/table/incident</td>
<td>2</td>
<td>All users</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Limit Incidents by import admin Role</td>
<td>GET /now/v2/table/incident</td>
<td>3</td>
<td>Users with role</td>
<td>import admin</td>
<td></td>
</tr>
<tr>
<td>Limit Incidents by llb Role</td>
<td>GET /now/v2/table/incident</td>
<td>5</td>
<td>Users with role</td>
<td>llb</td>
<td></td>
</tr>
<tr>
<td>Limit Incidents by user</td>
<td>GET /now/v2/table/incident</td>
<td>10</td>
<td>Single user</td>
<td>IT User</td>
<td></td>
</tr>
</tbody>
</table>

These rate limit rules are applied in the following order:
1. **Limit Incidents by User** applies to ITIL User, who can submit up to 10 requests per hour.

2. **Limit Incidents by import admin Role** applies to each user with the import_admin role. Each user with the import_admin role can submit up to three requests per hour.

3. **Limit Incidents by itil Role** applies to each user with the itil role. Each user with the itil role can submit up to five requests per hour.

4. **Limit Incidents** applies to all users. Each user can submit up to two requests per hour.

When ITIL User makes the request `GET /now/v2/table/incident`, the request matches the criteria for three rules: Limit Incidents, Limit Incidents by itil Role, and Limit Incidents by User. Only the Limit Incidents by User rule is applied because it takes precedence over the other rules. As a result, ITIL User can submit a maximum of 10 requests per hour.

If a user has two or more roles matching the criteria of multiple rate limiting rules for a REST API resource, the rule allowing the lowest number of requests applies to the user's requests for the resource. For the example rules in the figure above, assume that user Abel Tuter has both the import_admin role and the itil role. When Abel Tuter submits a request, it meets the criteria for both the Limit Incidents by admin Role rule and the Limit Incidents by itil Role rule. Only the Limit Incidents by admin Role rule is applied because it allows the lowest number of requests. As a result, Abel Tuter can submit a maximum of three requests per hour.

**REST API response headers**
You can generate inbound REST API requests using the Use the REST API Explorer or an HTTP client, such as Postman. If the request matches a rate limit rule, several HTTP response headers provide information about rate limiting:

- **X-RateLimit-Limit** displays the number of requests allowed per hour.
- **X-RateLimit-Reset** displays the Unix time until the next scheduled reset.
• **X-RateLimit-Rule** displays the sys_id of the rate limit rule that is being enforced.

<table>
<thead>
<tr>
<th>Body</th>
<th>Cookies (4)</th>
<th>Headers (19)</th>
<th>Test Results</th>
<th>Status: 200 OK</th>
<th>Time: 1151 ms</th>
<th>Size: 167.22 KB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache-control → no-cache,no-store,must-revalidate,max-age=1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Content-Encoding → gzip</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Content-Type → application/json;charset=UTF-8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date → Wed, 28 Mar 2018 21:11:01 GMT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expires → 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pragma → no-store,no-cache</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Server → ServiceNow</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set-Cookie → gilda_user=Secure; Max-Age=0; Expires=Thu, 01-Jan-1970 00:00:00 GMT; Path=/; HttpOnly</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set-Cookie → gilda_user_session=Secure; Max-Age=0; Expires=Thu, 01-Jan-1970 00:00:00 GMT; Path=/; HttpOnly</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set-Cookie → gilda_user_route=gilda.f0ed326504b8434b67874c7355011e0;Secure; Max-Age=2147483647; Expires=Thu, 16-Apr-2096 00:25:08 GMT; Path=/; HttpOnly</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set-Cookie → gilda_session_store=54f74e77e1309130592f540224450af;Secure; Max-Age=1800; Expires=Wed, 28-Mar-2018 21:41:01 GMT; Path=/; HttpOnly</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Strict-Transport-Security → max-age=63072000; IncludeSubDomains</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer-Encoding → chunked</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X-4s-Logged-In → true</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X-RateLimit-Limit → 10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X-RateLimit-Rosat → 1522774000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X-RateLimit-Rule → bae1d2d2e1309130592f5402224450af55</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X-Total-Count → 55</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X-Transaction-4D → b:3d4eb7e1309</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

If a request is denied because it exceeds the rate limit, the system returns a **Retry After** response header in addition to the response headers about rate limiting. The **Retry After** response header displays the number of seconds after which you can retry the request to avoid exceeding the rate limit. The following error response is returned:

```
{
    "error": {
        "message": "Rate limit exceeded",
        "detail": "Rate limit of 10 requests per hour for Table API exceeded"
    },
    "status": "failure"
}
```

The status of a denied request is **429 Too Many Requests**.
Create an inbound REST API rate limit

Create rate limit rules to limit the number of inbound REST API requests processed per hour.

Before you begin
Role required: rate_limit_admin

About this task
Set rate limits for all users, users with specific roles, or all users.

Note: As requests reach an instance, each node maintains a rate limit count per user. Every 30 seconds, the count is committed to the database. As a result, a rate limit rule may not take effect for up to 30 seconds.

Procedure
1. Navigate to System Web Services > REST > Rate Limit Rules.
2. Click New and enter the following field values.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST API resource</td>
<td>Value derived from the values entered at the following fields.</td>
</tr>
<tr>
<td>Name</td>
<td>Unique name for the rate limit rule.</td>
</tr>
<tr>
<td>REST API</td>
<td>REST API selected from the list of all external-facing REST APIs for the instance.</td>
</tr>
<tr>
<td>Version</td>
<td>Version of the REST API. Values listed depend on the REST API selected.</td>
</tr>
<tr>
<td><strong>Field</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Resource</td>
<td>Resource for the <strong>Version</strong>. Values listed depend on the <strong>Version</strong> selected.</td>
</tr>
<tr>
<td>Table</td>
<td>Table that you want to target. Appears only when you select <strong>Table API</strong> as the <strong>REST API</strong>.</td>
</tr>
<tr>
<td>Import set table</td>
<td>Import set table that you want to target. Appears only you select <strong>Import Set API</strong> as the <strong>REST API</strong>.</td>
</tr>
<tr>
<td>Active</td>
<td>Check box to indicate that the rate limit rule is active. Rate limit rules are activated by default as soon as you create them. You can deactivate rate limit rules to stop enforcing a rate limit or activate rate limit rules to resume enforcing a rate limit.</td>
</tr>
<tr>
<td>Request limit per hour</td>
<td>Maximum number of requests allowed per hour.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Whenever you update the value of this field, the Now Platform resets the count of requests to 0 and deletes all violations for the current hour.</td>
</tr>
<tr>
<td>Apply to</td>
<td>Users restricted by this rule:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Single user</strong> applies the rate limit to a specific user.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Users with role</strong> applies the rate limit to all users with a specific role.</td>
</tr>
<tr>
<td></td>
<td>• <strong>All users</strong> applies the rate limit to all users.</td>
</tr>
<tr>
<td>Role</td>
<td>Role to which the rate limit applies. Appears only when you select <strong>Users with role</strong> at the <strong>Apply to</strong> field.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>User</td>
<td>User to whom the rate limit applies. Appears only when you select <strong>Single user</strong> at the <strong>Apply to</strong> field.</td>
</tr>
</tbody>
</table>

3. Click **Submit**. The new rate limit goes into effect.

**What to do next**
After you submit the rule, the Now Platform adds the following related lists to the rule record:

- **Rate Limit Counts**
  Lists, by user, the number of inbound REST API requests affected by this rate limit rule.

- **Rate Limit Violations**
  Lists, by user, the violations of this rate limit rule.

You can use these related lists to **Monitor inbound REST API rate limit counts and violations**.

**Reset an inbound REST API rate limit**
Reset a rate limit rule to reset the rate limit count to zero (0) and delete any violations for the current hour.

**Before you begin**
Role required: rate_limit_admin

**Procedure**
1. Navigate to **System Web Services > REST > Rate Limit Rules**.
2. Select the rate limit rule for which you want to reset the rate limit count.
3. Click the **Reset Rate Limit Counts** related link.

**Results**
For the current hour, the system resets the rate limit count for the rate limit rule to zero (0) and removes all violations. The system begins incrementing the rate limit counts and violations as REST API requests are received for processing.

**Monitor inbound REST API rate limit counts and violations**
To determine if you have set a rate limit rule appropriately, monitor the counts and violations for inbound REST API requests that are restricted by the rule.
Before you begin
Role required: rate_limit_admin

Procedure
1. Navigate to System Web Services > REST > Rate Limits.
2. Select the rate limit rule for which you want to monitor rate limit counts and violations.
3. View the information in the following related lists:
   • In the Rate Limit Counts related list, view the count of inbound REST API requests limited by the rule. This list is cleared daily.
   • In the Rate Limit Violations related list, view the number of inbound REST API requests that exceeded the Request limit per hour value for the rule. This list is cleared biweekly.

Investigate inbound REST API rate limit violations
Investigate rate limit violations to determine which rate limit rules are being exceeded and which users are exceeding those rate limits.

Before you begin
Role required: rate_limit_admin

About this task
Violations are created when inbound REST API requests reach the maximum allowed request count for a specific REST API for that hour.

Procedure
1. Navigate to System Web Services > REST > Rate Limit Violations.
2. On the Rate Limit Violations page, select the rate limit rule that you want to investigate.
3. On the Rate Limit Violations related list, review the violations by user. This list is cleared biweekly.

What to do next
You may need to reevaluate the Request limit per hour value for a rate limit rule, depending on the number of violations of that rule. You may also need to educate users about rate limits, depending on how many times specific users violate rate limit rules.
Debug REST queries

You can debug REST queries by reviewing the session debug log.

When the `glide.rest.debug` property is `true`, all REST processing is logged in the session debug log.

REST Logging includes processing durations, headers, and the request body. Prolonged use of this property can affect performance, so it is best to use it while debugging REST processing, and then set the property back to `false`.

⚠️ **Note:** You may not see the resulting log statements if you are not on the application node that processed your REST request. In this case, please contact Technical Support.

You can include session debug logs in a REST response body by passing the `X-WantSessionDebugMessages` header in the request. For more information, see Returning session debug logs in a REST response. To view debug logs, see Display debugging logs.

Sample log output

```
#28 /api/now/table/incident
2014-03-19 11:10:37 (653) REST API-thread-1 SYSTEM DEBUG: [REST API] RESTAPIProcessor : Started initializing REST Request
2014-03-19 11:10:37 (656) REST API-thread-1 SYSTEM DEBUG: [REST API] RESTAPIProcessor : Request Header: user-agent:Mozilla/5.0 (Macintosh; Intel Mac OS X 10.7; rv:12.0) Gecko/20100101 Firefox/12.0
2014-03-19 11:10:37 (657) REST API-thread-1 SYSTEM DEBUG: [REST API] RESTAPIProcessor : Request Header: content-type:application/json; charset=UTF-8
2014-03-19 11:10:37 (657) REST API-thread-1 SYSTEM DEBUG: [REST API] RESTAPIProcessor : Request Header: cookie:glide_user_route=glide.20e7f4cd6bdc0d444810117aacc0eeae;JSESSIONID=F07CE6ACF8AF237CB239AF43B7F360BF;
```
2014-03-19 11:10:38 (357) REST API-thread-1 SYSTEM [REST API] RouteRegistry : Loaded Routes to Cache
2014-03-19 11:10:38 (357) REST API-thread-1 SYSTEM DEBUG: [REST API] RouteRegistry : Route loading time 0:00:00.105
2014-03-19 11:10:38 (357) REST API-thread-1 SYSTEM DEBUG: [REST API] RouteRegistry : Route loading time 0:00:00.105
2014-03-19 11:10:38 (424) REST API-thread-1 SYSTEM DEBUG: [REST API] RESTAPIProcessor : Pre-Service processing duration 0:00:00.000
2014-03-19 11:10:39 (508) REST API-thread-1 083A6031D7231100261253B2B252035C DEBUG: [REST API] TableAPIService : Glide Record Insert Duration 0:00:00.956
Return session debug logs in a REST response

You can include session debug logs in a REST response body by passing the X-WantSessionDebugMessages header in the request.

To return session debug messages when session debugging is enabled for the current session, set the header X-WantSessionDebugMessages to true in the REST request.

⚠️ Note: You must enable session debugging before sending this header.

**Example: Request**

This example demonstrates a Table API request made using REST API Explorer with Session Debug SQL enabled.

```
GET api/now/table/incident/9c573169c611228700193229fff72400
X-WantSessionDebugMessages:true
Content-Type: application/json; charset=UTF-8
Accept: application/json, text/plain,*/*
X-UserToken: <user token>
```
Example: Response body

```json
{
    "result": {
        "description": "User can't access email on mail.company.com.",
        "number": "INC0000001"
    },
    "session": {
        "debug_logs": [
            {
                "type": "sql",
                "customerUpdate": false,
                "line": "17:17:27.777: Time: 0:00:00.000 for: glide_master_db[glide.5] .... sys_user_session0.id = '3BEA7001EB230200C46AC2EEF106FE2A'",
                "debugClassSet": ""
            },
            {
                "type": "sql",
                "customerUpdate": false,
                "line": "17:17:27.779: Time: 0:00:00.002 for: glide_master_db[glide.6] .... \`sys_id` = '7f97b01ebc230200c46ac2eeff106f2a'",
                "debugClassSet": ""
            }
        ]
    }
}
```

Related information

**Session debug**

**CORS domain requirements**

When you define a cross-origin resource sharing (CORS) rule, the value you enter in the **Domain** field must meet certain requirements. Each CORS rule supports a single wildcard to match incoming Origin headers.

**Requirements**

The value you enter in the **Domain** field must meet the following requirements.

- Begins with **HTTP://** or **HTTPS://**.
- Is a domain pattern or IP address.
- Ends with alphanumeric characters preceded by a period, such as .com.
- Includes at most a single wildcard character immediately following the scheme and hierarchical portion of the domain pattern.

### Wildcard

You can use a single wildcard character (*) in the domain pattern. Use this wildcard immediately following the scheme and hierarchical portion of the domain pattern, such as http://*.domain.com to include all subdomains. The wildcard must immediately follow the scheme and hierarchical portion of the domain pattern. If you use an IP address instead of domain pattern, you must enter the full IP address without a wildcard.

**Note:** You cannot use multiple wildcards, or specify a wildcard without a domain pattern. Values such as * or *.* are not supported.

### Domain matching

When evaluating the Origin header in a request, ServiceNow prioritizes rules that match the domain pattern exactly. If no exact match is found, the next closest match is used.

For example, if there are rules for the domain patterns `http://*.blog.mysite.com` and `http://*.mysite.com`, a request from `http://alice.blog.mysite.com` will match the `http://*.blog.mysite.com` pattern.

### Examples of valid and invalid domains

<table>
<thead>
<tr>
<th>Valid domain</th>
<th>Invalid domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>http://*.ms.net</td>
<td>https://*com</td>
</tr>
<tr>
<td>https://*.ms.com</td>
<td>http://*.com</td>
</tr>
<tr>
<td><a href="http://192.168.1.1">http://192.168.1.1</a></td>
<td>http://*.168.1.126</td>
</tr>
<tr>
<td>http://*.service-now.com</td>
<td><a href="http://blog.*.service-now.com">http://blog.*.service-now.com</a></td>
</tr>
<tr>
<td>http://*.com</td>
<td>http://*com</td>
</tr>
</tbody>
</table>
**Define a CORS rule**

You can define a CORS rule to control which domains can access specific REST API endpoints.

**Before you begin**

Role required: web_service_admin

**Procedure**

1. Navigate to **System Web Services > CORS Rules**.
2. Click **New**.
3. Populate the form.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST API</td>
<td>Select the REST API this CORS rule applies to, such as the Table API.</td>
</tr>
<tr>
<td>Domain</td>
<td>Enter the domain that this CORS rule applies to. This CORS rule is evaluated against requests from the specified domain. You can specify a domain pattern or an IP address. When using a domain pattern you can specify a single wildcard to match incoming origin headers.</td>
</tr>
<tr>
<td>HTTP Methods</td>
<td>Select the HTTP methods allowed. Only the selected methods can be called from the specified domain.</td>
</tr>
<tr>
<td>HTTP Headers</td>
<td>Enter a comma-separated list of HTTP headers to send in the response. Specified headers are added to the Access-Control-Expose-Headers header.</td>
</tr>
<tr>
<td>Max Age</td>
<td>Enter the number of seconds to cache the client session. After an initial CORS request, further requests from the same client within the specified time do not require a preflight message. If you do not specify a value, the default value of 0 indicates that all requests require a preflight message.</td>
</tr>
</tbody>
</table>

4. Click **Submit**.

**Enable OAuth with inbound REST**

Using OAuth, you can pass a user ID and password once, and then use a token for subsequent REST requests instead of submitting credentials with each request.
About this task
OAuth can improve system security by reducing the number of times you submit user credentials. You can use OAuth to authenticate REST requests.
This video demonstrates how to authenticate to REST APIs using OAuth.
How to authenticate to REST APIs using OAuth

Procedure
1. Activate the OAuth 2.0 plugin.
2. Set the system property com.snc.platform.security.oauth.is.active to true.
3. Navigate to System OAuth > Application Registry.
4. Click New and then click Create an OAuth API endpoint for external clients.
5. Record the client_id and client_secret values from the previous step to use when requesting an access token.
6. To get an access token, use your REST client, such as cURL or Postman, to send a request to the OAuth endpoint (oauth_token.do).
   Format the request as a URL-encoded HTTP POST body and include the required parameters.
7. Record the access token and refresh token from the response.
8. Submit the access token with subsequent REST requests.

REST OAuth example
This example shows how to authenticate an inbound REST request using OAuth.
In this example, the OAuth token has a client_id of a329c4515612210071a5e0c298ee2be8 and a client_secret of password22.

Getting an access token

```sh
curl -d "grant_type=password&client_id=a329c4515612210071a5e0c298ee2be8&client_secret=password22&username=RESTUser&password=RESTUserPassword" https://<instance>.service-now.com/oauth_token.do
```

Sample token response

```
{
  "scope": "useraccount",
  "token_type": "Bearer",
  "expires_in": 1799,
}
```
REST request with OAuth token

```
2wRlsRCT2SYjCCJP91kwo2EFzj5qg40I33aC09e0-0h6Ib33YK7If-LMiNorNuglfqbkL4AfkYC92KYHUCcbpQ"
"https://<instance>.service-now.com/api/now/table/incident
```

REST API HTTP response codes

REST Messages sent to an instance return a specific HTTP response code.

<table>
<thead>
<tr>
<th>Status Code</th>
<th>Message</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Success</td>
<td>Success with response body.</td>
</tr>
<tr>
<td>201</td>
<td>Created</td>
<td>Success with response body.</td>
</tr>
<tr>
<td>204</td>
<td>Success</td>
<td>Success with no response body.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
<td>The request URI does not match the APIs in the system, or the operation failed for unknown reasons. Invalid headers can also cause this error.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized</td>
<td>The user is not authorized to use the API.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden</td>
<td>The requested operation is not permitted for the user. This error can also be caused by ACL failures, or business rule or data policy constraints.</td>
</tr>
<tr>
<td>404</td>
<td>Not found</td>
<td>The requested resource was not found. This can be caused by an ACL constraint or if the resource does not exist.</td>
</tr>
<tr>
<td>405</td>
<td>Method not allowed</td>
<td>The HTTP action is not allowed for the requested REST API, or it is not supported by any API.</td>
</tr>
<tr>
<td>406</td>
<td>Not acceptable</td>
<td>The endpoint does not support the response format specified in the request Accept header.</td>
</tr>
<tr>
<td>415</td>
<td>Unsupported media type</td>
<td>The endpoint does not support the format of the request body.</td>
</tr>
</tbody>
</table>
REST API reference
You can use REST interfaces to access data on your instance.

Account API
The Account REST API enables you to retrieve and update Customer Service Management (CSM) account records.

In addition, you can generate new social media profile records when creating an account.

The Account API requires the Customer Service plugin (com.sn_customerservice) and is provided within the now namespace.

Users require the csm_ws_integration role for full API access.

Account - GET /now/account
Retrieves a specified set of Customer Service Management (CSM) accounts.

URL format
Versioned URL: /api/now/{api_version}/account
Default URL: /api/now/account

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>api_version</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>sysparm_limit</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, <code>sysparm_offset</code> is set to &quot;0&quot;. To simply page through all available records, use <code>sysparm_offset=sysparm_offset+sysparm_limit</code>, until you reach the end of all records. Do not pass a negative number in the <code>sysparm_offset</code> parameter. Data type: Number Default: 0</td>
</tr>
</tbody>
</table>
| sysparm_query    | Encoded query used to filter the result set. For example:  

```plaintext
sysparm_query=caller_id=javascript:gs.getUserID()\^active=true
```

The encoded query supports order by. To sort responses based on certain fields, use the ORDERBY and ORDERBYDESC clauses in `sysparm_query`. For example, `sysparm_query=active=true\^ORDERBYnumber\^ORDERBYDESCcategory` filters all active records and orders the results in ascending order by number first, and then in descending order by category.

If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property `glide.invalid_query.returns_no_rows`. Set this property to true to return no rows on an invalid query.

**Note:** The `glide.invalid_query.returns_no_rows` property controls the behavior of all queries across the instance, such as in lists, scripts (`GlideRecord.query()`), and web service APIs.

Data type: String
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account_code</td>
<td>Unique combination of values that an application uses to identify budget forecasts and budget plans. Data type: String, Maximum length: 255</td>
</tr>
<tr>
<td>account_parent</td>
<td>Sys_id of the parent account of this account. Located in the Account [customer_account] table. Data type: String</td>
</tr>
<tr>
<td>account_path</td>
<td>Path from the parent to the child accounts in the account hierarchy. Data type: String, Maximum length: 255</td>
</tr>
<tr>
<td>active_escalation</td>
<td>Sys_id of the active escalation associated with the account. Located in the Escalation [sn_customerservice_escalation] table. Data type: String</td>
</tr>
<tr>
<td>apple_icon</td>
<td>Icon for iPhone home page bookmarks. Data type: Image</td>
</tr>
<tr>
<td>banner_image</td>
<td>Banner image that appears on the customer portal. Data type: Image</td>
</tr>
<tr>
<td>banner_image_light</td>
<td>Small banner image. Data type: Image</td>
</tr>
<tr>
<td>banner_text</td>
<td>Banner text that appears on the customer portal. Data type: String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Maximum length: 4,000</td>
<td></td>
</tr>
<tr>
<td>city</td>
<td>City in which the company that is associated with this account resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>contact</td>
<td>Sys_id of a contact record associated with this account. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>country</td>
<td>Country in which the company that is associated with this account resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: USA</td>
</tr>
<tr>
<td>customer</td>
<td>Flag that indicates whether the account is a customer account, as opposed to a partner account. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Customer account</td>
</tr>
<tr>
<td></td>
<td>• false: Partner account</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>discount</td>
<td>Discount given to the account on purchases.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 15</td>
</tr>
<tr>
<td>fax_phone</td>
<td>Primary fax phone number for the company associated with this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>fiscal_year</td>
<td>Fiscal year for the company associated with the account.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>lat_long_error</td>
<td>Difference in the actual location as compared to latitude and longitude information.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 1,000</td>
</tr>
<tr>
<td>latitude</td>
<td>Latitude of the company associated with this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (floating point number)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>longitude</td>
<td>Longitude of the company associated with this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (floating point number)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>manufacturer</td>
<td>Flag that indicates whether the company associated with this account manufactures goods.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Manufactures goods</td>
</tr>
<tr>
<td></td>
<td>• false: Does not manufacture goods</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>market_cap</td>
<td>Market value of the associated company's publicly traded stock shares.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>name</td>
<td>Name of the company associated with this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 80</td>
</tr>
<tr>
<td>notes</td>
<td>Additional information about the company.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>num_employees</td>
<td>Number of people employed by the company.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>number</td>
<td>Number that identifies this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>parent</td>
<td>Sys_id of the parent account of this account. Located in the Company [core_company] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>partner</td>
<td>Flag that indicates whether the account is a partner account or a customer account.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Partner account</td>
</tr>
<tr>
<td></td>
<td>• false: Customer account</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>phone</td>
<td>Primary phone number for the company.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>primary</td>
<td>Flag that indicates whether this is a primary account.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Primary account</td>
</tr>
<tr>
<td></td>
<td>• false: Secondary account</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>primary_contact</td>
<td>Sys_id of the primary contact for the account. Located in the Contact</td>
</tr>
<tr>
<td></td>
<td>[customer_contact] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>profits</td>
<td>Profit information entered for this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Currency)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>publicly_traded</td>
<td>Flag that indicates whether the company associated with this account is</td>
</tr>
<tr>
<td></td>
<td>publicly traded on the stock exchange.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Publicly traded</td>
</tr>
<tr>
<td></td>
<td>• false: Private company</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>rank_tier</td>
<td>Type of account.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>blacklist</td>
<td>• blacklist</td>
</tr>
<tr>
<td>strategic</td>
<td>• strategic</td>
</tr>
<tr>
<td>tactical</td>
<td>• tactical</td>
</tr>
<tr>
<td>valued</td>
<td>• valued</td>
</tr>
<tr>
<td>other</td>
<td>• other</td>
</tr>
<tr>
<td>registration_code</td>
<td>Unique code that customers use when requesting a login on the customer portal. This code provides a method for validating the customer on the company before granting access.</td>
</tr>
<tr>
<td>revenue_per_year</td>
<td>Revenue produced by the company associated with this account.</td>
</tr>
<tr>
<td>state</td>
<td>State in which the company resides.</td>
</tr>
<tr>
<td>stock_price</td>
<td>Price of the company stock.</td>
</tr>
<tr>
<td>stock_symbol</td>
<td>Stock symbol of the company.</td>
</tr>
<tr>
<td>street</td>
<td>Street address of the company.</td>
</tr>
</tbody>
</table>

Data type: String
Maximum length: 40
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_class_name</td>
<td>Table that contains the associated account record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_created_by</td>
<td>User that originally created the account.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sys_created_on</td>
<td>Date and time that the account was originally created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id for the account record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_mod_count</td>
<td>Number of times the account information has been updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td>sys_updated_by</td>
<td>User that last modified the account information.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sys_updated_on</td>
<td>Date and time the account information was last updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>theme</td>
<td>Sys_id of the customer portal theme used by this account. Located in the</td>
</tr>
<tr>
<td></td>
<td>Theme [sys_ui_theme] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>vendor</td>
<td>Flag that indicates whether the company associated with this account is a</td>
</tr>
<tr>
<td></td>
<td>vendor.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td><strong>true</strong>: Vendor</td>
</tr>
<tr>
<td></td>
<td><strong>false</strong>: Not a vendor</td>
</tr>
<tr>
<td>Data type:</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default:</td>
<td>false</td>
</tr>
<tr>
<td>vendor_manager</td>
<td>List of sys_ids of the vendor managers for the account. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td>Data type:</td>
<td>String</td>
</tr>
<tr>
<td>vendor_type</td>
<td>List of sys_ids of the type of vendor such as, applications, hardware, services, or software. Located in the Vendor Type [vendor_type] table.</td>
</tr>
<tr>
<td>Data type:</td>
<td>String</td>
</tr>
<tr>
<td>website</td>
<td>URL of the website for the company.</td>
</tr>
<tr>
<td>Data type:</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 1,024</td>
</tr>
<tr>
<td>zip</td>
<td>Zip code of the company.</td>
</tr>
<tr>
<td>Data type:</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl "https://instance.servicenow.com/api/now/account?sysparm_limit=2&sysparm_offset=2>;rel="next" \  --request GET \
   --header "Accept:application/json" \
   --user 'username':'password'

{
 "result": {
 
  "banner_image_light": "",
  "country": "USA",
```
"parent": "",
"notes": "",
"stock_symbol": "",
"discount": "",
"active_escalation": "",
"sys_updated_on": "2019-01-03 19:37:55",
"apple_icon": "",
"number": "ACCT0000003",
"sys_updated_by": "admin",
"fiscal_year": "",
"sys_created_on": "2018-12-23 05:25:17",
"contact": "",
"stock_price": "",
"state": "",
"banner_image": "",
"sys_created_by": "admin",
"longitude": "",
"zip": "BR1 3QR",
"profits": "0",
"phone": "+44 20 8466 9992",
"fax_phone": "",
"name": "Boxeo EMEA",
"banner_text": "",
"account_code": "~~~~3",
"primary": "false",
"city": "Bromley",
"latitude": "",
"sys_class_name": "customer_account",
"manufacturer": "false",
"account_parent": "86837a386f0331003b3c498f5d3ee4ca",
"sys_id": "3eedd08413651200042ab3173244b088",
"market_cap": "0",
"num_employees": "",
"rank_tier": "",
"street": "18 London Rd",
"vendor": "false",
"lat_long_error": "",
"theme": "",
"vendor_type": "",
"website": "",
"revenue_per_year": "0",
"publicly_traded": "false",
"sys_mod_count": "5",
"sys_tags": "",

4999

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


"partner": "false",
"registration_code": "BOXEO-EMEA",
"vendor_manager": "",
"account_path": "~~~~1/~~~~3",
"primary_contact": "ff66c1254fb81200025ba3618110c76e",
"customer": "true"
},
{
"banner_image_light": "",
"country": "USA",
"parent": "",
"notes": "",
"stock_symbol": "",
"discount": "",
"active_escalation": "",
"sys_updated_on": "2019-01-03 19:38:04",
"apple_icon": "",
"number": "ACCT0000004",
"sys_updated_by": "admin",
"fiscal_year": "",
"sys_created_on": "2018-12-23 05:19:24",
"contact": "",
"stock_price": "",
"state": "",
"banner_image": "",
"sys_created_by": "admin",
"longitude": "",
"zip": "V5L 2G4",
"profits": "0",
"phone": "+1 604-255-9797",
"fax_phone": "",
"name": "Boxeo Canada",
"banner_text": "",
"account_code": "~~~~4",
"primary": "false",
"city": "Vancouver",
"latitude": "",
"sys_class_name": "customer_account",
"manufacturer": "false",
"account_parent": "86837a386f0331003b3c498f5d3ee4ca",
"sys_id": "609cd80413651200042ab3173244b03e",
"market_cap": "0",
"num_employees": "",
"rank_tier": "",

5000

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/account?sysparm_limit=1&sysparm_offset=2>;rel="next"

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
Account - GET /now/account/{id}
Retrieves the specified Customer Service Management (CSM) account.

**URL format**

Versioned URL: /api/now/{api_version}/account/{id}
Default URL: /api/now/account/{id}

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>Sys_id of the account to retrieve. Located in the Customer [customer_account] table. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>404</td>
<td>Indicates that the request is invalid. Could be due to one of the following reasons:</td>
</tr>
</tbody>
</table>
Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>•</td>
<td>Requested case does not exist.</td>
</tr>
<tr>
<td>•</td>
<td>User does not have access to the account record.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account_code</td>
<td>Unique combination of values that an application uses to identify budget forecasts and budget plans. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>account_parent</td>
<td>Sys_id of the parent account of this account. Located in the Account [customer_account] table. Data type: String</td>
</tr>
<tr>
<td>account_path</td>
<td>Path from the parent to the child accounts in the account hierarchy. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>active_escalation</td>
<td>Sys_id of the active escalation associated with the account. Located in the Escalation [sn_customerservice_escalation] table. Data type: String</td>
</tr>
<tr>
<td>apple_icon</td>
<td>Icon for iPhone home page bookmarks. Data type: Image</td>
</tr>
<tr>
<td>banner_image</td>
<td>Banner image that appears on the customer portal.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>banner_image_light</td>
<td>Small banner image.</td>
</tr>
<tr>
<td></td>
<td>Data type: Image</td>
</tr>
<tr>
<td>banner_text</td>
<td>Banner text that appears on the customer portal.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>city</td>
<td>City in which the company that is associated with this account resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>contact</td>
<td>Sys_id of a contact record associated with this account. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>country</td>
<td>Country in which the company that is associated with this account resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: USA</td>
</tr>
<tr>
<td>customer</td>
<td>Flag that indicates whether the account is a customer account, as opposed to a partner account. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Customer account</td>
</tr>
<tr>
<td></td>
<td>• false: Partner account</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>discount</td>
<td>Discount given to the account on purchases.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| fax_phone         | Primary fax phone number for the company associated with this account.  
|                   | Data type: String  
|                   | Maximum length: 40  |
| fiscal_year       | Fiscal year for the company associated with the account.  
|                   | Data type: String  |
| lat_long_error    | Difference in the actual location as compared to latitude and longitude information.  
|                   | Data type: String  
|                   | Maximum length: 1,000  |
| latitude          | Latitude of the company associated with this account.  
|                   | Data type: Number (floating point number)  
|                   | Maximum length: 40  |
| longitude         | Longitude of the company associated with this account.  
|                   | Data type: Number (floating point number)  
|                   | Maximum length: 40  |
| manufacturer      | Flag that indicates whether the company associated with this account manufactures goods.  
|                   | Possible values:  
|                   | • true: Manufactures goods  
<p>|                   | • false: Does not manufacture goods  |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>market_cap</td>
<td>Market value of the associated company's publicly traded stock shares.</td>
</tr>
<tr>
<td>name</td>
<td>Name of the company associated with this account.</td>
</tr>
<tr>
<td>notes</td>
<td>Additional information about the company.</td>
</tr>
<tr>
<td>num_employees</td>
<td>Number of people employed by the company.</td>
</tr>
<tr>
<td>number</td>
<td>Number that identifies this account.</td>
</tr>
<tr>
<td>parent</td>
<td>Sys_id of the parent account of this account. Located in the Company [core_company] table.</td>
</tr>
</tbody>
</table>
| partner         | Flag that indicates whether the account is a partner account or a customer account. Possible values:
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
|                      | • true: Partner account  
• false: Customer account  
Data type: Boolean  
Default: false     |
| phone                | Primary phone number for the company.  
Data type: String     |
| primary              | Flag that indicates whether this is a primary account.  
Possible values:  
• true: Primary account  
• false: Secondary account  
Data type: Boolean  
Default: false     |
| primary_contact      | Sys_id of the primary contact for the account. Located in the Contact [customer_contact] table.  
Data type: String     |
| profits              | Profit information entered for this account.  
Data type: Number (Currency)  
Maximum length: 40     |
| publicly_traded      | Flag that indicates whether the company associated with this account is publicly traded on the stock exchange.  
Possible values:  
• true: Publicly traded  
• false: Private company     |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rank_tier</td>
<td>Type of account. Possible values: blacklist, strategic, tactical, valued, other. Data type: String  Maximum length: 40</td>
</tr>
<tr>
<td>registration_code</td>
<td>Unique code that customers use when requesting a login on the customer portal. This code provides a method for validating the customer on the company before granting access. Data type: String  Maximum length: 40</td>
</tr>
<tr>
<td>revenue_per_year</td>
<td>Revenue produced by the company associated with this account. Data type: Number (Currency)  Maximum length: 20</td>
</tr>
<tr>
<td>state</td>
<td>State in which the company resides. Data type: String  Maximum length: 40</td>
</tr>
<tr>
<td>stock_price</td>
<td>Price of the company stock. Data type: String  Maximum length: 40</td>
</tr>
<tr>
<td>stock_symbol</td>
<td>Stock symbol of the company. Data type: String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Maximum length: 40</td>
<td></td>
</tr>
<tr>
<td>street</td>
<td>Street address of the company.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Maximum length: 255</td>
<td></td>
</tr>
<tr>
<td>sys_class_name</td>
<td>Table that contains the associated account record.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>sys_created_by</td>
<td>User that originally created the account.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Maximum length: 40</td>
<td></td>
</tr>
<tr>
<td>sys_created_on</td>
<td>Date and time that the account was originally created.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id for the account record.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>sys_mod_count</td>
<td>Number of times the account information has been updated.</td>
</tr>
<tr>
<td>Data type: Number (Integer)</td>
<td></td>
</tr>
<tr>
<td>sys_updated_by</td>
<td>User that last modified the account information.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Maximum length: 40</td>
<td></td>
</tr>
<tr>
<td>sys_updated_on</td>
<td>Date and time the account information was last updated.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>theme</td>
<td>Sys_id of the customer portal theme used by this account. Located in the Theme [sys_ui_theme] table.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>vendor</td>
<td>Flag that indicates whether the company associated with this account is a vendor. Possible values: true: Vendor false: Not a vendor. Data type: Boolean Default: false</td>
</tr>
<tr>
<td>vendor_manager</td>
<td>List of sys_ids of the vendor managers for the account. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>vendor_type</td>
<td>List of sys_ids of the type of vendor such as, applications, hardware, services, or software. Located in the Vendor Type [vendor_type] table. Data type: String</td>
</tr>
<tr>
<td>website</td>
<td>URL of the website for the company. Data type: String Maximum length: 1,024</td>
</tr>
<tr>
<td>zip</td>
<td>Zip code of the company. Data type: String Maximum length: 40</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl "https://instance.servicenow.com/api/now/account/bf60bef46f0331003b3c498f5d3ee41a" \
   --request GET \ 
   --header "Accept: application/json" \ 
   --user "username":"password"
```

```json
{
   "result": {
```
"banner_image_light": "",
"country": "USA",
"parent": "",
"notes": "",
"stock_symbol": "",
"discount": "",
"active_escalation": "e4fa242887150300fe4433d4c6cb0b5f",
"sys_updated_on": "2020-07-30 21:57:54",
"apple_icon": "",
"number": "ACCT0000009",
"sys_updated_by": "admin",
"fiscal_year": "",
"sys_created_on": "2019-09-16 21:19:27",
"contact": "bea1fef46f0331003b3c498f5d3ee4c5",
"stock_price": "",
"state": "California",
"banner_image": "",
"sys_created_by": "venki",
"longitude": "-122.116445",
"zip": "94022",
"profits": "0",
"phone": "(877) 729-4269",
"fax_phone": "",
"name": "Avid Corporation",
"banner_text": "",
"account_code": "~~~~9",
"primary": "false",
"city": "Los Altos",
"latitude": "37.402666",
"sys_class_name": "customer_account",
"manufacturer": "false",
"account_parent": "",
"sys_id": "bf60bef46f0331003b3c498f5d3ee41a",
"market_cap": "0",
"num_employees": "",
"rank_tier": "",
"street": "4440 El Camino Real",
"vendor": "false",
"lat_long_error": "",
"theme": "",
"vendor_type": "",
"website": "http://www.avidcorp.com",
"revenue_per_year": "0",
"publicly_traded": "false",

5013

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/account/59e788fbd1b1200b6075200cf9619d2'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<response>
  <result>
    <country>USA</country>
    <notes/><gender>Female</gender>
    <city>Santa Clara</city>
  </result>
</response>
```
ActivitySubscriptions API
Use the ActivitySubscriptions API to retrieve Activity entities from the Subscriptions and Activity Feed Framework.

This API requires activation of the Subscriptions and Activity Feed Framework (com.snc.activity_subscriptions) plugin. Activate the Customer Central
(com.sn_csm_customer_central) plugin to seed configuration in all required tables. For more information on customer central configuration settings, see Configure customer central.

**ActivitySubscriptions - GET /now/actsub/activities**

Retrieves activity records from the Subscriptions and Activity Feed Framework.

No role is required to access this endpoint, but data returned is limited by user access settings.

**URL format**

Versioned URL: /api/now/{api_version}/actsub/activities

Default URL: /api/now/actsub/activities

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>api_version</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>before</td>
</tr>
<tr>
<td>context_instance</td>
</tr>
</tbody>
</table>
## Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>to retrieve. Located in the appropriate table for the activity context.</td>
</tr>
<tr>
<td></td>
<td>As an example, if you set <code>context</code> to the sys_id of the Consumer activity context, set this parameter to the sys_id of the Consumer <code>[csm_consumer]</code> table record for the consumer whose activities you want to retrieve.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>end_date</td>
<td>Date/time value. The request returns only records created during the time period defined by <code>start_date</code> and this parameter. Must be set along with <code>start_date</code>.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Format: <code>YYYY-MM-DD hh:mm:ss</code> or <code>YYYY-MM-DD</code></td>
</tr>
<tr>
<td>facets</td>
<td>Comma-separated list of sys_ids of activity facet types to retrieve for the specified activity context. These sys_ids are located in the Activity Facet <code>[sn_actsub_facet]</code> table.</td>
</tr>
<tr>
<td></td>
<td>For more details on activity facets, see Create facets for activity contexts.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: Return all activity facet types configured for the activity context.</td>
</tr>
<tr>
<td>last</td>
<td>Index value of the first result row omitted from the response body. The result row index starts at 0, so the last row included in the response body is the one with the index <code>last-1</code>.</td>
</tr>
<tr>
<td></td>
<td>For example, setting <code>last=5</code> omits rows with index 5 and above and returns at most only the first five result rows (with index values 0 through 4). Setting <code>stFrom=1</code> and <code>last=5</code> skips the first result row and returns the second through fifth result rows (with index values 1 through 4).</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (integer)</td>
</tr>
<tr>
<td></td>
<td>Default: The value of the <code>com.snc.actsub.activities.api.fetch.limit</code> system property (10 by default).</td>
</tr>
</tbody>
</table>
## Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>record_id</td>
<td>Sys_id of a record to use when rendering dynamic facets. Located in the appropriate table for the activity type you want to retrieve. For example, if you want to fetch activities for a Case record, set this parameter to the sys_id of the Case record in question. For more details on dynamic facets, see <a href="#">Create facets for activity contexts</a>. Data type: String</td>
</tr>
<tr>
<td>start_date</td>
<td>Date/time value. The request returns only records created during the time period defined by this parameter and end_date. Must be set along with end_date. Data type: String</td>
</tr>
<tr>
<td>stFrom</td>
<td>Index value of the first result row to include in the response body. The result row index starts at 0. For example, stFrom=2 skips the first two result rows and returns results from the third row onward. Data type: Number (integer) Default: 0</td>
</tr>
</tbody>
</table>
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected. The response body <code>result.message</code> parameter describes the nature of the error. Example error messages:</td>
</tr>
</tbody>
</table>

- Make sure a valid context and context_instance are passed or Make sure a valid stream and user are passed indicates that the request lacks one or both of the `context` and `context_instance` query parameters. These parameters are required. 
- `stFrom` can not be greater than `last` indicates that the `stFrom` query parameter value exceeds the `last` query parameter value. 
- Please provide `start_date` and `end_date` indicates that the request includes only one of the `start_date` and `end_date` query parameters, which cannot be used separately. 
- `start_date` can not be ahead of `end_date` indicates that the specified `start_date` query parameter value occurs after the specified `end_date` query parameter value. |
Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Result object.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>result</em>: {</td>
<td></td>
</tr>
<tr>
<td>result.activities</td>
<td>Array of objects in which each object represents an activity record from the specified activity context and context instance.</td>
<td>Array</td>
</tr>
<tr>
<td></td>
<td><em>activities</em>: [Array],</td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>message</em>: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>status</em>: Number,</td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>stream</em>: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>user</em>: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>&quot;created&quot;: &quot;String&quot;, &quot;icon&quot;: &quot;String&quot;, &quot;source_table_name&quot;: &quot;String&quot;, &quot;subheader_fields&quot;: [Array], &quot;subobject_sys_id&quot;: &quot;String&quot;, &quot;subobject_table_name&quot;: &quot;String&quot;, &quot;sys_id&quot;: &quot;String&quot;, &quot;title&quot;: &quot;String&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>result.activities.activity_type_id</td>
<td>Sys_id of the record from the Activity Types [sn_actsub_activity_type] table that corresponds to the type of this activity record.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.activities.content_fields</td>
<td>Array of objects in which each object represents a field to display as content in the tile for the activity record.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>content_fields</em>: [</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;deeplink_to_subobject&quot;: &quot;String&quot;, &quot;display_as_timeago&quot;: &quot;String&quot;, &quot;label&quot;: &quot;String&quot;, &quot;show_label&quot;: &quot;String&quot;, &quot;type&quot;: &quot;String&quot;, &quot;value&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>}</td>
<td></td>
</tr>
<tr>
<td>result.activities.content_fields.deeplink_to_subobject</td>
<td>Flag indicating whether or not the content field displays as a hyperlink to the record for the object on which this activity was performed.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
</table>
| result.activities.content_fields.display_as_timeago | Flag indicating whether or not the content field displays in time ago format. Valid values:  
• true: The content field displays in time ago format.  
• false: The content field does not display in time ago format.  | String          |
| result.activities.content_fields.label          | Label for the content field.                                                                                                                                                                               | String          |
| result.activities.content_fields.show_label    | Flag indicating whether or not the content field label displays for content field values in the tile for this activity record. Valid values:  
• true: The content field label displays for content field values in the tile for this activity record.  
• false: The content field label does not display for content field values in the tile for this activity record.  | String          |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.activities.content_fields.type</td>
<td>Type of the content field. Data type: String</td>
</tr>
<tr>
<td>result.activities.content_fields.value</td>
<td>Value of the content field. Data type: String</td>
</tr>
<tr>
<td>result.activities.created</td>
<td>Creation date/time for this activity record. Data type: String Format: YYYY-DD-MM hh:mm:ss</td>
</tr>
<tr>
<td>result.activities.icon</td>
<td>Name of the icon displayed on the tile for this activity record. Data type: String</td>
</tr>
<tr>
<td>result.activities.source_table_name</td>
<td>Name of the table in which this activity record is located. Determined by the Activity Source defined in the Activity Types [sn_actsub_activity_type] table for the type associated with this activity record. Data type: String</td>
</tr>
<tr>
<td>result.activities.subheader_fields</td>
<td>Array of objects in which each object represents a field displayed in the subheader of the tile for this activity record. Data type: Array</td>
</tr>
</tbody>
</table>

```
"subheader_fields": [ 
  { 
    "deeplink_to_subobject": "String",
    "display_as_timeago": "String",
    "label": "String",
    "show_label": "String",
    "type": "String",
    "value": "String"
  } 
]
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| result.activities.subheader_fields.deeplink_to_subobject | Flag indicating whether or not the subheader field displays as a hyperlink to the record for the object on which this activity was performed. Valid values:  
  • true: The subheader field displays as a hyperlink to the record for the object on which this activity was performed.  
  • false: The subheader field does not display as a hyperlink to the record for the object on which this activity was performed.  
  Data type: String |
| result.activities.subheader_fields.display_as_timeago | Flag indicating whether or not the subheader field displays in time ago format. Valid values:  
  • true: The subheader field displays in time ago format.  
  • false: The subheader field does not display in time ago format.  
  Data type: String |
| result.activities.subheader_fields.label | Label for the subheader field.  
  Data type: String |
<p>| result.activities.subheader_fields.show_label | Flag indicating whether or not the subheader field label displays for subheader field values in the tile for this activity record. Valid values: |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.activities.subheader_fields.type</td>
<td>Type of the subheader field. Data type: String</td>
</tr>
<tr>
<td>result.activities.subheader_fields.value</td>
<td>Value of the subheader field. Data type: String</td>
</tr>
<tr>
<td>result.activities.subobject_sys_id</td>
<td>Sys_id of the record for the object on which this activity was performed. Located in the table specified in the subobject_table_name parameter value. Data type: String</td>
</tr>
<tr>
<td>result.activities.subobject_table_name</td>
<td>Name of the table containing the record for the object on which this activity was performed. Determined by the Table Name defined in the Activity Group [sn_actsub_subscribable_object] table for the group containing the activity type associated with this activity record. Data type: String</td>
</tr>
<tr>
<td>result.activities.sys_id</td>
<td>Sys_id of this activity record. Located in the table specified as the source_table_name response body parameter value.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.activities.title</td>
<td>Title that displays on the tile for this activity record.</td>
</tr>
<tr>
<td>result.hasMoreRecords</td>
<td>Flag indicating whether or not the list of activities in the response body is truncated by the <code>last</code> query parameter value. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The list of activities in the response body is truncated by the <code>last</code> query parameter value. Repeating the request with a higher <code>last</code> query parameter value will return additional activities for this context and context instance.</td>
</tr>
<tr>
<td></td>
<td>• false: The list of activities in the response body is not truncated by the <code>last</code> query parameter value. No further activities are available for this context and context instance.</td>
</tr>
<tr>
<td>result.message</td>
<td>Error message describing the problem encountered during request processing.</td>
</tr>
<tr>
<td>result.status</td>
<td>HTTP status code returned for request.</td>
</tr>
<tr>
<td></td>
<td>Valid values: Refer to Status Codes.</td>
</tr>
<tr>
<td>result.stream</td>
<td>Sys_id specified in request <code>context</code> query parameter. Located in the Activity Context</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.user</td>
<td>Sys_id specified in request context_instance query parameter. Located in the appropriate table for the activity context. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

Request the second and third activity records for a specific Contact, limiting to data from 2020:

```bash
curl "https://instance.servicenow.com/api/now/actsub/activities?context=4a6a035a73d30010e37d71ef64f6a714&context_instance=ddce70866f933103b3c498f5d3ee417&stFrom=1&last=3&start_date=2020-01-01%2000:00:00&end_date=2020-12-31%2023:59:59" --request GET --header "Accept: application/json" --user "username:password"
```

The response body includes the two requested activity records for the specified Contact:

```
{
  "result": {
    "hasMoreRecords": true,
    "activities": [
      {
        "subobject_table_name": "sn_customerservice_case",
        "source_table_name": "sn_customerservice_case",
        "content_fields": [
          {
            "value": "Test",
            "show_label": "false",
            "type": "string",
            "label": "Short description",
            "display_as_timeago": "false",
            "deeplink_to_subobject": "false"
          }
        ]
      }
    ]
  }
}
```
Example: Python request

Request activity records from 2020 for a specific Contact, showing only records from the Contracts and Entitlements facets for the Contact activity context:
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/actsub/activities'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the query parameters
params = {
    'context_instance': 'ddce70866f9331003b3c498f5d3ee417',
    'context': '4a6a035a73d30010e37d71ef64f6a714',
    'facets': '0a63b6e673570010e37d71ef64f6a72b, d9d4fa2a73570010e37d71ef64f6a70a',
    'start_date': '2020-01-01 00:00:00',
    'end_date': '2020-12-31 23:59:59'
}

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers, params=params)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

The response body includes all 2020 Contracts and Entitlements activity records associated with the specified Contact:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <activities>
      <subobject_sys_id>451a446073181010e37d71ef64f6a703</subobject_sys_id>
      <sys_id>451a446073181010e37d71ef64f6a703</sys_id>
      <activity_type_id>fd2ff6d273d30010e37d71ef64f6a7be</activity_type_id>
    </activities>
  </result>
</response>
```
<subobject_table_name>service_entitlement</subobject_table_name>

<subheader_fields>
  <deeplink_to_subobject>true</deeplink_to_subobject>
  <display_as_timeago>false</display_as_timeago>
  <label>Name</label>
  <type>string</type>
  <show_label>false</show_label>
  <value>C+A5</value>
</subheader_fields>

<subheader_fields>
  <deeplink_to_subobject>false</deeplink_to_subobject>
  <display_as_timeago>true</display_as_timeago>
  <label>Created</label>
  <type>glide_date_time</type>
  <show_label>false</show_label>
  <value>2020-05-26 07:05:00</value>
</subheader_fields>

<subheader_fields>
  <deeplink_to_subobject>false</deeplink_to_subobject>
  <display_as_timeago>false</display_as_timeago>
  <label>Start Date</label>
  <type>glide_date</type>
  <show_label>false</show_label>
  <value>2020-05-26</value>
</subheader_fields>

<subheader_fields>
  <deeplink_to_subobject>false</deeplink_to_subobject>
  <display_as_timeago>false</display_as_timeago>
  <label>End Date</label>
  <type>glide_date</type>
  <show_label>false</show_label>
  <value>2023-05-26</value>
</subheader_fields>

<created>2020-05-26 07:05:00</created>

<icon>handshake_outline</icon>

<source_table_name>service_entitlement</source_table_name>

<title>MyCorp purchased an entitlement</title>

</activities>

<activities>
  <subobject_sys_id>7179cc2073181010e37d71ef64f6a7a0</subobject_sys_id>
  <sys_id>7179cc2073181010e37d71ef64f6a7a0</sys_id>
  <activity_type_id>fd2ff6d273d30010e37d71ef64f6a7be</activity_type_id>
  <subobject_table_name>service_entitlement</subobject_table_name>
  <subheader_fields>
<activities>
  <activity_type_id>328e72d273d30010e37d71ef64f6a726</activity_type_id>
  <subobject_table_name>ast_contract</subobject_table_name>
  <subheader_fields>
    <deeplink_to_subobject>true</deeplink_to_subobject>
    <display_as_timeago>false</display_as_timeago>
    <label>Name</label>
    <type>string</type>
    <show_label>false</show_label>
    <value>Customer Central - 24x7 Phone Contact</value>
  </subheader_fields>
  <subheader_fields>
    <deeplink_to_subobject>false</deeplink_to_subobject>
    <display_as_timeago>true</display_as_timeago>
    <label>Created</label>
    <type>glide_date_time</type>
    <show_label>false</show_label>
    <value>2020-05-26 07:00:00</value>
  </subheader_fields>
  <subheader_fields>
    <deeplink_to_subobject>false</deeplink_to_subobject>
    <display_as_timeago>false</display_as_timeago>
    <label>Start Date</label>
    <type>glide_date</type>
    <show_label>false</show_label>
    <value>2020-05-26</value>
  </subheader_fields>
  <subheader_fields>
    <deeplink_to_subobject>false</deeplink_to_subobject>
    <display_as_timeago>false</display_as_timeago>
    <label>End Date</label>
    <type>glide_date</type>
    <show_label>false</show_label>
    <value>2023-05-26</value>
  </subheader_fields>
  <created>2020-05-26 07:00:00</created>
</activities>
ActivitySubscriptions - GET /now/actsub/facets/{activity_context}/
{context_instance}

Retrieves facets configured for an activity context from the Subscriptions and Activity Feed Framework. Displays activity counts and data for each facet as determined by activity type, group, and context settings.

Users must have the actsub_user or admin role to access this endpoint. Data returned is limited by user access settings.

For more details on activity context facets, see Create facets for activity contexts.
URL format

Versioned URL: /api/now/{api_version}/actsub/facets/{activity_context}/
{context_instance}

Default URL: /api/now/actsub/facets/{activity_context}/
{context_instance}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>activity_context</td>
<td>Sys_id of an activity context. Located in the Activity Contexts [sn_actsub_activity_context] table. Predefined activity contexts include Consumer and Contact. Navigate to Customer Central &gt; Customer Activity &gt; Activity Contexts to view the full list of activity contexts defined for your instance. Data type: String</td>
</tr>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>context_instance</td>
<td>Sys_id of an instance of the specified activity context, representing the initiator of the activities you want to retrieve facets for. Located in the context table specified for the activity context record in the Activity Contexts [sn_actsub_activity_context] table. As an example, if you set activity_context to the sys_id of the Contact activity context, you should set this parameter to the sys_id of the Contact [customer_contact] record for the contact whose activity facets you want to retrieve. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>end_date</td>
<td>Date/time value. The request returns only records created during the time period defined by start_date and this parameter. Must be set along with start_date.</td>
</tr>
</tbody>
</table>
**Query parameters (continued)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Format: YYYY-MM-DD hh:mm:ss or YYYY-MM-DD</td>
</tr>
<tr>
<td>facets</td>
<td>Comma-separated list of sys_ids of activity facets to retrieve for the specified activity context. To see the activity facet types defined for an activity context, navigate to Customer Central &gt; Customer Activity &gt; Activity Contexts and examine the Facets related list. Data type: String Default: Return all activity facets configured for the activity context.</td>
</tr>
</tbody>
</table>
| get_activity_count | Flag indicating whether or not to include activity counts for each facet in the response body. Valid values:  
• true: Include facet activity counts.  
• false: Omit facet activity counts.  
Data type: Boolean Default: false                                                                                                                                                                                                                                               |
| lazy_load      | Flag indicating whether or not to improve performance by omitting facet activity data and activity counts from the response body. This parameter takes precedence over get_activity_count. Valid values:  
• true: Omit facet activity data and activity counts.  
• false: Retrieve facet activity data and activity counts.  
Data type: Boolean Default: false                                                                                                                                                                                                                                               |
| start_date     | Date/time value. The request returns only records created during the time period defined by this parameter and end_date. Must be set along with end_date. Data type: String Format: YYYY-MM-DD hh:mm:ss or YYYY-MM-DD                                                                 |
### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

#### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected. The response body <code>result.message</code> parameter describes the nature of the error. Example error messages:</td>
</tr>
</tbody>
</table>
## Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Make sure a valid activity context details are passed indicates that the <code>activity_context</code> path parameter value does not represent a valid activity context, the <code>context_instance</code> path parameter value does not represent a valid instance of the specified activity context, or both.</td>
<td></td>
</tr>
<tr>
<td>• Please provide <code>start_date</code> and <code>end_date</code> indicates that the request includes only one of the <code>start_date</code> and <code>end_date</code> query parameters, which cannot be used separately.</td>
<td></td>
</tr>
<tr>
<td>• <code>start_date</code> can not be ahead of <code>end_date</code> indicates that the specified <code>start_date</code> query parameter value occurs after the specified <code>end_date</code> query parameter value.</td>
<td></td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| result | Array of objects in which each object represents an activity facet.  
Data type: Array |

```
"result": [ 
  { 
    "activity_count": Number, 
    "children": [Array], 
    "facets": [Array], 
    "icon": "String", 
    "is_dynamic": "String", 
    "message": "String", 
    "name": "String", 
    "status": Number 
  } 
]
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.activity_count</td>
<td>Number of activity records included in this facet (and its children, if any). Only displayed when <code>lazy_load</code> query parameter is false and <code>get_activity_count</code> query parameter is true. Data type: Number</td>
</tr>
<tr>
<td>result.children</td>
<td>Array of objects in which each object represents an activity record (if this is a dynamic facet) or a child facet (if this is a static facet). Data type: Array</td>
</tr>
<tr>
<td></td>
<td>{ &quot;children&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{ &quot;activity_count&quot;: Number, &quot;facet&quot;: &quot;String&quot;, &quot;icon&quot;: &quot;String&quot;, &quot;title&quot;: &quot;String&quot; } ] }</td>
</tr>
<tr>
<td>result.children.activity_count</td>
<td>Number of activity records included in this child facet (and its children, if any). Only displayed when the <code>lazy_load</code> query parameter is false and the <code>get_activity_count</code> query parameter is true. Data type: Number</td>
</tr>
<tr>
<td>result.children.created</td>
<td>Creation date/time for this activity record. Data type: String Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>result.children.facet</td>
<td>Sys_id of this child facet record. Located in the Activity Facet [sn_actsub_facet] table. Data type: String</td>
</tr>
<tr>
<td>result.children.icon</td>
<td>Name of the icon associated with this child facet. Data type: String</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.children.sys_id</td>
<td>Sys_id of this activity record. Located in the appropriate table for the parent facet type. As an example, an activity record included in the Cases facet has a sys_id located in the Case [sn_customerservice_case] table. Data type: String</td>
</tr>
<tr>
<td>result.children.metadata</td>
<td>Array of objects in which each object includes activity record data for a facet field configured in the Activity Facet [sn_actsub_facet] table record for this facet type. Data type: Array</td>
</tr>
<tr>
<td>&quot;metadata&quot;: [</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;key&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;label&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>result.children.metadata.key</td>
<td>Name of this facet field. Data type: String</td>
</tr>
<tr>
<td>result.children.metadata.label</td>
<td>Label of this facet field. Data type: String</td>
</tr>
<tr>
<td>result.children.metadata.type</td>
<td>Type of this facet field. Data type: String</td>
</tr>
<tr>
<td>result.children.metadata.value</td>
<td>Value of this facet field. Data type: String</td>
</tr>
<tr>
<td>result.children.title</td>
<td>Name of this activity record or child facet. Data type: String</td>
</tr>
<tr>
<td>result.facets</td>
<td>Array of sys_ids associated with this facet record. Located in the Activity Facet [sn_actsub_facet] table. For a dynamic facet,</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>this array contains the sys_id of the facet itself. For a static facet, it instead contains the sys_ids of all facets which are children of the facet.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;facets&quot;: [</td>
</tr>
<tr>
<td></td>
<td>&quot;510d7e2e73570010e37d71ef64f6a70d&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;bc9bfaaa73570010e37d71ef64f6a771&quot; ]</td>
</tr>
<tr>
<td>result.icon</td>
<td>Name of the icon associated with this facet.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.is_dynamic</td>
<td>Flag indicating whether or not this facet is a dynamic facet. This value determines the type of information included in the children response body parameter. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: This is a dynamic facet. The children response body parameter value contains activity record details, and the facets parameter contains the dynamic facet sys_id.</td>
</tr>
<tr>
<td></td>
<td>• false: This is a static facet (a facet group). The children response body parameter value contains child facets, and the facets parameter contains the sys_ids of these child facets.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>result.message</td>
<td>Error message describing problem encountered during request processing.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.name</td>
<td>Name of the facet or facet group.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.status</td>
<td>HTTP status code returned for request.</td>
</tr>
<tr>
<td>Valid values:</td>
<td>See Status Code table.</td>
</tr>
</tbody>
</table>
Example: cURL request

Retrieve Chat, Phone, and Knowledge articles facets and activity counts for a Contact:

curl
  "https://instance.servicenow.com/api/now/actsub/facets/4a6a035a73d30010e37d71ef64f6a714/ddce70866f9331003b3c498f5d33ee417?get_activity_count=true&facets=510d7e2e73570010e37d71ef64f6a70d,bc9bfaaa73570010e37d71ef64f6a771,c72d7e2e73570010e37d71ef64f6a7b8" \
  --request GET \
  --header "Accept: application/json" \
  --user "username:password"

The response body includes the three requested facets as children of their facet groups, with activity counts displayed:

```json
{
  "result": [
    {
      "is_dynamic": false,
      "name": "Interactions",
      "activity_count": 4.0,
      "facets": [
        "510d7e2e73570010e37d71ef64f6a70d",
        "bc9bfaaa73570010e37d71ef64f6a771"
      ],
      "children": [
        {
          "facet": "510d7e2e73570010e37d71ef64f6a70d",
          "icon": "chat_outline",
          "activity_count": 2.0,
          "title": "Chat"
        },
        {
          "facet": "bc9bfaaa73570010e37d71ef64f6a771",
          "icon": "phone_outline",
          "activity_count": 2.0,
          "title": "Phone"
        }
      ]
    }
  ]
}
```
Example: Python request

Retrieve facets and activity counts for activity records associated with a Contact and created in 2020:

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/actsub/facets/4a6a035a73d30010e37d71ef64f6a714/dec70866f9331003b3c498f5d3e417'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the query parameters
params = {
    'start_date': '2020-01-01 00:00:00',
    'end_date': '2020-12-31 23:59:59',
    'get_activity_count': 'true'
}

# Set the proper headers
```
headers = { 'Accept':'application/xml' }

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers, params=params)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status: ', response.status_code, 'Headers: ', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

The response body includes activity facets and facet groups for the specified Contact, with activity counts displayed for each facet and facet group:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <children>
      <sys_id>095c17868d059010f87705a074e52a78</sys_id>
      <metadata>
        <label>Number</label>
        <type>string</type>
        <value>CS0001010</value>
        <key>number</key>
      </metadata>
      <metadata>
        <label>Created</label>
        <type>glide_date_time</type>
        <value>2020-05-29 14:23:11</value>
        <key>sys_created_on</key>
      </metadata>
      <metadata>
        <label>Active escalation</label>
        <type>reference</type>
        <value/>
        <key>active_escalation</key>
      </metadata>
      <created>2020-05-29 21:23:11</created>
      <title>Test</title>
    </children>
  </result>
</response>
```
<children>
  <sys_id>d92d88e073181010e37d71ef64f6a7e3</sys_id>
  <metadata>
    <label>Number</label>
    <type>string</type>
    <value>CS0001085</value>
    <key>number</key>
  </metadata>
  <metadata>
    <label>Created</label>
    <type>glide_date_time</type>
    <value>2020-05-25 08:00:00</value>
    <key>sys_created_on</key>
  </metadata>
  <metadata>
    <label>Active escalation</label>
    <type>reference</type>
    <value/>
    <key>active_escalation</key>
  </metadata>
  <created>2020-05-25 15:00:00</created>
  <title>Router installation incomplete</title>
</children>

<children>
  <sys_id>e6dd842473181010e37d71ef64f6a716</sys_id>
  <metadata>
    <label>Number</label>
    <type>string</type>
    <value>CS0001086</value>
    <key>number</key>
  </metadata>
  <metadata>
    <label>Created</label>
    <type>glide_date_time</type>
    <value>2020-05-24 03:00:00</value>
    <key>sys_created_on</key>
  </metadata>
  <metadata>
    <label>Active escalation</label>
    <type>reference</type>
    <value/>
    <key>active_escalation</key>
  </metadata>
</children>
<type>reference</type>
<value/>
</metadata>
</children>
</children>

<children>
<sys_id>9e6ecc2473181010e37d71ef64f6a796</sys_id>
<metadata>
  <label>Number</label>
  <type>string</type>
  <value>CS0001087</value>
  <key>number</key>
</metadata>
<metadata>
  <label>Created</label>
  <type>glide_date_time</type>
  <value>2020-05-23 06:00:00</value>
  <key>sys_created_on</key>
</metadata>
<metadata>
  <label>Active escalation</label>
  <type>reference</type>
  <value>ESC0001001</value>
  <key>active_escalation</key>
</metadata>
<created>2020-05-23 13:00:00</created>
<title>Loud noise from router</title>
</children>
</children>
<children>
<sys_id>e63fc86473181010e37d71ef64f6a7d9</sys_id>
<metadata>
  <label>Number</label>
  <type>string</type>
  <value>CS0001088</value>
  <key>number</key>
</metadata>
<metadata>
  <label>Created</label>
  <type>glide_date_time</type>
  <value>2020-05-21 06:00:00</value>
  <key>sys_created_on</key>
</metadata>
<created>2020-05-21 13:00:00</created>
<title>Router disconnects often</title>
</children>
<metadata>
  <label>Active escalation</label>
  <type>reference</type>
  <value/>
  <key>active_escalation</key>
</metadata>
<created>2020-05-21 13:00:00</created>
<title>Upgrade to new router not successful</title>
</children>

<children>
  <sys_id>baef84a473181010e37d71ef64f6a734</sys_id>
  <metadata>
    <label>Number</label>
    <type>string</type>
    <value>CS0001089</value>
    <key>number</key>
  </metadata>
  <metadata>
    <label>Created</label>
    <type>glide_date_time</type>
    <value>2020-05-20 03:00:00</value>
    <key>sys_created_on</key>
  </metadata>
  <metadata>
    <label>Active escalation</label>
    <type>reference</type>
    <value/>
    <key>active_escalation</key>
  </metadata>
  <created>2020-05-20 10:00:00</created>
  <title>Advanced router setting not working</title>
</children>

<children>
  <sys_id>63405ca473181010e37d71ef64f6a7e8</sys_id>
  <metadata>
    <label>Number</label>
    <type>string</type>
    <value>CS0001090</value>
    <key>number</key>
  </metadata>
  <metadata>
    <label>Created</label>
    <type>glide_date_time</type>
    <value>2020-05-19 05:00:00</value>
    <key>sys_created_on</key>
  </metadata>
  <metadata>
    <label>Active escalation</label>
    <type>reference</type>
    <value/>
    <key>active_escalation</key>
  </metadata>
</children>
<children>
  <children>
    <sys_id>e1a014e473181010e37d71ef64f6a7c2</sys_id>
    <metadata>
      <label>Number</label>
      <type>string</type>
      <value>CS0001091</value>
      <key>number</key>
    </metadata>
    <metadata>
      <label>Created</label>
      <type>glide_date_time</type>
      <value>2020-05-18 06:00:00</value>
      <key>sys_created_on</key>
    </metadata>
    <metadata>
      <label>Active escalation</label>
      <type>reference</type>
      <value/>
      <key>active_escalation</key>
    </metadata>
    <created>2020-05-19 12:00:00</created>
    <title>Wireless setting keeps changing</title>
  </children>
  <children>
    <sys_id>00215ce473181010e37d71ef64f6a7a2</sys_id>
    <metadata>
      <label>Number</label>
      <type>string</type>
      <value>CS0001092</value>
      <key>number</key>
    </metadata>
    <metadata>
      <label>Created</label>
      <type>glide_date_time</type>
      <value>2020-05-18 13:00:00</value>
      <key>sys_created_on</key>
    </metadata>
    <metadata>
      <label>Active escalation</label>
      <type>reference</type>
      <value/>
      <key>active_escalation</key>
    </metadata>
    <created>2020-05-19 12:00:00</created>
    <title>Router range not as expected</title>
  </children>
</children>
<children>
  <sys_id>fa81542873181010e37d71ef64f6a7dc</sys_id>
  <metadata>
    <label>Number</label>
    <type>string</type>
    <value>CS0001093</value>
    <key>number</key>
  </metadata>
  <metadata>
    <label>Created</label>
    <type>glide_date_time</type>
    <value>2020-05-16 03:00:00</value>
    <key>sys_created_on</key>
  </metadata>
  <metadata>
    <label>Active escalation</label>
    <type>reference</type>
    <value/>
    <key>active_escalation</key>
  </metadata>
  <created>2020-05-16 10:00:00</created>
  <title>Power supply to router not stable</title>
</children>

<activity_count>16.0</activity_count>
<name>Cases</name>
<is_dynamic>true</is_dynamic>
<icon>clipboard_outline</icon>
<facets>19e9935273170010e37d71ef64f6a72b</facets>
<activity_count>2.0</activity_count>
<title>Chat</title>
<facets>510d7e2e73570010e37d71ef64f6a70d</facets>
</children>
</children>
<activity_count>2.0</activity_count>
<title>Phone</title>
<facets>bc9bfaaa73570010e37d71ef64f6a771</facets>
</children>
<activity_count>4.0</activity_count>
</children>
</result>
<result>
<children>
<activity_count>5.0</activity_count>
<title>Knowledge articles</title>
<facets>c72d7e2e73570010e37d71ef64f6a7b8</facets>
</children>
<activity_count>5.0</activity_count>
</children>
</result>
<result>
<children>
<sys_id>27f2c82c73d41010e37d71ef64f6a7bf</sys_id>
<metadata>
<label>Start date</label>
<type>glide_date</type>
/value>2020-05-26</value>
</metadata>
<created>2020-05-26 05:01:25</created>
<title>Boxeo - Customer Central Demo Contract</title>
</children>
<activity_count>1.0</activity_count>
</result>
<result>
<children>
<activity_count>1.0</activity_count>
</children>
Agent Client Collector API

The Agent Client Collector enables managing actions on available agents and managing policies.

This API requires the Agent Client Collector Framework (sn_agent) store application and is provided within the sn_agent namespace. The endpoints in this API require the agent_client_collector_admin role. For more information, refer to Agent Client Collector.

Agent management endpoints
• Get extensive information of one or more agents. These endpoints are also accessible to users with the agent_client_collector_user role.
  ◦ GET /agents/{agent_id}
  ◦ GET /agents/list

• Submit a request to grab an agent log and retrieving information about the request progress.
  ◦ GET /agents/{agent_id}/log
  ◦ GET /agents/log/{request_id}/

• Disable or enable agent data collection.
  ◦ GET /agents/{agent_id}/data/off
  ◦ GET /agents/{agent_id}/data/on

• Restart an agent with GET /agents/{agent_id}/restart.

• Run discovery on an agent with GET /agents/{agent_id}/discovery.

For information on performing similar tasks in a script include, see AccAgentsAPI.

Policy management and workflow

Use policy management APIs to view details, activate/deactivate a policy, update a policy, and publish a policy.

To update a policy:

1. Get a list of policies and details with GET /agents/policies/list. This endpoint requires the agent_client_collector_user role.
   • To update a policy in the Draft state, use the sys_ids retrieved from the policies list in the update endpoints.
   • To update a policy in Published or Published* state, get an editable sandbox copy with GET /agents/policy/sandbox_from_published/{policy_id}. Use the sys_ids from this response to modify properties using an update endpoint.

   • POST /agents/update/policy/{policy_id}
   • POST /agents/update/check/{check_id}
• POST /agents/update/check_param/{param_id}
• POST /agents/update/check_secure_param/{param_id}

3. Publish the policy using GET /agents/policy/publish/{policy_id}.

Once published, the policy becomes Active. This API also includes endpoints for activating or deactivating a published policy:
• GET /agents/policy/activate/{policy_id}
• GET /agents/policy/deactivate/{policy_id}

Agent Client Collector - GET /agents/{agent_id}
Gets the information of a specified agent.

**URL format**
/api/sn_agent/agents/{agent_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent_id</td>
<td>Unique ID of an agent listed in the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table. For a list of agent IDs and other details, run the GET /agents/list endpoint. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_user role.</td>
</tr>
<tr>
<td>404</td>
<td>Agent with provided ID not found.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Array&gt;</td>
<td>Array of JSON objects containing extended agent information.</td>
</tr>
</tbody>
</table>

```json
{
  "agent_id": "String",
  "data_collection": Number,
  "ip_address": "String",
  "is_duplicate": Boolean,
  "is_restart_enabled": Boolean,
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td><code>&quot;name&quot;: &quot;String&quot;,</code></td>
</tr>
<tr>
<td></td>
<td><code>&quot;number_of_running_checks&quot;: Number,</code></td>
</tr>
<tr>
<td></td>
<td><code>&quot;status&quot;: Number,</code></td>
</tr>
<tr>
<td></td>
<td><code>&quot;up_since&quot;: &quot;String&quot;,</code></td>
</tr>
<tr>
<td></td>
<td><code>&quot;version&quot;: &quot;String&quot;</code></td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>agent_id</td>
<td>ID of the agent as submitted.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>data_collection</td>
<td>Data collection indicates whether scheduled checks are to be run. These checks are a part of the policies scheduled for this agent to run. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: On – Checks run as scheduled.</td>
</tr>
<tr>
<td></td>
<td>• 1: Off (manual) – Checks have been disabled manually.</td>
</tr>
<tr>
<td></td>
<td>• 2: Off (auto) – Checks have been disabled automatically due to high CPU consumption by the</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>ip_address</td>
<td>Agent IP address.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>is_duplicate</td>
<td>Flag that indicates whether this agent is a duplicate of another. There should be only a single agent on a given host.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: The agent has the same host as an Alive/Up agent with a different agent ID. Turn off or uninstall the duplicate</td>
</tr>
<tr>
<td></td>
<td>• false: This agent has no duplicates in the Alive/Up state.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>is_restart_enabled</td>
<td>Flag that indicates whether restart is enabled. Agent restart is not configurable. It depends on</td>
</tr>
</tbody>
</table>
### Name | Description
--- | ---
**the OS and the version of the OS the agent is running on.**
Possible values:
- **true**: Restart is enabled for this agent.
- **false**: Restart is disabled for this agent.
Data type: Boolean

**name** | Name of the agent.
Data type: String

**number_of_running_checks** | The number of checks the agent is scheduled to run. These checks are a part of the policies scheduled for this agent to run.
Data type: Number

**status** | Status of the agent.
Possible values:
- **0**: Alive/Up – The agent is active.
- **1**: Warning – The agent has not received a keep-alive message in the past few minutes.
- **2**: Down – The agent has not received a keep-alive message in a long time.
- **3**: Restarting – The agent is restarting.
Data type: Number

**up_since** | UTC time since the agent’s status became alive/up. The value is in GlideDateTime format.
Data type: String

**version** | Version of Agent Client Collector the agent is running.
Data type: String

---

**Example: cURL request**
The following example shows how to get agent details.

```
curl "https://instance.service-now.com/api/sn_agent/agents/<agent_id>" \
--request GET \
```
--header "Accept:application/json" \n--user 'username':'password'

Output:

```
{
    "name": "WIN-V26KAP?PI2G",
    "status": 2,
    "agent_id": "074b14e2eb3ce9d4",
    "ip_address": "10.196.55.14",
    "number_of_running_checks": 11,
    "data_collection": 0,
    "is_restart_enabled": true,
    "is_duplicate": false,
    "up_since": "2021-03-31 12:02:17",
    "version": "2.3.0"
}
```

**Agent Client Collector - GET /agents/{agent_id}/data/off**

Disables data collection for a specified agent in the up/alive state.

To determine if an agent’s data collection is on or off, run the GET /agents/{agent_id} endpoint.

**URL format**

/api/sn_agent/agents/{agent_id}/data/off

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent_id</td>
<td>Unique ID of an agent listed in the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table. For a list of agent IDs and other details, run the GET /agents/list endpoint. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_admin role.</td>
</tr>
<tr>
<td>404</td>
<td>Agent not found or is not in the up/alive state.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Message containing operation success or failure results.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

The following example shows how to turn off agent data collection.

curl "https://instance.service-now.com/api/sn_agent/agents/<agent_id>/data/off" \
--request GET \
--header "Accept:application/json" \
--user 'username':'password'

Output:

{
  "message": "Data Collection Disabled For Agent With ID: <agent_id>"
}

Agent Client Collector - GET /agents/{agent_id}/data/on

Enables data collection for a specified agent in the up/alive state.

To determine if an agent’s data collection is on or off, run the GET /agents/ {agent_id} endpoint.

URL format

/api/sn_agent/agents/{agent_id}/data/on

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>agent_id</td>
</tr>
<tr>
<td>Data type: String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>
**Request body parameters (JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the <code>agent_client_collector_admin</code> role.</td>
</tr>
<tr>
<td>404</td>
<td>Agent not found or is not in the up/alive state.</td>
</tr>
</tbody>
</table>

**Response body parameters (JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Message containing operation success or failure results.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request
The following example shows how to turn on agent data collection.

```
curl "https://instance.service-now.com/api/sn_agent/agents/<agent_id>/data/on" \
--request GET \
--header "Accept:application/json" \
--user 'username':'password'
```

Output:

```
{
  "message": "Data Collection Enabled For Agent With ID: <agent_id>"
}
```

Agent Client Collector - GET /agents/{agent_id}/discovery
Runs a discovery check to locate CIs related to an agent. The specified agent
must be in alive/up status.

URL format
/api/sn_agent/agents/{agent_id}/discovery

Supported request parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent_id</td>
<td>Unique ID of an agent listed in the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table. For a list of agent IDs and other details, run the GET /agents/list endpoint. Data type: String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_admin role.</td>
</tr>
<tr>
<td>404</td>
<td>Agent not found or is not in the up/alive state.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Message containing operation success or failure results. Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example shows how to run discovery on an agent with alive/up status.
curl "https://instance.service-now.com/api/sn_agent/agents/<agent_id>/discovery" \
  --request GET \
  --header "Accept:application/json" \
  --user 'username':'password'

Output:

{
  "message": "Running Discovery For Agent With ID: <agent_id>"
}

Agent Client Collector - GET /agents/check_defs/{check_def_id}
Gets a specified check definition with details.

URL format
@api/sn_agent/agents/check_defs/{check_def_id}

Supported request parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check_def_id</td>
<td>Sys_id of the check definition listed in the Check Definitions [sn_agent_check_def] table.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-Include-Check-Params</td>
<td>Flag that indicates whether existing check parameter details are returned. Information for each standard and secure check parameter is included in a JSON object. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Return check parameter details.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not return check parameter details.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
|          | Data type: Boolean                  
|          | Default: False                       |

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>404</td>
<td>Check definition was not found with the sys_id provided.</td>
</tr>
</tbody>
</table>

**Response body parameters (JSON)**

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check</td>
<td>Details of the specified check definition.</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;background&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>Properties</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
|                | "check_group": "String",
|                | "check_type": "String",
|                | "command": "String",
|                | "error": "String",
|                | "name": "String",
|                | "params": [Array],
|                | "plugins": [Array],
|                | "proxy_valid": Boolean,
|                | "secure_params": [Array],
|                | "sys_id": "String",
|                | "timeout": Number                                                                                                                                  |

**background**  
Flag that indicates whether this check definition is a background check. A background check is a check which the agent starts execution of and doesn't wait for it to finish running.  
Valid values:  
- **true**: This check definition is a background check.  
- **false**: This check definition is not a background check.  
  Data type: Boolean

**check_group**  
Group specified for this check definition.  
Data type: String

**check_type**  
Type of check.  
Possible values:  
- **Events** – Check results are transformed into an Event Management event.  
- **Metrics** – Values from the check result are transformed to metrics.  
  Data type: String

**command**  
Command that the Agent Client Collector executes.  
Data type: String

**error**  
Message if there is an error. Null otherwise.  
Data type: String

**name**  
Name of the check.
<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>params</td>
<td>List of parameter definitions associated with the check definition. These results are only included if the <code>withParams</code> parameter is set to true.</td>
</tr>
<tr>
<td></td>
<td>&quot;params&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;active&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;default_value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;mandatory&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>params.active</td>
<td>Flag that indicates whether the check parameter is active. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The check parameter is active.</td>
</tr>
<tr>
<td></td>
<td>• false: The check parameter is inactive.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>params.default_value</td>
<td>Specifies the default value for this check parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>params.mandatory</td>
<td>Flag that indicates whether the check parameter is required. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The check parameter is required.</td>
</tr>
<tr>
<td></td>
<td>• false: The check parameter is optional.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>params.name</td>
<td>Name of the check parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>params.sys_id</td>
<td>Sys_id of the check parameter listed in the Check Secure Parameter Definitions [sn_agent_check_param_def] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Properties</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>plugins</td>
<td>List of [Agent Client Collector plugins](https:// servicenow.com) associated with this check.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>proxy_valid</td>
<td>Flag that indicates whether the check definition policy is set to work as a proxy.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: This check definition policy is set to work as a proxy.</td>
</tr>
<tr>
<td></td>
<td>• false: This check definition policy is not set to work as a proxy.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>secure_params</td>
<td>List of [secure parameters](https:// servicenow.com) assigned to this check. These results are only included if the <code>withParams</code> parameter is set to true.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;secure_params&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;active&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;order&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>secure_params.active</td>
<td>Flag that indicates whether the secure parameter is active.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The secure parameter is active.</td>
</tr>
<tr>
<td></td>
<td>• false: The secure parameter is inactive.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>secure_params.name</td>
<td>Name of the secure parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>secure_params.order</td>
<td>Order in which the parameter is sent to the check command/script.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>Properties</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>secure_params.sys_id</td>
<td>Sys_id of the secure parameter listed in the Check Secure Parameter Definitions [sn_agent_check_secure_param_def] table. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the check definition listed in the Check Definitions [sn_agent_check_def] table. Data type: String</td>
</tr>
<tr>
<td>timeout</td>
<td>Timeout in seconds. Data type: Number</td>
</tr>
</tbody>
</table>

**Example: cURL request**

The following example shows how to get information for a specified check definition.

```bash
curl
"https://instance.service-now.com/api/sn_agent/agents/check_defs/94436b237f705300f128134f8dfa91a4" \
--request GET \
--header "Accept:application/json" \
--user 'username':'password'
```

**Output:**

```json
{}
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
"mandatory": true,
"default_value": "80",
"sys_id": "58436b237f705300f128134f8d9a18"
},
{
"name": "path",
"active": true,
"mandatory": true,
"default_value": "/server-status?auto",
"sys_id": "98436b237f705300f128134f8d9a19a"
},
{
"name": "scheme",
"active": false,
"mandatory": false,
"default_value": null,
"sys_id": "a4e57a96db3bb4035305c55dc9619f6"
},
{
"name": "host",
"active": true,
"mandatory": true,
"default_value": "127.0.0.1",
"sys_id": "d4436b237f705300f128134f8d9a16"
},
{
"name": "ssl_secure_connection",
"active": false,
"mandatory": false,
"default_value": null,
"sys_id": "e3b272c4530100106ffeeddeeff7b1275"
}
"secure_params": [
{
"name": "cred_user_name",
"active": true,
"order": 1,
"sys_id": "2494cd6e53170010f42cddeeff7b1273"
},
{
"name": "cred_password",
"active": true,
"order": 2,
Agent Client Collector - GET /agents/check_defs/list

Gets a list of check definitions with details.

URL format

/api/sn_agent/agents/check_defs/list

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-Enc-Query</td>
<td>An <em>encoded query string</em> to filter the check definition result list. Use null for an unfiltered list of check definitions in the system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>X-Include-Check-Params</td>
<td>Flag that indicates whether existing check parameter details are returned. Information for each standard and secure check parameter is included in a JSON object.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Return check parameter details.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not return check parameter details.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: False</td>
</tr>
</tbody>
</table>
Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-Limit</td>
<td>Limits the number of returned records. Set to null to use the default value. Data type: Number Default: 20,000</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>

**Response body parameters (JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check definitions</td>
<td>List of check definition and details provided as JSON objects.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| `background` | Flag that indicates whether this check definition is a background check. A background check is a check which the agent starts execution of and doesn’t wait for it to finish running. Valid values:  
  • true: This check definition is a background check.  
  • false: This check definition is not a background check.  
  Data type: Boolean |
| `check_group`| Group specified for this check definition.  
  Data type: String |
| `check_type` | Type of check. Possible values:  
  • Events – Check results are transformed into an Event Management event.  
  • Metrics – Values from the check result are transformed to metrics.  
  Data type: String |
| `command`    | Command that the Agent Client Collector executes. |

```json
[
  {
    "background": Boolean,
    "check_group": "String",
    "check_type": "String",
    "command": "String",
    "name": "String",
    "params": [Array],
    "plugins": [Array],
    "proxy_valid": Boolean,
    "secure_params": [Array],
    "sys_id": "String",
    "timeout": Number
  }
]
```

Data type: Array
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the check.</td>
</tr>
<tr>
<td>params</td>
<td>List of parameter definitions associated with the check definition. These results are only included if the <code>withParams</code> parameter is set to true.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>```json</td>
<td>&quot;params&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;active&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;default_value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;mandatory&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>```</td>
<td></td>
</tr>
<tr>
<td>params.active</td>
<td>Flag that indicates whether the check parameter is active.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The check parameter is active.</td>
</tr>
<tr>
<td></td>
<td>• false: The check parameter is inactive.</td>
</tr>
<tr>
<td>params.default_value</td>
<td>Specifies the default value for this check parameter.</td>
</tr>
<tr>
<td>params.mandatory</td>
<td>Flag that indicates whether the check parameter is required.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The check parameter is required.</td>
</tr>
<tr>
<td></td>
<td>• false: The check parameter is optional.</td>
</tr>
<tr>
<td>params.name</td>
<td>Name of the check parameter.</td>
</tr>
</tbody>
</table>

Data type: String

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>params.sys_id</td>
<td>Sys_id of the check parameter listed in the Check Secure Parameter Definitions [sn_agent_check_param_def] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>plugins</td>
<td>List of Agent Client Collector plugins associated with this check.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>proxy_valid</td>
<td>Flag that indicates whether the check definition policy is set to work as a proxy.</td>
</tr>
<tr>
<td></td>
<td>Valid values: \n  • <code>true</code>: This check definition policy is set to work as a proxy.</td>
</tr>
<tr>
<td></td>
<td>• <code>false</code>: This check definition policy is not set to work as a proxy.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>secure_params</td>
<td>List of secure parameters assigned to this check. These results are only included if the withParams parameter is set to true.</td>
</tr>
</tbody>
</table>
|                  | `{"secure_params": [  
|                  |   {  
|                  |     "active": Boolean,  
|                  |     "name": "String",  
|                  |     "order": Number,  
|                  |     "sys_id": "String"  
|                  |   }  
|                  | ]}`                                                                                                                                                                                                       |
|                  | Data type: Array                                                                                                                                                                                          |
| secure_params.active | Flag that indicates whether the secure parameter is active.                                                                                                                                                |
|                  | Valid values: \n  • `true`: The secure parameter is active.                                                                                                                                                 |
|                  |  • `false`: The secure parameter is inactive.                                                                                                                                                             |
|                  | Data type: Boolean                                                                                                                                                                                          |
| secure_params.name | Name of the secure parameter.                                                                                                                                                                             |
### Name | Description
---|---
secure_params.order | Order in which the parameter is sent to the check command/script. 
| Data type: Number 
secure_params.sys_id | Sys_id of the secure parameter listed in the Check Secure Parameter Definitions [sn_agent_check_secure_param_def] table. 
| Data type: String 
sys_id | Sys_id of the check definition listed in the Check Definitions [sn_agent_check_def] table. 
| Data type: String 
timeout | Timeout in seconds. 
| Data type: Number

**Example: cURL request**
The following example shows how to retrieve a list of two check definitions with parameter values.

```bash
curl "https://instance.service-now.com/api/sn_agent/agents/check_defs/list" \
--request GET \
--header "Accept:application/json" \
--user 'username':'password'
```

**Output:**
```
{
  "check_definitions": [
    {
      "name": "checks_api_test",
      "command": "echo hello",
      "plugins": [],
      "timeout": 9,
      "proxy_valid": true,
      "background": false,
      "check_type": "TestCheck",
      "check_group": "computer",
      "sys_id": "7f1f9026dba530106f4810284b96194f",
      "params": [],
      "secure_params": [
```
Agent Client Collector - GET /agents/exec/background/stop/{request_id}

Stops a background check.

To start a background check, use the POST /agents/check_defs/{check_def_id}/run API.
**URL format**

/api/sn_agent/agents/exec/background/stop/{request_id}

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_id</td>
<td>The ID of a background check request generated by running the POST /agents/check_defs/{check_def_id}/run API.</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>404</td>
<td>The request with the ID provided is not found.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Example: cURL request

The following example shows how to stop a background check.

```
curl
   "https://instance.service-now.com/api/sn_agent/agents/exec/background/stop/02359174db2a30108a0751ff3961997"
   --request GET
   --header "Accept:application/json"
   --user 'username':'password'
```

Agent Client Collector - GET /agents/exec/run/{request_id}

Gets status of the request with the given ID.

**URL format**

/api/sn_agent/agents/exec/run/{request_id}

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_id</td>
<td>The ID of a background check request generated by running the POST /agents/check_defs/{check_def_id}/run API.</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>202</td>
<td>Message with provided ID indicating that the request is in progress.</td>
</tr>
<tr>
<td>400</td>
<td>Error in the arguments provided in the request body.</td>
</tr>
<tr>
<td>404</td>
<td>The request with the ID provided is not found.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>408</td>
<td>Execution timeout for the request with the ID provided.</td>
</tr>
<tr>
<td>500</td>
<td>Error checking the status or the request with the ID provided.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>Request status. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• done – Check is successful.</td>
</tr>
<tr>
<td></td>
<td>• failure – Check has failed. See error message for details.</td>
</tr>
<tr>
<td></td>
<td>• mid_flow – Request output is being handled by the MID server.</td>
</tr>
<tr>
<td></td>
<td>• processing – Check is in progress.</td>
</tr>
<tr>
<td></td>
<td>• timeout – Check processing exceeded time limit set in the runCheckForCis() method.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>err_msg</td>
<td>Error message if any. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• No agents found for relevant CIs.</td>
</tr>
<tr>
<td></td>
<td>• No background check request with given ID.</td>
</tr>
<tr>
<td></td>
<td>• No request with given ID.</td>
</tr>
<tr>
<td></td>
<td>• No test result with given ID.</td>
</tr>
<tr>
<td></td>
<td>• Request timeout.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Example: cURL request

The following example shows how to get the status of a request.

```
curl
"https://instance.service-now.com/api/sn_agent/agents/exec/run/12fed13cdb2a30108a0751f4f3961981" \ 
--request GET 
```
Agent Client Collector - GET /agents/exec/test/{test_result_id}

Gets the test check status of the given test result.

URL format

/api/sn_agent/agents/exec/test/{test_result_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>test_result_id</td>
<td>A test result ID generated by creating a test check request.</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>202</td>
<td>Message with provided ID indicating that the request is in progress.</td>
</tr>
<tr>
<td>404</td>
<td>The request with the ID provided is not found.</td>
</tr>
<tr>
<td>408</td>
<td>Execution timeout for the request with the ID provided.</td>
</tr>
<tr>
<td>500</td>
<td>Error checking the status of the request with the ID provided.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
</table>
| status     | Status of the test results. Possible values:  
• 0: Pending  
• 1: In progress  
• 2: Complete  
• 3: No test result with given ID  
Data type: String |
| output     | Output describing the status.  
Data type: String |

Example: cURL request

The following example shows how get result status of a completed test check request.
curl
  "https://instance.service-now.com/api/sn_agent/agents/check_instances/99e12466dba530106f4810284b961976/test"
  --request POST
  --header "Accept:application/json"
  --user 'username': 'password'

Agent Client Collector - GET /agents/list

Gets a list of agents with related information.

URL format
/api/sn_agent/agents/list

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>X-Limit</td>
<td>Restricts results to a maximum number of agents. Use null or undefined for both if they are not required. Default/Max: 20,000. Data type: Number</td>
</tr>
</tbody>
</table>

### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_user role.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Array&gt;</td>
<td>Array of JSON objects containing extended agent information.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>[</td>
</tr>
<tr>
<td></td>
<td>&quot;agent_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;data_collection&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>agent_id</td>
<td>ID of the agent as submitted.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>data_collection</td>
<td>Data collection indicates whether scheduled checks are to be run. These checks are a part of the policies scheduled for this agent to run. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: On – Checks run as scheduled.</td>
</tr>
<tr>
<td></td>
<td>• 1: Off (manual) – Checks have been disabled manually.</td>
</tr>
<tr>
<td></td>
<td>• 2: Off (auto) – Checks have been disabled automatically due to high CPU consumption by the agent.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>ip_address</td>
<td>Agent IP address.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>is_duplicate</td>
<td>Flag that indicates whether this agent is a duplicate of another. There should be only a single agent on a given host. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: The agent has the same host as an Alive/Up agent with a different agent ID. Turn off or uninstall the duplicate</td>
</tr>
<tr>
<td></td>
<td>• false: This agent has no duplicates in the Alive/Up state.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| is_restart_enabled    | Flag that indicates whether restart is enabled. Agent restart is not configurable. It depends on the OS and the version of the OS the agent is running on. Possible values:  
  - true: Restart is enabled for this agent.  
  - false: Restart is disabled for this agent.  
  Data type: Boolean |
| name                  | Name of the agent.  
  Data type: String |
| number_of_running     | The number of checks the agent is scheduled to run. These checks are a part of the policies scheduled for this agent to run.  
  Data type: Number |
| status                | Status of the agent. Possible values:  
  - 0: Alive/Up – The agent is active.  
  - 1: Warning – The agent has not received a keep-alive message in the past few minutes.  
  - 2: Down – The agent has not received a keep-alive message in a long time.  
  - 3: Restarting – The agent is restarting.  
  Data type: Number |
| up_since              | UTC time since the agent’s status became alive/up. The value is in GlideDateTime format.  
  Data type: String |
| version               | Version of Agent Client Collector the agent is running.  
  Data type: String |

**Example: cURL request**

The following example shows how to restrict results by query and number. The query returns all agents that are not in the down state with a maximum of two results.
curl "https://instance.service-now.com/api/sn_agent/agents/list" \
  --request GET \
  --header "Accept:application/json" \
  --header "X-Enc-Query: agent_extended_info.status!=2" \
  --header "X-Limit: 2" \
  --user 'username':'password'

Output:

```
{
  "agents": [
    {
      "name": "007-175",
      "status": 0,
      "agent_id": "007-175",
      "ip_address": "11.222.63.66",
      "number_of_runningChecks": 0,
      "data_collection": 0,
      "is_restart_enabled": false,
      "is_duplicate": false,
      "up_since": "2021-03-24 14:36:45",
      "version": "2.4.0"
    },
    {
      "name": "win2016-dc-64bit",
      "status": 0,
      "agent_id": "007-64",
      "ip_address": "10.222.333.42",
      "number_of_runningChecks": 1,
      "data_collection": 0,
      "is_restart_enabled": true,
      "is_duplicate": false,
      "up_since": "2021-03-24 11:04:38",
      "version": "2.4.0"
    }
  ]
}
```

Example: cURL request

The following example shows how to list every agent in the system. This example uses no query and no maximum number of results.

```
curl "https://instance.service-now.com/api/sn_agent/agents/list" \
  --request GET \
```
Agent Client Collector - GET /agents/{agent_id}/log

Requests the log of a specified agent with alive/up status.

ℹ️ **Note:** To retrieve the log and check its progress, pass the request ID returned to the GET /agents/log/{request_id}/ endpoint.

**URL format**

/api/sn_agent/agents/{agent_id}/log

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent_id</td>
<td>Unique ID of an agent listed in the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table. For a list of agent IDs and other details, run the GET /agents/list endpoint. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_admin role.</td>
</tr>
<tr>
<td>404</td>
<td>Agent not found or is not in the up/alive state.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_id</td>
<td>Sys_id of a request in the Agent Client Collector Requests [sn_agent_request] table.</td>
</tr>
<tr>
<td></td>
<td>You can use this ID to retrieve the log and check its progress with the GET /agents/log/{request_id}/ endpoint.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example shows how to get a log request ID.

```bash
curl "https://instance.service-now.com/api/sn_agent/agents/<sys_id>/log" \
--request GET \
--header "Accept:application/json" \
--user 'username':'password'
```
Output:

```
"request_id": "<sys_id>"
```

**Agent Client Collector - GET /agents/log/{request_id}/**

Checks the status of a grab log request.

Detects changes in the log retrieving request sent with GET /api/sn_agent/agents/{agent_id}/log.

**URL format**

/api/sn_agent/agents/log/{request_id}/

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_id</td>
<td>Sys_id of a request in the Agent Client Collector Requests [sn_agent_request] table.</td>
</tr>
<tr>
<td></td>
<td>For the request id, run GET /api/sn_agent/agents/{agent_id}/log.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request status is done and the grabbed log is ready.</td>
</tr>
<tr>
<td>202</td>
<td>Grab log request with provided ID is still in progress.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_admin role.</td>
</tr>
<tr>
<td>404</td>
<td>Grab log request with provided ID not found.</td>
</tr>
<tr>
<td>408</td>
<td>Grab log request timed out.</td>
</tr>
<tr>
<td>500</td>
<td>Grab log request encountered an error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>output</td>
<td>Information describing the status.</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example shows how to use a request ID to get the status of a grab log request.

curl "https://instance.service-now.com/api/sn_agent/agents/log/<request_ID>" \
--request GET \
Output:

```json
{
  "output": "SensuSnReadFile OK:
  \{"component":\"agent\",\"level\":\"info\",\"msg\":\"Agent Protection: cpu of all checks: 0\",\"time\":\"2021-04-05T00:21:41-07:00\"\},...
}
```

Agent Client Collector - GET /agents/policies/list

Gets a list of policies that are in published or unpublished draft state.

**URL format**

/api/sn_agent/agents/policies/list

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (JSON)**

```
None
```

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>
| X-Include-Check-Params | Optional. Flag that indicates whether to return check instances and their parameters in the results. Valid values:  
- true: Includes check instances and their parameters in the results.  
- false: Do not include check instances and their parameters in the results.  
  Default: false  
  Data type: Boolean |
| X-Include-Checks-And-Agents | Optional. Flag that indicates whether to include check instances and agents in the results. Valid values:  
- true: Includes check instances and agents in the results.  
- false: Do not include checks and agents in the results.  
  Default: false  
  Data type: Boolean |

### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_user role.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON)

#### Object

**Property**

**policies**

List of policies retrieved. Includes checks and agent in results if queried using specific request headers. For information on policies, see default checks and policies.

```json
{
  "policies": [
    {
      "active": "Boolean",
      "agent_ids": "String",
      "checks": [Array],
      "cred_alias": "String",
      "credential_alias": "String",
      "filter": "String",
      "interval": "Number",
      "monitored_ci_group": "String",
      "monitored_ci_script": "String",
      "monitored_ci_type_filter": "Boolean",
      "monitored_ci_type_group": "Boolean",
      "monitored_ci_type_script": "String",
      "name": "String",
      "params": [Array],
      "publish_status": "String",
      "secure_params": [Array],
      "sys_id": "String"
    }
  ]
}```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_updated_on</td>
<td>&quot;sys_updated_on&quot;: &quot;String&quot;, &quot;table&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>policies.active</td>
<td>Flag that indicates whether the policy is active. Valid values: • true: Policy is active. • false: Policy is not active. Data type: Boolean</td>
</tr>
<tr>
<td>policies.agent_ids</td>
<td>Unique ID of an agent listed in the Agent ID column of the Agent Client Collectors [sn_agent_cmbd_ci_agent] table. These results only display if the X-Include-Checks-And-Agents header parameter is set to true. For extended information of an agent, run the ID in the GET / agents/{agent_id} endpoint.</td>
</tr>
<tr>
<td>policies.checks</td>
<td>List of objects defining checks listed in the Check Instances [sn_agent_check] table. These results only display if the X-Include-Checks-And-Agents or X-Include-Check-Params header parameter is set to true.</td>
</tr>
<tr>
<td></td>
<td>&quot;checks&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;active&quot;: &quot;Boolean&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;auto_generate&quot;: &quot;Boolean&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;check_type&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;command_prefix&quot;: &quot;String&quot;,</td>
</tr>
</tbody>
</table>
### Object (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;command&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;event_status_change_threshold&quot;: &quot;Number&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;event_status_repair_threshold&quot;: &quot;Number&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;interval&quot;: &quot;Number&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;timeout&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>policies.checks.active</td>
<td>Flag that indicates whether the policy check is active.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Policy check is active.</td>
</tr>
<tr>
<td></td>
<td>• false: Policy check is inactive.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>policies.checks.auto_generate</td>
<td>Flag that indicates whether to automatically generate the command with the command prefix value.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Automatically populate the command property with active parameter values.</td>
</tr>
<tr>
<td></td>
<td>• false: Command is not automatically generated.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>policies.checks.check_type</td>
<td>Check type specifying the monitoring option.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>policies.checks.command</td>
<td>Command that the Agent Client Collector executes. Parameter taken from a template or monitored CI.</td>
</tr>
<tr>
<td>policies.checks.command_prefix</td>
<td>If the auto_generate property is true, this command is used for automatic generation. The prefix consists of any portion of the command which is static (does not change), such as the script name.</td>
</tr>
<tr>
<td>policies.checks.event_status_change_threshold</td>
<td>The number of consecutive times that a check’s response status must happen before a new event is sent. Returns null if not set. For example, if this value is 3, a check whose response</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>policies.checks.event_status_repair_threshold</td>
<td>The number of consecutive times that a check's response status must improve to close the previous event. Returns null if not set. For example, if this value is 3, a check whose response status changes from Error to OK closes the previous event and generates a new event with an OK status after the third consecutive occurrence of the status change. Data type: Number</td>
</tr>
<tr>
<td>policies.checks.interval</td>
<td>The amount of time in seconds to wait between check executions. For example, a value of 60 means that the check runs every 60 seconds. Data type: Number</td>
</tr>
<tr>
<td>policies.checks.name</td>
<td>Name of the check. Data type: String</td>
</tr>
<tr>
<td>policies.checks.sys_id</td>
<td>Sys_id of the check listed in the Check Instances [sn_agent_check] table. Data type: String</td>
</tr>
<tr>
<td>policies.checks.timeout</td>
<td>The amount of time, in seconds, after which the check execution stops when no output is returned. For example, a value of 60 means...</td>
</tr>
</tbody>
</table>
### Object (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>that when the check execution doesn't return a value for 60 seconds, the execution stops.</td>
</tr>
<tr>
<td>policies.cred_alias</td>
<td>Name listed in the Credentials [discovery_credentials] table.</td>
</tr>
<tr>
<td>policies.credential_alias</td>
<td>Sys_id of the credential alias in the Connection &amp; Credential Aliases [sys_alias] table.</td>
</tr>
<tr>
<td>policies.filter</td>
<td>Filter restricting the policy's checks to monitor only the specified criteria.</td>
</tr>
<tr>
<td>policies.interval</td>
<td>The amount of time in seconds to wait between policy checks. For example, a value of 60 means that the check runs every 60 seconds.</td>
</tr>
<tr>
<td></td>
<td>Note: The value of the checks.interval property overrides the value configured in this field.</td>
</tr>
<tr>
<td>policies.monitored_ci_group</td>
<td>Name of the CMDB groups associated with the policy. This CMDB is listed in the CMDB Groups [cmdb_group] table. This field is only applied if the value of the monitored_ci_type_group property is true.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>policies.monitored_ci_script</td>
<td>Script for monitoring CIs. This field is only applied if the value of the policies.monitored_ci_type_script property is true.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>policies.monitored_ci_type_filter</td>
<td>Flag that indicates whether filtering by CI type is enabled. The CI type is listed in the table property.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Filtering by checks group is enabled.</td>
</tr>
<tr>
<td></td>
<td>• false: Filtering by checks group is disabled.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>policies.monitored_ci_type_group</td>
<td>Flag that indicates whether monitoring by CMDB group type is enabled.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: CMDB group type is enabled.</td>
</tr>
<tr>
<td></td>
<td>• false: CMDB group type is disabled.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>policies.monitored_ci_type_script</td>
<td>Flag that indicates whether the script for monitoring CIs is enabled.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true: Script for monitoring CIs is enabled.</td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>false: Script for monitoring CIs is disabled.</td>
<td></td>
</tr>
<tr>
<td>policies.name</td>
<td>Name of the policy.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>policies.publish_status</td>
<td>Indicates whether the policy is published. Possible values:</td>
</tr>
<tr>
<td>Draft: Policy has not been published and is editable using the update endpoints.</td>
<td>Data type: String</td>
</tr>
<tr>
<td>Published: Policy is published. The draft (sandbox copy) and published copy are identical.</td>
<td></td>
</tr>
<tr>
<td>Published*: Policy is published, but the draft copy (sandbox view) has changes not found in the published copy.</td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>policies.sys_updated_on</td>
<td>Date and time that the policy was last updated.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>policies.table</td>
<td>Monitored CI-type field on the policy. This field is only applied if monitored_ci_type_filter is true.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>
## Object (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>policies.params</td>
<td>List of objects containing information of check parameters listed in the Check Parameters [sn_agent_check_param] table. These results only display if the X-Include-Check-Params header parameter is set to true.</td>
</tr>
<tr>
<td></td>
<td>```json</td>
</tr>
<tr>
<td></td>
<td>&quot;params&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;active&quot;: &quot;Boolean&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;flag&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;mandatory&quot;: &quot;Boolean&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value_required&quot;: &quot;Boolean&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>policies.params.active</td>
<td>Flag that indicates whether the check parameter is active.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Check parameter is active.</td>
</tr>
<tr>
<td></td>
<td>• false: Check parameter is inactive.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>policies.params.flag</td>
<td>Parameter flag to use during check invocation.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>policies.params.mandatory</td>
<td>Flag that indicates whether this check is mandatory.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• true: This check is mandatory.</td>
</tr>
<tr>
<td></td>
<td>• false: This check is optional.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>policies.params.name</td>
<td>Name of the parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>policies.params.sys_id</td>
<td>Sys_id of the parameter listed in the Check Parameters [sn_agent_check_param] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>policies.params.value</td>
<td>Value of the parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>policies.params.value_required</td>
<td>Flag that indicates whether the information provided by the value property is required. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The value property is required.</td>
</tr>
<tr>
<td></td>
<td>• false: The value property is null or not required.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>policies.secure_params</td>
<td>List of objects containing information of check secure parameters listed in the Check Secure Parameters [sn_agent_check_secure_param] table. For more information, see Create a secure parameter for a check. These results only display if the X-Include-Check-Params header parameter is set to true.</td>
</tr>
</tbody>
</table>

```json
"secure_params": [
  
]```
Object (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>policies.secure_params.active</td>
<td>Flag that indicates whether the check secure parameter is active. Valid values: • true: The check secure parameter is active. • false: The check secure parameter is inactive. Data type: Boolean</td>
</tr>
<tr>
<td>policies.secure_params.name</td>
<td>Name of the secure parameter. Data type: String</td>
</tr>
<tr>
<td>policies.secure_params.order</td>
<td>Order in which the parameter is sent to the check command/script. Data type: Number</td>
</tr>
<tr>
<td>policies.secure_params.sys_id</td>
<td>Sys_id of the record in the Check Secure Parameter [sn_agent_check_secure_param] table. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

The following example shows how to restrict results by query and number. The query returns all active policies and the ID of the associated agent.

```bash
curl "https://instance.service-now.com/api/sn_agent/agents/policies/list" \
--request GET \
```
--header "Accept: application/json"
--header 'X-Enc-Query: active=true'
--header 'X-Include-Checks-And-Agents: true'
--user 'username': 'password'

Output:

```json
{
  "policies": [{
    "name": "Basic Discovery",
    "sys_id": "68bfd27c536113006dfeddeff7b12be",
    "active": "true",
    "interval": "43200",
    "sys_updated_on": "2020-07-21 10:14:12",
    "monitored_ci_type_filter": "true",
    "filter": "discovery_source-AgentClientCollector^ORlast_discoveredRELATIVELT@dayofweek@ago@14",
    "table": "cmdb_ci_server",
    "monitored_ci_type_script": "false",
    "monitored_ci_script": "/*
      Provide a script to get monitored CI type. ..."",
    "monitored_ci_type_group": "false",
    "monitored_ci_group": "null// group name as seen in cmdb_group table",
    "cred_alias": "null// credential name as seen in discovery_credentials table",
    "credential_alias": "null// credential alias sys id as seen in sys_alias table",
    "publish_status": "Published",
    "checks": [
      {
        "name": "check-discovery-basic",
        "sys_id": "5b10c644c7e10010b9a4362c14c260aa",
        "active": "true",
        "command": "check_discover.rb",
        "command_prefix": "check_discover.rb",
        "auto_generate": "true",
        "timeout": "60",
        "interval": "43200",
        "event_status_change_threshold": null,
        "event_status_repair_threshold": null,
        "check_type": "Discovery"
      }
    ],
    "agent_ids": "b1faba21b066256f,a088b75b1b25b0a0"
  }]
}
```
Example: Python request

The following example shows how to get policy, check, and parameter information for a policy called Apache Metrics.

```python
import requests
data = { 'username' : 'username', 'password' : 'password' }
url = "https://instance.service-now.com/api/sn_agent/agents/policies/list"
payload = {}
headers = {
    'X-Include-Check-Params': 'true',
    'X-Enc-Query': 'sys_name=Apache Metrics'
}
response = requests.request("GET", url, headers=headers, data = payload)
print(response.text.encode('utf8'))
```

Output:

```json
{
    "policies": [
        {
            "name": "Apache Metrics",
            "sys_id": "12bdb4f8c7a10010b9a4362c14c2603e",
            "active": "false",
            "interval": "60",
            "sys_updated_on": "2020-07-21 10:21:38",
            "monitored_ci_type_filter": "true",
            "filter": "operational_status=1",
            "table": "cmdb_ci_apache_web_server",
            "monitored_ci_type_script": "false",
            "monitored_ci_script": "*/\nProvide a script to get monitored CI type.",
            "monitored_ci_type_group": "false",
            "monitored_ci_group": "null// group name as seen in cmdb_group table",
            "cred_alias": "null// credential name as seen in discovery_credentials table",
            "credential_alias": "null// credential alias sys id as seen in sys_alias table",
            "publish_status": "Published",
            "checks": [
```
{
    "name": "app.apache.metrics-apache",
    "sys_id": "5abdb4f8c7a10010b9a4362c14c2603f",
    "active": "true",
    "command": "metrics-apache-graphite.rb -p {{.labels.params_port}} -h
{{.labels.params_host}} --path {{.labels.params_path}}",
    "command_prefix": "metrics-apache-graphite.rb",
    "auto_generate": "true",
    "timeout": "60",
    "interval": "60",
    "event_status_change_threshold": null,
    "event_status_repair_threshold": null,
    "check_type": "Metrics",
    "params": [
        {
            "name": "port",
            "sys_id": "1ebdb4f8c7a10010b9a4362c14c26042",
            "value": "80",
            "active": "true",
            "mandatory": "true",
            "value_required": "true",
            "flag": "-p"
        },
        ...
    ],
    "secure_params": [
        {
            "name": "cred_user_name",
            "sys_id": "3d3d782a53170010f42cddeeff7b1c2",
            "active": "true",
            "order": "1"
        },
        {
            "name": "cred_password",
            "sys_id": "6a8d7c2a53170010f42cddeeff7b1c5"
        }
    ]
}
Agent Client Collector - GET /agents/policy/activate/{policy_id}

Activates a published policy.

For a list of published policies, use GET /agents/policies/list. This endpoint only supports sys_ids in which the value of the policy's publish_status property is Published or Published*.

**URL format**

/api/sn_agent/agents/policy/activate/{policy_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <strong>application/json</strong>.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes
The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_admin role.</td>
</tr>
<tr>
<td>404</td>
<td>No record found with the provided sys_id.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Message containing operation success or failure results.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

The following example shows how to activate a policy.

```
curl "https://instance.service-now.com/api/sn_agent/agents/policy/activate/<sys_id>" \
  --request GET \
  --header "Accept:application/json" \
  --user 'username' : 'password'
```

Output:

```
{
  "message": "Operation was successful"
}
```

Agent Client Collector - GET /agents/policy/deactivate/{policy_id}

Deactivates a published policy.

For a list of published policies, use GET /agents/policies/list. This endpoint only supports sys_ids in which the value of the policy's publish_status property is Published or Published*.

**URL format**

/api/sn_agent/agents/policy/activate/{policy_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_admin role.</td>
</tr>
<tr>
<td>404</td>
<td>No record found with the provided sys_id.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Message containing operation success or failure results. Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example shows how to deactivate a policy.

curl "https://instance.service-now.com/api/sn_agent/agents/policy/deactivate/<sys_id>" \  
--request GET \  
--header "Accept:application/json" \  
--user 'username' : 'password'

Output:

```json
{
   "message": "Operation was successful"
}
```

Agent Client Collector - GET /agents/policy/publish/{policy_id}

Publishes a draft policy.

Use any of the following endpoints to modify a draft or sandbox copy prior to publishing:

- POST /agents/update/policy/{policy_id}
- POST /agents/update/check/{check_id}
- POST /agents/update/check_param/{param_id}
- POST /agents/update/check_secure_param/{param_id}

URL format

/api/sn_agent/agents/policy/publish/{policy_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>policy_id</td>
<td>Sys_id of a policy in the Policies [sn_agent_policy] table that is in Draft state or a sandbox copy.</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• For a list of policies and details including publish state, use GET /</td>
</tr>
<tr>
<td></td>
<td>agents/policies/list.</td>
</tr>
<tr>
<td></td>
<td>• To retrieve sandbox copy of a policy, use GET /agents/policy/</td>
</tr>
<tr>
<td></td>
<td>sandbox_from_published/{policy_id}.</td>
</tr>
</tbody>
</table>

Data type: String

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the <code>agent_client_collector_admin</code> role.</td>
</tr>
<tr>
<td>404</td>
<td>No record found with the provided <code>sys_id</code>.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Message containing operation success or failure results. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

The following example shows how to publish a policy.

```bash
curl "https://instance.service-now.com/api/sn_agent/agents/policy/publish/<sys_id>" \
--request GET \
--header "Accept:application/json" \
--user 'username' : 'password'
```

**Output:**

```json
{
   "message": "Operation was successful"
}
```

**Agent Client Collector - GET /agents/policy/sandbox_from_published/{policy_id}**

Gets the sandbox copy of a published policy and provides policy details.

Use the sandbox copy to make updates to a policy and publish. You can use the `sys_ids` in the response body to work with the following endpoints:

- **POST /agents/update/policy/{policy_id}**
- **POST /agents/update/check/{check_id}**
• POST /agents/update/check_param/{param_id}
• POST /agents/update/check_secure_param/{param_id}

For a list of published policies, use GET /agents/policies/list. This endpoint only supports sys_ids in which the value of the policy’s publish_status property is Published or Published*.

URL format
/api/sn_agent/agents/policy/sandbox_from_published/{policy_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>policy_id</td>
<td>Sys_id of the published policy listed in the Policies [sn_agent_policy] table. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_admin role.</td>
</tr>
<tr>
<td>404</td>
<td>No record found with the provided sys_id.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Object&gt;</td>
<td>Extended details of the sandbox copy associated with the policy. For information on policies, see default checks and policies.</td>
</tr>
</tbody>
</table>

```json
{
    "active": "Boolean",
    "agent_ids": "String",
    "checks": [Array],
    "cred_alias": "String",
    "credential_alias": "String",
    "filter": "String",
    "interval": "Number",
    "monitored_ci_group": "String",
    "monitored_ci_script": "String",
    "monitored_ci_type_filter": "Boolean",
    "monitored_ci_type_group": "Boolean",
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>Flag that indicates whether the policy is active. Valid values: • true: Policy is active. • false: Policy is not active. Data type: Boolean</td>
</tr>
<tr>
<td>agent_ids</td>
<td>Unique ID of an agent listed in the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table. These results only display if the x-Include-Checks-And-Agents header parameter is set to true. For extended information of an agent, run the ID in the GET /agents/{agent_id} endpoint.</td>
</tr>
<tr>
<td>checks</td>
<td>List of objects defining checks listed in the Check Instances [sn_agent_check] table.</td>
</tr>
</tbody>
</table>

```json
"checks": [
  {
    "active": "Boolean",
    "auto_generate": "Boolean",
    "check_type": "String"
    "command_prefix": "String",
    "command": "String",
    "event_status_change_threshold": "Number",
    "monitored_ci_type_script": "String",
    "name": "String",
    "params": [Array],
    "publish_status": "String",
    "secure_params": [Array],
    "sys_id": "String",
    "sys_updated_on": "String",
    "table": "String"
  }
]```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
|                           | ```json
"event_status_repair_threshold":
  "Number",
  "interval": "Number",
  "name": "String",
  "sys_id": "String",
  "timeout": "String"
} ``` |
|                           | Data type: Array                                                                                                                           |
| checks.active             | Flag that indicates whether the policy check is active.                                                                                     |
|                           | Valid values:                                                                                                                              |
|                           | • true: Policy check is active.                                                                                                               |
|                           | • false: Policy check is inactive.                                                                                                          |
|                           | Data type: Boolean                                                                                                                         |
| checks.auto_generate      | Flag that indicates whether to automatically generate the command with the `command_prefix` value.                                         |
|                           | Valid values:                                                                                                                              |
|                           | • true: Automatically populate the `command` property with active parameter values.                                                        |
|                           | • false: Command is not automatically generated.                                                                                           |
|                           | Data type: Boolean                                                                                                                         |
| checks.check_type         | Check type specifying the monitoring option. Possible values:                                                                              |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>checks.command</td>
<td>Command that the Agent Client Collector executes. Parameter taken from a template or monitored CI.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If <code>auto_generate</code> is true, this property is automatically populated with the prefix and flags of the active parameters listed in the <code>parameters</code> object.</td>
</tr>
<tr>
<td>checks.command_prefix</td>
<td>If the <code>auto_generate</code> property is true, this command is used for automatic generation. The prefix consists of any portion of the command which is static (does not change), such as the script name.</td>
</tr>
<tr>
<td>checks.event_status_change_threshold</td>
<td>The number of consecutive times that a check's response status must happen before a new event is sent. Returns null if not set. For example, if this value is 3, a check whose response status changes from OK to Error generates a new event with an Error status after the third consecutive occurrence of the status change.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>checks.event_status_repair_threshold</td>
<td>The number of consecutive times that a check's response status must improve to close the previous event. Returns null if not set. For example, if this value is 3, a check whose response status changes from Error to OK closes the previous event and generates a new event with an OK status after the third consecutive occurrence of the status change.</td>
</tr>
<tr>
<td>checks.interval</td>
<td>The amount of time in seconds to wait between check executions. For example, a value of 60 means that the check runs every 60 seconds.</td>
</tr>
<tr>
<td>checks.name</td>
<td>Name of the check.</td>
</tr>
<tr>
<td>checks.sys_id</td>
<td>Sys_id of the check listed in the Check Instances [sn_agent_check] table. The POST /agents/update/check/{check_id} endpoint takes this value to update the sandbox copy.</td>
</tr>
<tr>
<td>checks.timeout</td>
<td>The amount of time, in seconds, after which the check execution stops when no output is returned. For example, a value of 60 means that when the check execution doesn’t return a value for 60 seconds, the execution stops.</td>
</tr>
<tr>
<td>cred_alias</td>
<td>Name listed in the Credentials [discovery_credentials] table.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>credential_alias</td>
<td>Sys_id of the credential alias in the Connection &amp; Credential Aliases [sys_alias] table. Data type: String</td>
</tr>
<tr>
<td>filter</td>
<td>Filter restricting the policy's checks to monitor only the specified criteria. Data type: String</td>
</tr>
<tr>
<td>interval</td>
<td>The amount of time in seconds to wait between policy checks. For example, a value of 60 means that the check runs every 60 seconds. <strong>Note:</strong> The value of the checks.interval property overrides the value configured in this field. Data type: Number</td>
</tr>
<tr>
<td>monitored_ci_group</td>
<td>Name of the CMDB groups associated with the policy. This CMDB is listed in the CMDB Groups [cmdb_group] table. This field is only applied if the value of the monitored_ci_type_group property is true. Data type: String</td>
</tr>
<tr>
<td>monitored_ci_script</td>
<td>Script for monitoring CIs. This field is only applied if the value of the policies.monitored_ci_type_script property is true. Data type: String</td>
</tr>
<tr>
<td>monitored_ci_type_filter</td>
<td>Flag that indicates whether filtering by CI type is enabled. The CI type is listed in the table property. Valid values:</td>
</tr>
</tbody>
</table>

Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• true: Filtering by checks group is enabled.</td>
</tr>
<tr>
<td></td>
<td>• false: Filtering by checks group is disabled.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>monitored_ci_type_group</td>
<td>Flag that indicates whether monitoring by CMDB group type is enabled.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: CMDB group type is enabled.</td>
</tr>
<tr>
<td></td>
<td>• false: CMDB group type is disabled.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>monitored_ci_type_script</td>
<td>Flag that indicates whether the script for monitoring CIs is enabled.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Script for monitoring CIs is enabled.</td>
</tr>
<tr>
<td></td>
<td>• false: Script for monitoring CIs is disabled.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>name</td>
<td>Name of the policy.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>publish_status</td>
<td>Indicates whether the policy is published.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Draft: Policy has not been published and is editable using the update</td>
</tr>
<tr>
<td></td>
<td>endpoints.</td>
</tr>
<tr>
<td></td>
<td>• Published: Policy is published.</td>
</tr>
<tr>
<td></td>
<td>The draft (sandbox copy) and published copy are identical.</td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
</tr>
<tr>
<td></td>
<td>• Published*: Policy is published, but the draft copy (sandbox view) has changes not found in the published copy.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>params</strong></td>
<td>List of objects containing information of check parameters listed in the Check Parameters [sn_agent_check_param] table. These results only display if the <strong>X-Include-Check-Params</strong> header parameter is set to true.</td>
</tr>
<tr>
<td></td>
<td>&quot;params&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;active&quot;: &quot;Boolean&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;flag&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;mandatory&quot;: &quot;Boolean&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value_required&quot;: &quot;Boolean&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td><strong>params.active</strong></td>
<td>Flag that indicates whether the check parameter is active.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Check parameter is active.</td>
</tr>
<tr>
<td></td>
<td>• false: Check parameter is inactive.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td><strong>params.flag</strong></td>
<td>Parameter flag to use during check invocation.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>params.mandatory</strong></td>
<td>Flag that indicates whether this check is mandatory.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: This check is mandatory.</td>
</tr>
<tr>
<td></td>
<td>• false: This check is optional.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>params.name</td>
<td>Name of the parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>params.sys_id</td>
<td>Sys_id of the parameter listed in the Check Parameters [sn_agent_check_param] table. The POST /agents/update/check_param/{param_id} endpoint takes this value to update the sandbox copy.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>params.value</td>
<td>Value of the parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>params.value_required</td>
<td>Flag that indicates whether the information provided by the value property is required.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The value property is required.</td>
</tr>
<tr>
<td></td>
<td>• false: The value property is null or not required.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>secure_params</td>
<td>List of objects containing information of check secure parameters listed in the Check Secure Parameters [sn_agent_check_secure_param] table. For more information, see Create a secure parameter for a check. These results only display if the X-Include-Check-Params header parameter is set to true.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
|                     | {"active": "Boolean",
|                     |    "name": "String",
|                     |    "order": "Number",
|                     |    "sys_id": "String"
|                     | }                                                                                                                                                                                                          |
|                     | Data type: Array                                                                                                                                                                                              |
| secure_params.active | Flag that indicates whether the check secure parameter is active. Valid values:
|                     | • true: The check secure parameter is active.
|                     | • false: The check secure parameter is inactive.                                                                                                                                                             |
|                     | Data type: Boolean                                                                                                                                                                                            |
| secure_params.name  | Name of the secure parameter.                                                                                                                                                                                |
|                     | Data type: String                                                                                                                                                                                             |
| secure_params.order | Order in which the parameter is sent to the check command/script.                                                                                                                                               |
|                     | Data type: Number                                                                                                                                                                                             |
| secure_params.sys_id| Sys_id of the record located in the Check Secure Parameter [sn_agent_check_secure_param] table. The POST /agents/update/check_secure_param/{param_id} endpoint takes this value to update the sandbox copy. |
|                     | Data type: String                                                                                                                                                                                             |
| sys_id              | Sys_id of the policy listed in the Policies [sn_agent_policy] table. The POST /agents/update/policy/{policy_id} endpoint takes this value to update the sandbox copy.                                             |
|                     | Data type: String                                                                                                                                                                                             |
**Property** | **Description**  
--- | ---  
sys_updated_on | Date and time that the policy was last updated.  
Data type: String  
table | Monitored CI-type field on the policy. This field is only applied if monitored_ci_type_filter is true.  
Data type: String  

**Example: cURL request**

The following shows how to get information of the Docker Container Metrics policy.

```bash
curl  
"https://instance.service-now.com/api/sn_agent/agents/policy/sandbox_from_published/<sys_id>" 
--request GET 
--header "Accept:application/json" 
--user 'username':'password'
```

**Output:**

```
{
  "name": "Docker Container Metrics",
  "sys_id": "b01e609a1b9fe4943e7f0b05464bcb91",
  "active": "false",
  "interval": "60",
  "sys_updated_on": "2021-04-05 19:52:28",
  "monitored_ci_type_filter": "true",
  "filter": "operational_status=1",
  "table": "cmdb_ci_docker_container",
  "monitored_ci_type_script": "false",
  "monitored_ci_script": "/*
  Provide a script to get monitored CI type.",
  "monitored_ci_type_group": "false",
  "monitored_ci_group": "null// group name as seen in cmdb_group table",
  "cred_alias": "null// credential name as seen in discovery_credentials table",
  "credential_alias": "null// credential alias sys id as seen in sys_alias table",
  "publish_status": "Published",
  "checks": [
    
    {
      "name": "container.docker.metrics-docker",
      "sys_id": "701e609a1b9fe4943e7f0b05464bcb94",
```
Agent Client Collector - GET /agents/{agent_id}/restart

Restarts a specified agent with alive/up status.

If Agent Client Collector performance issues occur, you can restart the agent. Manual restart is supported in the following environments:

- Linux-based agents using systemd
- Windows agents
**URL format**

/api/sn_agent/agents/{agent_id}/restart

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent_id</td>
<td>Unique ID of an agent listed in the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table. For a list of agent IDs and other details, run the <code>GET /agents/list</code> endpoint. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Request body parameters (JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_admin role.</td>
</tr>
<tr>
<td>404</td>
<td>No record found with the provided sys_id or this agent does not support restart.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Message containing operation success or failure results. Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example shows how to restart an agent.

```bash
curl "https://instance.service-now.com/api/sn_agent/agents/<agent_id>/restart" \
    --request GET \
    --header "Accept:application/json" \
    --user 'username':'password'
```

Output:

```json
{
   "message": "Restarting Agent With ID: <agent_id>"
}
```

Agent Client Collector - POST /agents/check_defs/{check_def_id}/run

Runs a check against the given configuration item.

To stop a background check, use the request ID provided in the GET /agents/exec/background/stop/{request_id} API.
URL format
/api/sn_agent/agents/check_defs/{check_def_id}/run

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check_def_id</td>
<td>Sys_id of a check definition in the Check Definitions [sn_agent_check_def] table.</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| params | Map of parameter names and values. These settings can be used to override the parameter records of the check definition and its specified values.  
"params": { 
    "<parameter name>": "String" 
}  
Data type: Object |
| priority | Priority of the request to be set on the ECC queue. Possible values:  
• 0: interactive  
• 1: expedited  
• 2: standard  
Data type: Number |
| query | Encoded query for retrieving the GlideRecord from the table specified in the table property.  
Data type: String |
| table | Name of the cmdb_ci table for this background check. |
Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>timeout</td>
<td>Value of the timeout for the request in seconds.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>The is an error in the arguments provided in the request body.</td>
</tr>
<tr>
<td>404</td>
<td>The check definition with the ID provided is not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestId</td>
<td>Sys_id of the generated background check request.</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example shows how to run a background check and get its request ID.

```
curl
  "https://instance.service-now.com/api/sn_agent/agents/check_defs/a90d3c361be1301060d2773ad544cb6f/run" 
--request POST 
--header "Accept:application/json" 
--header "Content-Type:application/json" 
--data "{"table":"sn_agent_check_def"}"
--user 'username': 'password'
```

Output:

```
{
  "request_id": "278c0170db2a30108a0751f4f3961926"
}
```

Agent Client Collector - POST /agents/check_defs/{check_def_id}/test

Enables building test check requests on check definitions.

Use this API for the following tasks:

- Set the check definition to test
- Set the configuration item to run the test against

You can also specify one of the following identifiers to use during the test:

- Credential sys_id
- Credential alias id
- Credential name

URL format

/api/sn_agent/agents/check_defs/{check_def_id}/test
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check_def_id</td>
<td>Sys_id of the check definition listed in the Check Definitions [sn_agent_check_def] table.</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ci_id</td>
<td>Sys_id of a CMDB Configuration Item.</td>
</tr>
<tr>
<td>credentials_id</td>
<td>Sys_id of a credentials record.</td>
</tr>
<tr>
<td>credentials_name</td>
<td>Name of the credentials record.</td>
</tr>
<tr>
<td>credentials_alias_id</td>
<td>Sys_id of a credentials alias record.</td>
</tr>
<tr>
<td>credentials_alias_name</td>
<td>Name of a credentials alias.</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>500</td>
<td>Error building the test request.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result_id</td>
<td>Sys_id of the test result record.</td>
</tr>
</tbody>
</table>

Agent Client Collector - POST /agents/check_instances/{check_instance_id}/test

Enables building test check requests on check instances.

Use this API for the following tasks:

- Set the check instance to test
- Set the configuration item to run the test against

You can also specify one of the following identifiers to use during the test:

- Credential sys_id
- Credential alias id
- Credential name

URL format

/api/sn_agent/agents/check_instances/{check_instance_id}/test
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check_instance_id</td>
<td>Sys_id of the check definition listed in the Check Definitions [sn_agent_check_def] table.</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ci_id</td>
<td>Sys_id of a CMDB Configuration Item.</td>
</tr>
<tr>
<td>credentials_id</td>
<td>Sys_id of a credentials record.</td>
</tr>
<tr>
<td>credentials_name</td>
<td>Name of the credentials record.</td>
</tr>
<tr>
<td>credentials_alias_id</td>
<td>Sys_id of a credentials alias record.</td>
</tr>
<tr>
<td>credentials_alias_name</td>
<td>Name of a credentials alias.</td>
</tr>
<tr>
<td>proxy_agent_id</td>
<td>Unique ID of an agent proxy to run this check. This value is listed in the Agent ID column of the Agent Client Collectors [sn_agent_cmdb_ci_agent] table.</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>500</td>
<td>Error building the test request.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result_id</td>
<td>Sys_id of the test result record.</td>
</tr>
</tbody>
</table>

Agent Client Collector - POST /agents/update/check/{check_id}

Updates a selected policy check.

To retrieve the properties of a policy sandbox copy, use GET /agents/policy/sandbox_from_published/{policy_id}.

URL format

/api/sn_agent/agents/update/check/{check_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check_id</td>
<td>Sys_id of a policy check sandbox copy in the Check Instances [sn_agent_check] table. Data type: String</td>
</tr>
</tbody>
</table>
## Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

## Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>Flag that indicates whether the policy check is active. Valid values: • true: Policy check is active. • false: Policy check is inactive. Data type: Boolean</td>
</tr>
<tr>
<td>auto_generate</td>
<td>Flag that indicates whether to automatically generate the command with the command_prefix value. Valid values: • true: Automatically populate the command property with active parameter values. • false: Command is not automatically generated. Data type: Boolean</td>
</tr>
<tr>
<td>check_type</td>
<td>Check type specifying the monitoring option. Possible values: • Discovery: Check that locates CIs related to the agent. • Events: The check's result is transformed into an Event Management event. • Metrics: The values from the check result are transformed to metrics. Data type: String</td>
</tr>
</tbody>
</table>
## Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>command</strong></td>
<td>Command that the Agent Client Collector executes. Parameter taken from a template or monitored CI.</td>
</tr>
<tr>
<td></td>
<td>⚠️ <strong>Note:</strong> If <code>auto_generate</code> is true, this property is automatically populated with the prefix and flags of the active parameters listed in the <code>parameters</code> object.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>command_prefix</strong></td>
<td>If the <code>auto_generate</code> property is true, this command is used for automatic generation. The prefix consists of any portion of the command which is static (does not change), such as the script name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>event_status_change_threshold</strong></td>
<td>The number of consecutive times that a check's response status must happen before a new event is sent. Returns null if not set. For example, if this value is 3, a check whose response status changes from OK to Error generates a new event with an Error status after the third consecutive occurrence of the status change.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td><strong>event_status_repair_threshold</strong></td>
<td>The number of consecutive times that a check's response status must improve to close the previous event. Returns null if not set. For example, if this value is 3, a check whose response status changes from Error to OK closes the previous event and generates a new event with an OK status after the third consecutive occurrence of the status change.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
</tbody>
</table>
Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interval</td>
<td>The amount of time in seconds to wait between check executions. For example, a value of 60 means that the check runs every 60 seconds. Data type: Number</td>
</tr>
<tr>
<td>name</td>
<td>Name of the check. Data type: String</td>
</tr>
<tr>
<td>timeout</td>
<td>The amount of time, in seconds, after which the check execution stops when no output is returned. For example, a value of 60 means that when the check execution doesn’t return a value for 60 seconds, the execution stops. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_admin role.</td>
</tr>
<tr>
<td>404</td>
<td>No record found with the provided sys_id.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Message containing operation success or failure results.</td>
</tr>
</tbody>
</table>

Data type: String

### Example: cURL request

The following example shows how to update the event change and repair properties of a policy check.

```bash
curl "https://instance.service-now.com/api/sn_agent/agents/update/check/<check_sys_id>" \
--request POST \
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--data "{
"event_status_change_threshold" : "2",
"event_status_repair_threshold" : "1"
}" \
--user 'username':'password'
```

Output:

```json
{
"message": "Operation was successful"
}
```

### Agent Client Collector - POST /agents/update/check_def_params/

{check_def_param_id}

Enables changing one or more field values of a specified check parameter.
**URL format**

/api/sn_agent/agents/update/check_def_params/{check_def_param_id}

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check_def_param_id</td>
<td>Sys_id of the check parameter listed in the Check Parameter Definitions [sn_agent_check_param_def] table. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>Flag that indicates whether the check parameter is active. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The check parameter is active.</td>
</tr>
<tr>
<td></td>
<td>• false: The check parameter is inactive.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>default_value</td>
<td>Specifies the default value for this check parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>mandatory</td>
<td>Flag that indicates whether the check parameter is required. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The check parameter is required.</td>
</tr>
<tr>
<td></td>
<td>• false: The check parameter is optional.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>name</td>
<td>Name of the check parameter.</td>
</tr>
</tbody>
</table>
Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Note:** Refer to the data dictionary for a comprehensive list of Check Definition fields and types.

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <strong>application/json</strong>.</td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

#### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>404</td>
<td>The check parameter was not found with sys_id provided.</td>
</tr>
<tr>
<td>500</td>
<td>Error updating the check parameter.</td>
</tr>
</tbody>
</table>
Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Success or error message.</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example shows how to activate a check parameter.

curl
"https://instance.service-now.com/api/sn_agent/agents/update/check_def_params/02d89bb01b307490f271ea42b24bcb63" \ 
--request POST \ 
--header "Accept:application/json" \ 
--header "Content-Type:application/json" \ 
--data "{"active":true}" \ 
--user 'username': 'password'

Output:

"message": "Check Definition Parameter Updated Successfully"

Agent Client Collector - POST /agents/update/check_def_secure_params/
{check_def_secure_param_id}

Enables changing one or more field values of a specified check secure parameter.

URL format

/api/sn_agent/agents/update/check_def_secure_params/
{check_def_secure_param_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check_def_secure_param_id</td>
<td>Sys_id of the secure parameter listed in the Check Secure Parameter Definitions [sn_agent_check_secure_param_def] table. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>Flag that indicates whether the secure parameter is active. Valid values: • true: The secure parameter is active. • false: The secure parameter is inactive. Data type: Boolean</td>
</tr>
<tr>
<td>name</td>
<td>Name of the secure parameter. Data type: String</td>
</tr>
<tr>
<td>order</td>
<td>Order in which the parameter is sent to the check command/script. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes
The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>404</td>
<td>The check secure parameter was not found with sys_id provided.</td>
</tr>
<tr>
<td>500</td>
<td>Error updating the check secure parameter.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Success or error message.</td>
</tr>
</tbody>
</table>

Example: cURL request
The following example shows how to activate a check secure parameter.

curl
   "https://instance.service-now.com/api/sn_agent/agents/update/check_def_secure_params/2d30a066dba530106f4810284b9619c1" \
   --request POST \n   --header "Accept:application/json" \n   --header "Content-Type:application/json" \n   --data "{"active":"true"}" \n   --user 'username':'password'

Output:
"message": "Check Definition Secure Parameter Updated Successfully"

Agent Client Collector - POST /agents/update/check_defs/{check_def_id}
Enables changing one or more field values of a specified check definition.

URL format
/api/sn_agent/agents/update/check_defs/{check_def_id}
# Supported request parameters

## Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>check_def_id</td>
<td>Sys_id of the check definition listed in the Check Definitions [sn_agent_check_def] table. Data type: String</td>
</tr>
</tbody>
</table>

## Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

## Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>Indicates whether this check definition is active. Valid values: • 0: This check definition is inactive. • 1: This check definition is active. Data type: Number</td>
</tr>
<tr>
<td>background</td>
<td>Flag that indicates whether this check definition is a background check. A background check is a check which the agent starts execution of and doesn't wait for it to finish running. Valid values: • true: This check definition is a background check. • false: This check definition is not a background check. Data type: Boolean</td>
</tr>
<tr>
<td>check_group</td>
<td>Group specified for this check definition.</td>
</tr>
<tr>
<td>check_type</td>
<td>Type of check. Possible values:</td>
</tr>
</tbody>
</table>
### Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Events – Check results are transformed into an Event Management event.</td>
</tr>
<tr>
<td></td>
<td>• Metrics – Values from the check result are transformed to metrics.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>command</td>
<td>Command that the Agent Client Collector executes.</td>
</tr>
<tr>
<td>name</td>
<td>Name of the check.</td>
</tr>
<tr>
<td>params</td>
<td>Map of parameter names and values. These settings can be used to override the</td>
</tr>
<tr>
<td></td>
<td>parameter records of the check definition and its specified values.</td>
</tr>
<tr>
<td></td>
<td>&quot;params&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;&lt;parameter name&gt;&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>plugins</td>
<td>List of Agent Client Collector plugins associated with this check.</td>
</tr>
<tr>
<td>proxy_valid</td>
<td>Flag that indicates whether the check definition policy is set to work as a</td>
</tr>
<tr>
<td></td>
<td>proxy.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: This check definition policy is set to work as a proxy.</td>
</tr>
<tr>
<td></td>
<td>• false: This check definition policy is not set to work as a proxy.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>query</td>
<td>Encoded query for retrieving the GlideRecord from the table specified in the</td>
</tr>
<tr>
<td></td>
<td><strong>table</strong> property.</td>
</tr>
<tr>
<td>table</td>
<td>Name of the cmdb_ci table for this check.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeout</td>
<td>Timeout in seconds.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
</tbody>
</table>

ℹ️ Note: Refer to the data dictionary for a comprehensive list of Check Definition fields and types.

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>404</td>
<td>The check definition was not found with sys_id provided.</td>
</tr>
<tr>
<td>500</td>
<td>Error updating the check definition.</td>
</tr>
</tbody>
</table>
Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Success or error message.</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example shows how to deactivate a check definition.

```cURL
curl "https://instance.service-now.com/api/sn_agent/agents/update/check_defs/99e12466dba530106f4810284b961976" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --data "{"active":false}" \
  --user 'username':'password'
```

Output:

"message": "Check Definition Updated Successfully"

Agent Client Collector - POST /agents/update/check_param/{param_id}

Updates a selected policy check parameter.

To retrieve the properties of a policy sandbox copy, use GET /agents/policy/sandbox_from_published/{policy_id}.

URL format

/api/sn_agent/agents/update/check_param/{param_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>param_id</td>
<td>Sys_id of policy check parameter sandbox copy in the Check Parameters [sn_agent_check_param] table. Data type: String</td>
</tr>
</tbody>
</table>
## Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

## Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>Flag that indicates whether the check parameter is active.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Check parameter is active.</td>
</tr>
<tr>
<td></td>
<td>• false: Check parameter is inactive.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>flag</td>
<td>Flag that indicates whether this check is mandatory.</td>
</tr>
<tr>
<td>mandatory</td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: This check is mandatory.</td>
</tr>
<tr>
<td></td>
<td>• false: This check is optional.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>name</td>
<td>Name of the parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>value</td>
<td>Value of the parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>value_required</td>
<td>Flag that indicates whether the information provided by the value property is required.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: The value property is required.</td>
</tr>
<tr>
<td></td>
<td>• false: The value property is null or not required.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the <code>agent_client_collector_admin</code> role.</td>
</tr>
<tr>
<td>404</td>
<td>No record found with the provided <code>sys_id</code>.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Message containing operation success or failure results. Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

The following example shows how to update several properties of a policy check parameter.

```bash
curl
  "https://instance.service-now.com/api/sn_agent/agents/update/check_param/<param_sys_id>" \
--request POST \
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--data "{
  "flag" : "-d",
  "mandatory" : "true",
  "name" : "scheme2",
  "value" : "120",
  "value_required" : "false"
}" \
--user 'username':'password'
```

Output:

```
{
  "message": "Operation was successful"
}
```

Agent Client Collector - POST /agents/update/check_secure_param/{param_id}

Updates a selected policy check secure parameter.

To retrieve the properties of a policy sandbox copy, use GET /agents/policy/sandbox_from_published/{policy_id}.

**URL format**

/api/sn_agent/agents/update/check_secure_param/{param_id}

**Supported request parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>param_id</td>
<td>Sys_id of the policy check secure parameter sandbox copy in the Check Secure Parameter [sn_agent_check_secure_param] table. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| active | Flag that indicates whether the check secure parameter is active.  
Valid values:
  • true: The check secure parameter is active.
  • false: The check secure parameter is inactive.  
Data type: Boolean |
| name | Name of the secure parameter.  
Data type: String |
| order | Order in which the parameter is sent to the check command/script.  
Data type: Number |

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_admin role.</td>
</tr>
<tr>
<td>404</td>
<td>No record found with the provided sys_id.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Message containing operation success or failure results. Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example shows how to update the properties of a policy check secure parameter.

```bash
curl "https://instance.service-now.com/api/sn_agent/agents/update/check_secure_param/<param_sys_id>" \  
--request POST \  
--header "Accept:application/json" \  
--header "Content-Type:application/json" \  
--data 
"{  
"name": "new name",  
"order": "2"  
}" \ 
--user 'username': 'password'
```

Output:

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Agent Client Collector - POST /agents/update/policy/{policy_id}

Updates a sandbox copy of a policy.

To retrieve the properties of a policy sandbox copy, use GET /agents/policy/sandbox_from_published/{policy_id}.

**URL format**
/api/sn_agent/agents/update/policy/{policy_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cred_alias</td>
<td>Name listed in the Credentials [discovery_credentials] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>credential_alias</td>
<td>Sys_id of the credential alias in the Connection &amp; credential Aliases [sys_alias] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>filter</td>
<td>Filter restricting the policy's checks to monitor only the specified criteria.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>interval</strong></td>
<td>The amount of time in seconds to wait between policy checks. For example, a value of 60 means that the check runs every 60 seconds.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The value of the <code>checks.interval</code> property overrides the value configured in this field.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td><strong>monitored_ci_group</strong></td>
<td>Name of the CMDB groups associated with the policy. This CMDB is listed in the CMDB Groups [cmdb_group] table. This field is only applied if the value of the <code>monitored_ci_type_group</code> property is true.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>monitored_ci_script</strong></td>
<td>Script for monitoring CIs. This field is only applied if the value of the <code>policies.monitored_ci_type_script</code> property is true.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>monitored_ci_type_filter</strong></td>
<td>Flag that indicates whether filtering by CI type is enabled. The CI type is listed in the <code>table</code> property. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Filtering by checks group is enabled.</td>
</tr>
<tr>
<td></td>
<td>• false: Filtering by checks group is disabled.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td><strong>monitored_ci_type_group</strong></td>
<td>Flag that indicates whether monitoring by CMDB group type is enabled. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: CMDB group type is enabled.</td>
</tr>
<tr>
<td></td>
<td>• false: CMDB group type is disabled.</td>
</tr>
</tbody>
</table>
Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>monitored_ci_type_script</td>
<td>Flag that indicates whether the script for monitoring CIs is enabled.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Script for monitoring CIs is enabled.</td>
</tr>
<tr>
<td></td>
<td>• false: Script for monitoring CIs is disabled.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>name</td>
<td>Name of the policy.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>table</td>
<td>Monitored CI-type field on the policy. This field is only applied if monitored_ci_type_filter is true.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the agent_client_collector_admin role.</td>
</tr>
<tr>
<td>404</td>
<td>No record found with the provided sys_id.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Message containing operation success or failure results. Data type: String</td>
</tr>
</tbody>
</table>

### Example: cURL request

The following example shows how to update the name and filter properties/fields of a policy.

```bash
curl "https://instance.service-now.com/api/sn_agent/agents/update/policy/<policy_sys_id>" 
  --request POST 
  --header "Accept:application/json" 
  --header "Content-Type:application/json" 
  --data "{
    "name": "new policy name",
    "filter": "operational_status=1"
  }" 
  --user 'username': 'password'
```

Output:

```json
{
  "message": "Operation was successful"
}
```

### Aggregate API

The Aggregate API allows you to compute aggregate statistics about existing table and column data.
For Aggregate API requests, you must have read access for all records in the table you query. If an ACL prevents the requesting user from accessing any record in the table, the request returns a 403 Forbidden error.

**Aggregate - GET /now/stats/{tableName}**

Retrieves records for the specified table and performs aggregate functions on the returned values.

You can specify which aggregate functions to perform by using either the `sysparm_<aggregate>_fields` parameter or `sysparm_having=<aggregate>^field^operator^value` parameter, substituting `<aggregate>` for one of these aggregate functions:

- `avg`
- `max`
- `min`
- `sum`

**URL format**

Versioned URL: `/api/now/{api_version}/stats/{tableName}`
Default URL: `/api/now/stats/{tableName}`

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>api_version</code></td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td><code>tableName</code></td>
<td>Name of the table for which to retrieve records. Data type: String</td>
</tr>
</tbody>
</table>
### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>An alternative to using the <code>sysparm_query</code> parameter. You can filter a query using key-value pairs where the key is the name of a field. For example, instead of using the parameter <code>&amp;sysparm_query=active=true</code>, you can use <code>&amp;active=true</code>. You can use the display value when the field is a choice or reference type field, such as <code>&amp;state=closed</code> instead of <code>&amp;state=7</code>. To specify multiple key-value pairs, separate each with an ampersand, such as <code>&amp;active=true&amp;assigned_to=john.smith</code>. Data type: String</td>
</tr>
<tr>
<td>sysparm_&lt;aggregate&gt;_fields</td>
<td>List of fields on which to perform each aggregate operation. You can specify multiple fields by separating each with a comma. For example, to get the average values from the duration and priority fields, use <code>sysparm_avg_fields=duration,priority</code>. Note: Specify this parameter, the <code>sysparm_count</code> parameter, or both for your query to return meaningful results. If neither parameter is passed, no aggregate operation is performed. Data type: String</td>
</tr>
<tr>
<td>sysparm_count</td>
<td>Flag that determines whether to return the number of records returned by the query. Note: Specify this parameter, the <code>sysparm_&lt;aggregate&gt;_fields</code> parameter, or both for your query to return meaningful results. If neither parameter is passed, no aggregate operation is performed. Data type: String</td>
</tr>
<tr>
<td>sysparm_display_value</td>
<td>Data retrieval operation when grouping by reference or choice fields. Based on this value, the query returns either the display value, the actual value in the database, or both. • <strong>true</strong>: Returns display values for all fields. • <strong>false</strong>: Returns actual values from the database. If a value is not specified, this parameter defaults to false. • <strong>all</strong>: Returns both actual and display values.</td>
</tr>
</tbody>
</table>
## Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_group_by</td>
<td>Fields by which to group the returned data. You can specify multiple fields by separating each field with a comma, such as <code>sysparm_group_by=priority, state</code>.</td>
<td>String</td>
</tr>
<tr>
<td>sysparm_having</td>
<td>Additional query that enables you to filter the data based on an aggregate operation. The value for this parameter must follow the syntax <code>aggregate^field^operator^value</code>, such as <code>count^priority^&gt;^3</code> to obtain the number of records within the query results with a priority greater than 3. You can specify multiple queries by separating each with a comma, such as <code>count^state^=^1, avg^priority^&gt;^3</code>.</td>
<td>String</td>
</tr>
<tr>
<td>sysparm_orderby</td>
<td>List of values by which to order grouped results. You can specify an order using a field or an aggregate. For example, if you specify <code>sysparm_orderby=AVG^state</code>, groups of results with lower average state values are returned first. You can also order by <code>COUNT</code> to arrange groups of records by the number of records in each group. When you specify an order, groups are ordered in ascending order by default. Use <code>^DESC</code> to sort in descending order, such as <code>sysparm_orderby=state^DESC</code>.</td>
<td>String</td>
</tr>
<tr>
<td>sysparm_query</td>
<td>An encoded query.</td>
<td>String</td>
</tr>
</tbody>
</table>

For example:

```plaintext
(sysparm_query=active=true)
(sysparm_query=caller_id=javascript:gs.getUserID()^active=true)
```
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Depends on specified table and specified request parameters.</td>
</tr>
</tbody>
</table>

Example: Sample cURL request

```bash
curl "https://instance.servicenow.com/api/now/stats/incident?sysparm_avg_fields=reassignment_count%2Cbusiness_stc&sysparm_group_by=assignment_group" \
--request GET \
--header "Accept:application/json" \
--user "username":"password"

{
  "result": [
    {
      "stats": {
        "avg": {
          "business_stc": "804162.7143",
          "reassignment_count": "1.0000"
        }
      },
      "groupby_fields": [
        {
          "value": "",
          "field": "assignment_group"
        }
      ]
    },
    {
      "stats": {
        "avg": {
          "business_stc": "2037371.0000",
          "reassignment_count": "1.5000"
        }
      },
      "groupby_fields": [
        {
          "value": "287ee6fe9fe198100ada7950d0b1b73",
          "field": "assignment_group"
        }
      ]
    }
  ]
}
```
"stats": {
"avg": {
"business_stc": "1821488.2857",
"reassignment_count": "1.1111"
}
},
"groupby_fields": [ {
"value": "8a5055c9c61122780043563ef53438e3",
"field": "assignment_group"
}
]}
},
"stats": {
"avg": {
"business_stc": "1730322.0000",
"reassignment_count": "1.2500"
}
},
"groupby_fields": [ {
"value": "287ebd7da9fe198100f92cc8d1d2154e",
"field": "assignment_group"
}
]}
},
"stats": {
"avg": {
"business_stc": "1564478.6250",
"reassignment_count": "1.2500"
}
},
"groupby_fields": [ {
"value": "d625dcecc0a8016700a222a0f7900d06",
"field": "assignment_group"
}
]}
}
Example: Sample Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url =
  'https://instance.servicenow.com/api/now/stats/incident?sysparm_avg_fields=reassignment_count%2Cbusiness_stc&sysparm_group_by=assignment_group'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Accept":"application/json"}

# Make the HTTP request
response = requests.delete(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
    response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <stats>
      <avg>
        <business_stc>804162.7143</business_stc>
        <reassignment_count>1.0000</reassignment_count>
      </avg>
    </stats>
  </result>
  <result>
    <stats>
      <avg>
        <business_stc>2037371.0000</business_stc>
        <reassignment_count>1.5000</reassignment_count>
      </avg>
    </stats>
  </result>
  <result>
    <stats>
      <avg>
        <business_stc>1821488.2857</business_stc>
        <reassignment_count>1.1111</reassignment_count>
      </avg>
    </stats>
  </result>
</response>
AI Search External User Mapping API

The AI Search External User Mapping API provides endpoints that enable ingestion of user mapping information from external sources into the ServiceNow® AI Search application's index.

AI Search External User Mapping - POST /ais/external_content/user_mapping/import_multiple/{target_table}

Imports a list of external user mappings into an AI Search user mapping table. Each mapping specifies externally defined user and group aliases for a Now Platform user. AI Search uses these aliases to determine which external document search results the user can view.

To learn how AI Search external content security utilizes external user mappings, see External content security for AI Search.

User and group names in external user mappings should match those specified in security access permissions for external documents ingested via the External Content Ingestion API's POST /ais/external_content/ingestDocument/{schema_table_name} endpoint. For more information on ingesting external documents with security access permissions for externally defined users and groups, see External Content Ingestion API.

To view import histories for external user mapping records imported through this endpoint, navigate to AI Search > External Content > User mapping import history. From a history record, you can view the Import Set [sys_import_set] and Multi Import Set [sys_multi_import_set] records for an import task. Use information from these records to verify that your external user mappings were imported correctly.

**URL format**

Versioned URL: /api/now/{api_version}/ais/external_content/user_mapping/import_multiple/{target_table}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>
### Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>target_table</td>
<td>The name of the AI Search user mapping table that you want the imported mappings to appear in. For example, x_snc_sharepoint_user_table.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> You must create the target AI Search user mapping table via the <strong>AI Search &gt; External Content &gt; Create User Mapping</strong> module before using this endpoint. If you specify a user mapping table that does not exist, the request fails.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>{object}</td>
<td>Required. Unnamed object.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;records&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>{object}.records</td>
<td>Required. Array of objects in which each object represents a user mapping to import into the specified target table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;records&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;external_group&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;external_user&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;mapping_value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{object}.records.external_group</code></td>
<td>Array of strings where each string is the name of an externally defined group to set as an alias for the Now Platform user specified by the <code>{object}.records.mapping_value</code> parameter. Data type: Array. Values can be in any format, depending on the names of the externally defined groups selected for the user mapping. Examples include:</td>
</tr>
<tr>
<td></td>
<td>&quot;external_group&quot;: [</td>
</tr>
<tr>
<td></td>
<td>&quot;itil&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;hr-admin&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;report-dev&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td><code>{object}.records.external_user</code></td>
<td>Array of strings where each string is the name of an externally defined user to set as an alias for the Now Platform user specified by the <code>{object}.records.mapping_value</code> parameter. Data type: Array. Values can be in any format, depending on the names of the externally defined user accounts selected for the user mapping. Examples include:</td>
</tr>
<tr>
<td></td>
<td>&quot;external_user&quot;: [</td>
</tr>
<tr>
<td></td>
<td>&quot;beth-anglin&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;ad\beth.anglin&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;beth-anglin@sharepoint&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td><code>{object}.records.mapping_value</code></td>
<td>Email field value that uniquely identifies an existing record in the User [sys_user] table. All externally defined user and group aliases in the request are mapped to the Now Platform user with this email address.</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="https://example.com" alt="Note" /> The API treats this parameter as the unique identifier for the user mapping record. If you import another user mapping with the same mapping_value as an existing user mapping record, the new record overwrites the existing record. Data type: String</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
## Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Object describing the error encountered during processing of the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;detail&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>error.detail</td>
<td>Details of the error encountered during processing of the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>error.message</td>
<td>Message for the error encountered during processing of the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>import_set_id</td>
<td>Sys_id for the new record created in the Import Set [sys_import_set] table by a successful request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>multi_import_set_id</td>
<td>Sys_id for the new record created in the Multi Import Set [sys_multi_import_set] table by a successful request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result</td>
<td>Result for an unsuccessful request. Includes a message describing the reason for the request failure.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>status</td>
<td>Status of an unsuccessful request.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• failure</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

Import externally defined user and group aliases for Now Platform users Beth Anglin and Abel Tuter into an AI Search user mapping table named `x_snc_sharepoint_user_table`.

```
curl -X POST
'https://instance.servicenow.com/api/now/v1/ais/external_content/user_mapping/import_multiple/x_snc_sharepoint_user_table' \
-H 'Accept: application/json' \
-H 'Content-Type: application/json' \
-u 'username': 'password' \
-d '{
  "records": [
    {
      "mapping_value": "beth.anglin@example.com",
      "external_user": [
        "ad\beth-anglin",
        "beth.anglin@sharepoint"
      ],
      "external_group": [
        "itil",
        "itil-admin",
        "itil-dev"
      ]
    },
    {
      "mapping_value": "abel.tuter@example.com",
      "external_user": [
        "ad\abel-tuter",
        "abel.tuter@sharepoint"
      ],
      "external_group": [
        "hr",
        "hr-admin",
        "hr-dev"
      ]
    }
  ]
}
```
The response shows the sys_ids for the generated Import Set and Multi Import Set records.

```json
{
  "import_set_id": "6e9ddb629d987010f877878bd9f0e9dd",
  "multi_import_set_id": "269ddb629d987010f877878bd9f0e9de"
}
```

**Example: cURL request**

Import invalid request containing an empty mapping record.

```cURL
curl 'https://instance.servicenow.com/api/now/v1/ais/external_content/user_mapping/import_multiple/u_ext_content_user_mapping' \   --request POST \   --user 'username': 'password' \   --header 'Accept: application/json' \   --header 'Content-Type: application/json' \   --data '{
  "records": [
    {
      "mapping_value": "beth.anglin@example.com",
      "external_user": [
        "ad\beth-anglin",
        "beth.anglin@sharepoint"
      ],
      "external_group": [
        "reports",
        "reports-admin",
        "reports-dev"
      ]
    },
    {}
  ]
}'
```

The request fails with status 400 and the response indicates the nature of the failure.
Example: Python request

Import external user and group aliases for Now Platform users Beth Anglin and Abel Tuter into an AI Search user mapping table named u_ext_content_user_mapping.

```python
# Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request
url = "https://instance.servicenow.com/api/now/v1/ais/external_content/user_mapping/import_multiple/u_ext_content_user_mapping"

# Set credentials
user = "username"
pwd = "password"

# Set HTTP headers
headers = {"Accept": "application/json", "Content-Type": "application/json"}

data = json.dumps(
    {
        "records": [
            {
                "mapping_value": "beth.anglin@example.com",
                "external_user": [
                "ad\beth-anglin",
                "beth.anglin@sharepoint"
                ],
                "external_group": [
                "itil",
                "itil-admin",
                "itil-dev"
                ]
            },
            {
                "mapping_value": "abel.tuter@example.com",
                "external_user": [
            ...
```
The response shows the sys_ids for the generated Import Set and Multi Import Set records.

```json
{
    "import_set_id": "3111eb2a9d987010f877878bd9f0e973",
    "multi_import_set_id": "f511eb2a9d987010f877878bd9f0e973"
}
```

**Alarm Management Open API**

Creates or updates data in the Events [em_event] table.

The telecommunications Alarm Management Open API (sn_ind_tmf642) is a ServiceNow® implementation of the TMforum Open API specification. This API is based on TMF642 Alarm Management API REST Specification Release 20, April 2020.

The calling user must have to access the sn_ind_tmf642.alarm_mgmt_integration endpoint.

This API is provided within the sn_ind_tmf642 namespace.
TMforum Open API specification requires a synchronous response for every inbound request it receives. Apply correlation and de-duplication in the source operational support system (OSS) before forwarding payloads to the system.

**Extending the Alarm Management Open API**

The Alarm Management Open API can be extended by editing the AlarmAPIProcessor script include or creating new script includes. Do not edit or create script includes without an understanding of the consequences of the changes.

- **AlarmAPIProcessor**: An empty script include file. Use this file to define any functions that you want to override from AlarmAPIProcessorOOB.

Extend the Alarm Management Open API to make the following customizations.

**Required parameters**

Alarm Management API utilizes JSON schemas to hold required parameters information. These JSON schemas are defined in the TMFAlarmAPIConstants script include. These schemas are used to validate whether request payloads are valid. These schemas are not referenced directly in script includes and are returned by the following methods in the AlarmAPIProcessorOOB script include:

- **getCreateEventSchema**: Returns the validating schema for creating an event.
- **getClearEventSchema**: Returns the validating schema for clearing an event alarm.
- **getPatchEventSchema**: Returns the validating schema for patching an event.

To override existing schemas, define new schemas in the AlarmAPIProcessor script include and override the AlarmAPIProcessorOOB script include methods.

```javascript
var AlarmAPIProcessor = Class.create();
AlarmAPIProcessor.prototype = Object.extendObject(AlarmAPIProcessorOOB, { // Define overriding functions here

    // Define getCreateEventSchema here to override OOTB method in AlarmAPIProcessorOOB
    getCreateEventSchema: function() {
        return JSON.parse(AlarmAPIProcessor.CUSTOMIZED_SCHEMA);
    },
    type: 'AlarmAPIProcessor'
});
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Request body validation

To perform additional validation on the request body, override AlarmAPIProcessorOOB functions. AlarmAPIProcessorOOB contains the following helper functions that return true by default. These functions are called in TMFAAlarmAPIUtil.

- `verifyCreateEventPayload()` - Called by `processCreateEvent()`.
- `verifyClearEventPayload()` - Called by `processClearEvent()`.
- `verifyPatchEventPayload()` - Called by `processPatchEvent()`.

If a helper function returns false, it stops the API operation. To apply custom validation, override AlarmAPIProcessorOOB helper functions with identically-named function names and parameters in AlarmAPIProcessor. The new AlarmAPIProcessor functions are called by TMFAAlarmAPIUtil to replace the default AlarmAPIProcessorOOB helper functions.

In this example, a function in AlarmAPIProcessor overrides a default function in AlarmAPIProcessorOOB to perform validation on the name attribute.

```javascript
var AlarmAPIProcessor = Class.create();
AlarmAPIProcessor.prototype = Object.extendObject(AlarmAPIProcessorOOB, {
    // Define overriding functions here

    verifyCreateEventPayload: function(eventPayload, apiResponseProcessor){
        // Returning false terminates the POST request
        // Make sure to assign error message and reason
        if (eventPayload.type != "unique") {
```
apiResponseProcessor.setMessage("Failed");
apiResponseProcessor.setReason("No reason needed");
return false;
}
},
type: 'AlarmAPIProcessor'
});

Additional REST operations

To create additional operations beyond the existing GET, PATCH, and POST operations, create additional scripted REST resources for the Alarm Management Open API. The logic of the new scripted REST resources should be consistent with the existing operations. Define functions for the new operations in TMFAlarmAPIUtil.

Field mapping

When creating or updating records, the API maps request body parameters to event record fields. When retrieving records, the API maps record fields to response object attributes. AlarmAPIProcessorOOB contains the following functions to map a PATCH or POST request body to a record in the Event [em_event] table.

• mapCreateAlarmObjectToEvent()
• mapClearAlarmObjectToEvent()
• mapPatchAlarmObjectToEvent()

AlarmAPIProcessorOOB contains the following functions to map an event GlideRecord to a JSON response object for GET, PATCH, or POST requests.

• modifyCreateEventResponse()
• modifyClearEventResponse()
• modifyPatchEventResponse()

Customize field mappings to add and retrieve data for additional Event [em_event] table fields or change the default field mappings. To work with mappings, create functions with identical names and parameters in AlarmAPIProcessor to override AlarmAPIProcessorOOB mapping functions. TMFAlarmAPIUtil uses these AlarmAPIProcessor functions to replace the default AlarmAPIProcessorOOB mapping functions.
In the following example, two `AlarmAPIProcessor` functions override the default functions in `AlarmAPIProcessorOOB` to change the Type field mapping and add a mapping for the Severity field.

```javascript
var AlarmAPIProcessor = Class.create();
AlarmAPIProcessor.prototype = Object.extendsObject(AlarmAPIProcessorOOB, {
    // Define overriding functions here
    mapCreateAlarmObjectToEvent: function(eventGr, eventObject) {
        // Override default mapping for the severity field
        eventGr.severity = "High";
    },

    modifyCreateEventResponse: function(eventGr, eventObject) {
        // Override default mapping for the type field
        eventGr.type = "Customized Event";
    },

    type: 'AlarmAPIProcessor'
});
```

**Alarm Management Open – POST – /sn_ind_tmf642/alarm_mgmt/alarm**

Creates an alarm as a new record in the Events (em_event) table and adds an ID to the Message key field.

The new alarm record this endpoint makes might result in creating an alert record based on the Event Management Alert management rules.

This endpoint writes the response body to the Additional Info field of the event record in the Events (em_event) table.

**URL format**

Default URL: `/api/sn_ind_tmf642/alarm_mgmt/alarm`

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type: String</th>
</tr>
</thead>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>affectedService</td>
<td>List of objects identifying one or more services affected by the alarm.</td>
</tr>
</tbody>
</table>
|                 | "affectedService": [{
|                 |   "id": "String",
|                 |   "href": "String"
|                 | },
<p>| affectedService.href | URL reference providing details of the affected service.                  |
| affectedService.id | Required. Provides the identifier of the service affected by the alarm. This value maps to the affected configuration item (CI) on the alert. |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alarmDetails</td>
<td>Additional details about the alarm.</td>
</tr>
<tr>
<td>alarmedObject</td>
<td>Required. Details of the alarming object.</td>
</tr>
<tr>
<td>alarmedObject.id</td>
<td>Required. Unique identifier of the alarming object. This value maps to a CI in the system.</td>
</tr>
<tr>
<td>alarmedObject.href</td>
<td>URL reference to get the details of the alarming object</td>
</tr>
<tr>
<td>alarmedObjectType</td>
<td>Type or class of object associated with the event.</td>
</tr>
<tr>
<td>alarmRaisedTime</td>
<td>Indicates the date and time at which the alarm occurs at its source.</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| alarmReportingTime        | Date and time at which the OSS reports its alarm. This value might differ from the `alarmRaisedTime` value. For example, if an element management system (EMS) maintains the alarm list:  
  - The `alarmRaisedTime` is the time of alarm detection by the network element (NE).  
  - The `alarmReportingTime` is the alarm report time of storage in the EMS alarm list.  
  Data type: String                                                                                                                                                                                                                                                                                                                                 |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
|                                            | "indicatorName": "String",
|                                            | "indicatorUnit": "String",
|                                            | "observedValue": "String",
|                                            | "thresholdCrossingDescription": "String",
|                                            | "thresholdId": "String",
|                                            | "thresholdRef": "String" |
|                                            | Data type: Object                                                                                                                                                                                          |
| crossedThresholdInformation.direction     | Indicates the threshold crossing direction. Possible values:  
|                                            | • UP  
|                                            | • DOWN  
|                                            | Data type: String                                                                                                                                                                                          |
| crossedThresholdInformation.granularity   | Granularity at which the indicator is evaluated for threshold crossing. For example, 5MINUTES.                                                                                                          
|                                            | Data type: String                                                                                                                                                                                          |
| crossedThresholdInformation.indicatorName | Name of the indicator that crosses the alarm threshold.                                                                                                                                                   
<p>|                                            | Data type: String                                                                                                                                                                                          |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crossedThresholdInformation.indicatorUnit</td>
<td>Measurement of the indicator corresponding to the crossed threshold. Data type: String</td>
</tr>
<tr>
<td>crossedThresholdInformation.observedValue</td>
<td>Number identifying the indicator crossing the threshold. Data type: String</td>
</tr>
<tr>
<td>crossedThresholdInformation.thresholdCrossingDescription</td>
<td>More threshold details of the threshold crossing alarm. Data type: String</td>
</tr>
<tr>
<td>crossedThresholdInformation.thresholdId</td>
<td>Required. Unique identifier of the threshold that caused the alarm. Data type: String</td>
</tr>
<tr>
<td>crossedThresholdInformation.thresholdRef</td>
<td>URL of the threshold object. Data type: String</td>
</tr>
<tr>
<td>externalAlarmId</td>
<td>Required. Unique identifier for the alarm from the source system posting the alarm. Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>Unique identifier for the system to identify and process</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Name</strong></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>perceivedSeverity</td>
<td>Required. Alarm severity. Possible values: • CRITICAL • MAJOR • MINOR • WARNING • CLEAR</td>
</tr>
<tr>
<td>plannedOutageIndication</td>
<td>Indicates that the managed object for this alarm is in a planned outage state. For example, planned maintenance or out-of-service. Data type: String</td>
</tr>
<tr>
<td>probableCause</td>
<td>Required. Identifies the most likely situation to trigger the alarm. Use with</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| alarmType           | to qualify the alarm.  
Data type: String                                                        |
| proposedRepairActions | If available to the system, provides a repair action to resolve the issue that triggers the alarm.  
Data type: String |
| serviceAffecting    | Flag that indicates whether the alarm affects service.  
Valid values:  
• true: Alarm affects service performance or availability.  
• false: Alarm does not affect service.  
Default: false |
| sourceSystemId      | Required. Instance ID of the source alarm.  
Data type: String |
| specificProblem     | Identifies the specific problem that triggers the alarm. Use with probableCause to qualify the alarm.  
Data type: String |
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>Defines the alarm state during its life cycle. Possible values: • RAISED • UPDATED • CLEARED Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>Bad request.</td>
</tr>
</tbody>
</table>

Possible reasons:
- Invalid payload. Missing required alarm ID.
- Invalid payload. Required field missing:<field name>.
- Invalid payload. Missing required object.
- Event not created, failed.
- Unable to clear event.
- Unable to update event.
- The event doesn’t exist.

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>affectedService</td>
<td>List of objects identifying one or more services affected by the alarm.</td>
</tr>
<tr>
<td></td>
<td>&quot;affectedService&quot;:[{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;href&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>affectedService.href</td>
<td>URL reference providing details of the affected service.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>affectedService.id</td>
<td>Provides the identifier of the</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>service</td>
<td>service affected by the alarm. This value maps to the affected configuration item (CI) on the alert.</td>
</tr>
<tr>
<td>data type:</td>
<td>String</td>
</tr>
<tr>
<td>alarmDetails</td>
<td>Additional details about the alarm.</td>
</tr>
<tr>
<td>data type:</td>
<td>String</td>
</tr>
<tr>
<td>alarmedObject</td>
<td>Details of the alarming object.</td>
</tr>
<tr>
<td></td>
<td>&quot;alarmedObject&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;href&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>},</td>
</tr>
<tr>
<td>data type:</td>
<td>Object</td>
</tr>
<tr>
<td>alarmedObject.id</td>
<td>Unique identifier of the alarming object. This value maps to a CI in the system.</td>
</tr>
<tr>
<td>data type:</td>
<td>String</td>
</tr>
<tr>
<td>alarmedObject.href</td>
<td>URL reference to get the details of the alarming object</td>
</tr>
<tr>
<td>data type:</td>
<td>String</td>
</tr>
<tr>
<td>alarmedObjectType</td>
<td>Type or class of object associated with the event.</td>
</tr>
<tr>
<td>data type:</td>
<td>String</td>
</tr>
<tr>
<td>alarmRaisedTime</td>
<td>Indicates the date and time at which the alarm occurs at its source.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>alarmReportingTime</td>
<td>Date and time at which the OSS reports its alarm. This value might differ from the <code>alarmRaisedTime</code> value. For example, if an element management system (EMS) maintains the alarm list:</td>
</tr>
<tr>
<td></td>
<td>• The <code>alarmRaisedTime</code> is the time of alarm detection by the network element (NE).</td>
</tr>
<tr>
<td></td>
<td>• The <code>alarmReportingTime</code> is the alarm report time of storage in the EMS alarm list.</td>
</tr>
<tr>
<td>alarmType</td>
<td>Alarm category.</td>
</tr>
<tr>
<td>crossedThresholdInformation</td>
<td>Details about the crossed threshold.</td>
</tr>
</tbody>
</table>

```json
"crossedThresholdInformation":
{
  "direction": "String",
  "granularity": "String",
  "indicatorName": "String",
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crossedThresholdInformation.direction</td>
<td>Indicates the threshold crossing direction. Possible values: • UP • DOWN</td>
</tr>
<tr>
<td>crossedThresholdInformation.granularity</td>
<td>Granularity at which the indicator is evaluated for threshold crossing. For example, 5MINUTES.</td>
</tr>
<tr>
<td>crossedThresholdInformation.indicatorName</td>
<td>Name of the indicator that crosses the alarm threshold.</td>
</tr>
<tr>
<td>crossedThresholdInformation.indicatorUnit</td>
<td>Measurement of the indicator corresponding to the crossed threshold.</td>
</tr>
</tbody>
</table>

Data type: Object
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crossedThresholdInformation.observedValue</td>
<td>Number identifying the indicator crossing the threshold.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>crossedThresholdInformation.thresholdCrossingDescription</td>
<td>More threshold details of the threshold crossing alarm.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>crossedThresholdInformation.thresholdCrossingId</td>
<td>Unique identifier of the threshold that caused the alarm.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>crossedThresholdInformation.thresholdRef</td>
<td>URL of the threshold object.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>externalAlarmId</td>
<td>Unique identifier for the alarm from the source system posting the alarm.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>Unique identifier for the system to identify and process the alarm payload as an event.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>perceivedSeverity</td>
<td>Alarm severity. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• CRITICAL</td>
</tr>
<tr>
<td></td>
<td>• MAJOR</td>
</tr>
<tr>
<td></td>
<td>• MINOR</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>plannedOutageIndication</td>
<td>Indicates that the managed object for this alarm is in a planned outage state. For example, planned maintenance or out-of-service.</td>
</tr>
<tr>
<td>probableCause</td>
<td>Identifies the most likely situation to trigger the alarm. Use with <code>alarmType</code> to qualify the alarm.</td>
</tr>
<tr>
<td>proposedRepairActions</td>
<td>If available to the system, provides a repair action to resolve the issue that triggers the alarm.</td>
</tr>
</tbody>
</table>
| serviceAffecting     | Flag that indicates whether the alarm affects service. Valid values:  
  • true: Alarm affects service performance or availability.  
  • false: Alarm does not affect service.                                                                                                           |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sourceSystemId</td>
<td>Instance ID of the source alarm. Data type: String</td>
</tr>
<tr>
<td>specificProblem</td>
<td>Identifies the specific problem that triggers the alarm. Use with probableCause to qualify the alarm. Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>Defines the alarm state during its lifecycle. Possible values: RAISED, UPDATED, CLEARED Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl "https://<instance>.service-now.com/api/sn_ind_tmf642/alarm_mgmt/alarm" \
--request POST \
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--data "{
    "href":"http://api/alarm/ROUTER_IF@Cisco-0000-0-0-0-0-0-0-0-- Xz0/00800",
    "externalAlarmId": "<external-alarm-id>",
    "alarmType": "QualityOfServiceAlarm",
    "type": "QualityOfServiceAlarm",
    "ackState": "acknowledged",
    "perceivedSeverity": "CRITICAL",
    "probableCause": "Threshold crossed",
    "specificProblem": "Inbound Traffic threshold crossed",
    "alarmedObjectType": "ROUTER",
    "alarmedObject":{
        "id": "vManage_000000",
    }
}"
```
"href": "http://api/alarmedobject/000000",

"sourceSystemId": "SOURCE_SYSTEM_vManage_00000_000_00",
"alarmDetails": "Software Failure on SD-WAN Controller vManage_000000",
"state": "RAISED",
"alarmChangedTime": "2017-08-15T07:04:15.666Z",
"proposedRepairActions": "Switch in standby equipment",
"alarmReportingTime": "2017-06-15T07:04:15.666Z",
"alarmRaisedTime": "2020-09-15T07:04:15.666Z",
"plannedOutageIndication": "IN_SERVICE",
"serviceAffecting": true,
"affectedService": [
{
   "id": "SD WAN Enterprise Solutions",
   "href": "http://api/service/vlan_dot0_dot0"
},
{
   "id": "Santa Clara SD WAN Application Service",
   "href": "http://api/service/vlan_dot0_dot0"
}
]

---

Output:

{
   "href": "http://api/alarmedobject/ROUTER_IF@Cisco-0000-0-0-0-0-0-0-X0/00/00",
   "externalAlarmId": "<external-alarm-id>",
   "alarmType": "QualityOfServiceAlarm",
   "type": "QualityOfServiceAlarm",
   "ackState": "acknowledged",
   "perceivedSeverity": "CRITICAL",
   "probableCause": "Threshold crossed",
   "specificProblem": "Inbound Traffic threshold crossed",
   "alarmedObjectType": "ROUTER",
   "alarmedObject": {
      "id": "vManage_000000",
      "href": "http://api/alarmedobject/000000"
   },
   "sourceSystemId": "SOURCE_SYSTEM_vManage_00000_000_00",
   "alarmDetails": "Software Failure on SD-WAN Controller vManage_000000",
   "state": "RAISED",
   "alarmChangedTime": "2017-08-15T07:04:15.666Z",
}
Alarm Management Open – POST – /sn_ind_tmf642/alarm_mgmt/alarm/{id}/clear

Clears a specified event alarm and closes any corresponding alert associated with the event in Event Management.

**URL format**

Default URL: /api/sn_ind_tmf642/alarm_mgmt/alarm/{id}/clear

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>Required. Unique ID in the Message key field of a record in the Events (em_events) table. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alarmClearedTime</td>
<td>Date and time at which the alarm clears at the source. This value is in the Time of event field of the record in the Events (em_events) table. Data type: String</td>
</tr>
<tr>
<td>clearSystemId</td>
<td>Required. Name of the instance from which the alarmCleared operation was invoked. Data type: String</td>
</tr>
<tr>
<td>clearUserId</td>
<td>Required. Unique ID of the who invoked the alarmCleared operation. This value is in the Source field of the record in the Events (em_events) table. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
## Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
| 400         | Bad request. Possible reasons:  
  • Invalid payload. Missing required alarm ID.  
  • Invalid payload. Required field missing:<field name>.  
  • Invalid payload. Missing required object.  
  • Event not created, failed.  
  • Unable to clear event.  
  • Unable to update event.  
  • The event doesn’t exist. |

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| alarmClearedTime| Date and time at which the alarm clears at the source. This value is in the `Time of event` field of the record in the Events (em_events) table.  
  Data type: String |
| clearSystemId   | Name of the instance from which the `alarmCleared` operation was invoked.  
  Data type: String |
| clearUserId     | Unique ID of the who invoked the `alarmCleared` operation. This value is in the `Source` field of the record in the Events (em_events) table.  
  Data type: String |
<p>| href            | URL of the alarm.                                                                                                                            |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Unique ID in the Message key field of a record in the Events (em_events) table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

The following example shows how to clear an alarm.

```
curl
  "https://<instance>.service-now.com/api/sn_ind_tmf642/alarm_mgmt/alarm/6e116bb8db09201087b9a8394b961950_<external-alarm-id>/clear" \ 
  --request POST \ 
  --header "Accept:application/json" \ 
  --header "Content-Type:application/json" \ 
  --data "{
  "alarmClearedTime": "2017-08-15 06:04:15",
  "clearSystemId": "<source_instance_name>",
  "clearUserId": "SOURCE_SYSTEM_vManage_00000_000_00"
}"
  --user 'user':'password'
```

**Output:**

```
{
  "result": {
    "alarmClearedTime": "2017-08-15 06:04:15",
    "clearSystemId": "<source_instance_name>",
    "clearUserId": "SOURCE_SYSTEM_vManage_00000_000_00",
    "id": "6e116bb8db09201087b9a8394b961950_<external-alarm-id>"
  }
}
```

**Alarm Management Open – PATCH – /sn_ind_tmf642/alarm_mgmt/alarm/{id}**

Creates an event and update any associated alert record in Event Management.

This endpoint writes the response body to the Additional Info field of the event record in the Events (em_event) table.

**URL format**

Default URL: /api/sn_ind_tmf642/alarm_mgmt/alarm/{id}
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>Required. Unique ID in the Message key field of a record in the Events (em_events) table.</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>affectedService</td>
<td>List of objects identifying one or more services affected by the alarm.</td>
</tr>
<tr>
<td></td>
<td>&quot;affectedService&quot;:[</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;href&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>affectedService.href</td>
<td>URL reference providing details of the affected service.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>affectedService.id</td>
<td>Required. Provides the identifier of the service affected by the alarm. This value maps to the affected configuration item (CI) on the alert. Data type: String</td>
</tr>
<tr>
<td>alarmChangedTime</td>
<td>Indicates the last date and time the alarm changes on the alarm-owning system. Any change to the alarm updates this value whether coming from the alarmed resource or triggered by a change from the client. Data type: String</td>
</tr>
<tr>
<td>alarmDetails</td>
<td>Additional details about the alarm. Data type: String</td>
</tr>
<tr>
<td>alarmEscalation</td>
<td>Flag that indicates an escalation in alarm severity. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: There is an escalation in alarm severity.</td>
</tr>
<tr>
<td></td>
<td>• false: The alarm severity is the same or lower.</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crossedThresholdInformation</td>
<td>Details about the crossed threshold.</td>
</tr>
<tr>
<td>crossedThresholdInformation.direction</td>
<td>Indicates the threshold crossing direction. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• UP</td>
</tr>
<tr>
<td></td>
<td>• DOWN</td>
</tr>
<tr>
<td>crossedThresholdInformation.granularity</td>
<td>Granularity at which the indicator</td>
</tr>
</tbody>
</table>

```json
"crossedThresholdInformation": {
  "direction": "String",
  "granularity": "String",
  "indicatorName": "String",
  "indicatorUnit": "String",
  "observedValue": "String",
  "thresholdCrossingDescription": "String",
  "thresholdId": "String",
  "thresholdRef": "String"
}
```

Data type: Object
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crossedThresholdInformation.indicatorName</td>
<td>Name of the indicator that crosses the alarm threshold. Data type: String</td>
</tr>
<tr>
<td>crossedThresholdInformation.indicatorUnit</td>
<td>Measurement of the indicator corresponding to the crossed threshold. Data type: String</td>
</tr>
<tr>
<td>crossedThresholdInformation.observedValue</td>
<td>Number identifying the indicator crossing the threshold. Data type: String</td>
</tr>
<tr>
<td>crossedThresholdInformation.thresholdCrossingDescription</td>
<td>More threshold details of the threshold crossing alarm. Data type: String</td>
</tr>
<tr>
<td>crossedThresholdInformation.thresholdId</td>
<td>Required. Unique identifier of the threshold that caused the alarm. Data type: String</td>
</tr>
<tr>
<td>crossedThresholdInformation.thresholdRef</td>
<td>URL of the threshold object. Data type: String</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>href</td>
<td>URL reference to the alarm.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>perceivedSeverity</td>
<td>Required. Alarm severity.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• CRITICAL</td>
</tr>
<tr>
<td></td>
<td>• MAJOR</td>
</tr>
<tr>
<td></td>
<td>• MINOR</td>
</tr>
<tr>
<td></td>
<td>• WARNING</td>
</tr>
<tr>
<td></td>
<td>• CLEAR</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>plannedOutageIndication</td>
<td>Indicates that the managed object for this alarm is in a planned outage</td>
</tr>
<tr>
<td></td>
<td>state. For example, planned maintenance or out-of-service.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>probableCause</td>
<td>Required. Identifies the most likely situation to trigger the alarm. Use</td>
</tr>
<tr>
<td></td>
<td>with alarmType to qualify the alarm.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>proposedRepairActions</td>
<td>If available to the system, provides a repair action to resolve the issue.</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>that triggers the alarm.</td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>serviceAffecting</td>
<td>Flag that indicates whether the alarm affects service.</td>
</tr>
<tr>
<td>Valid values:</td>
<td></td>
</tr>
<tr>
<td>• true: Alarm affects service performance or availability.</td>
<td></td>
</tr>
<tr>
<td>• false: Alarm does not affect service.</td>
<td></td>
</tr>
<tr>
<td>Default: false</td>
<td></td>
</tr>
<tr>
<td>specificProblem</td>
<td>Identifies the specific problem that triggers the alarm. Use with probableCause to qualify the alarm.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml.</td>
</tr>
<tr>
<td></td>
<td>Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad request. Possible reasons:</td>
</tr>
<tr>
<td></td>
<td>• Invalid payload. Missing required alarm ID.</td>
</tr>
<tr>
<td></td>
<td>• Invalid payload. Required field missing:&lt;field name&gt;.</td>
</tr>
<tr>
<td></td>
<td>• Invalid payload. Missing required object.</td>
</tr>
<tr>
<td></td>
<td>• Event not created, failed.</td>
</tr>
<tr>
<td></td>
<td>• Unable to clear event.</td>
</tr>
<tr>
<td></td>
<td>• Unable to update event.</td>
</tr>
<tr>
<td></td>
<td>• The event doesn’t exist.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alarmChangedTime</td>
<td>Date and time the alarm changed in GMT on the alarm-owning system. This value is in GlideDateTime format and is in the Time of event field of a record in the Events (em_events) table. This value updates when changes to the alarm from the alarmed resource occur or a change from a client update triggers the alarm. This value is required in the response. Data type: String</td>
</tr>
<tr>
<td>href</td>
<td>URL reference to the alarm.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>id</td>
<td>Unique identifier of the alarm. Value determined by the system owning the alarm. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

In the following example, the severity is reduced to MINOR.

```
curl  
  "https://<instance>.service-now.com/api/sn_ind_tmf642/alarm_mgmt/alarm/6e116bb8db09201087b9a8394b961950_<external-alarm-id>" \  
  --request PATCH \  
  --header "Accept:application/json" \  
  --header "Content-Type:application/json" \  
  --data "{
    "href":"http://api/alarm/ROUTER_IF@Cisco-0000-0-0-0-0-0-0--Xz0/000000", 
    "externalAlarmId":"<external-alarm-id>", 
    "alarmType":"QualityOfServiceAlarm", 
    "type":"QualityOfServiceAlarm", 
    "ackState":"acknowledged", 
    "perceivedSeverity":"MINOR", 
    "probableCause":"Threshold crossed", 
    "specificProblem":"Inbound Traffic threshold crossed", 
    "alarmedObjectType":"ROUTER", 
    "alarmedObject":{
      "id":"vManage_000000", 
      "href":" http://api/alarmedobject/000000" 
    }, 
    "sourceSystemId":"SOURCE_SYSTEM_vManage_00000_000_00", 
    "alarmDetails":"Software Failure on SD-WAN Controller vManage_000000", 
    "alarmEscalation": false, 
    "state":"RAISED", 
    "alarmChangedTime":"2017-08-15T07:04:15.666Z", 
    "proposedRepairActions":"Switch in standby equipment", 
    "alarmReportingTime":"2017-06-15T07:04:15.666Z", 
    "alarmRaisedTime":"2020-09-15T07:04:15.666Z", 
    "plannedOutageIndication":"IN_SERVICE", 
    "serviceAffecting":true, 
    "affectedService":[
      
      "id":"SD WAN Enterprise Solutions", 
    ]
  }  
```
Application Service API

Create, modify, and update application services using REST APIs. These APIs require that the user have the Application Service administrator [app_service_admin] role.

Application Service - GET /now/cmdb/app_service/{sys_id}/getContent

Retrieves a list of configuration items (CI) in an application service and the relationships between them. Only retrieves content for services that were manually created.

URL format

Default URL: /api/now/cmdb/app_service/{sys_id}/getContent

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>sys_id</td>
</tr>
</tbody>
</table>

Output:

```
{
  "result": {
    "alarmChangedTime": "2017-08-15T07:04:15.666Z",
    "id": "6e116bb8db09201087b9a8394b961950_<external-alarm-id>"
  }
}
```
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| Mode   | Amount of relationship data to return. Valid values:  
  • Shallow: Return parent and child relationships between CIs within a service.  
  • Full: Return CI attributes and lookup information in addition to the data returned in shallow mode.  
  The values are not case-sensitive.  
  Data type: String |

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or they do not have the app_service_admin role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmdb</td>
<td>List of objects that describe the CIs associated with the specified application service. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;cmdb&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;items&quot;: [Object],</td>
</tr>
<tr>
<td></td>
<td>&quot;relations&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>cmdb.items</td>
<td>CIs within the application service. If the full query parameter was passed in the request, this section includes attributes and lookup information for each CI. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;items&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;className&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;values&quot;: (Object)</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>cmdb.items.className</td>
<td>Name of the class that contains the CI. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>cmdb.items.values</td>
<td>Information to use to locate an associated CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;values&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>cmdb.items.values.name</td>
<td>Name of the CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>cmdb.items.values.sys_id</td>
<td>Sys_id of the CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>cmdb.relations</td>
<td>Relationship data for associated CIs, such as hosts and their children.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>service</td>
<td>List of services related to the identified service.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;service&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;service_relations&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>service.name</td>
<td>Name of the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>service.service_relations</td>
<td>Hierarchy data for the CIs within the application service. All CIs form pairs with a parent and child CI. The top-level CI, referred to as the entry point of an application service, does not have a parent CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;service_relations&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;child&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;parent&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| service.service_relations.child | Name of a child CI related to the CI.  
Data type: String |
| service.service_relations.parent | Name of a parent CI related to the CI.  
Data type: String |
| service.url | Relative path to the application service.  
Data type: String |

**Example: Sample cURL request**

curl

```
"https://instance.servicenow.com/api/now/cmdb/app_service/2fce42d80a0a0bb4004af34d7e3984c8/getContent?mode=shallow" \ 
--request GET \ 
--header "Accept:application/json" \ 
--user 'username':'password'
```

```
{
  "service": {
    "name": "PeopleSoft Portals",
    "url": "/api/now/table/cmdb_ci_service_discovered/2fce42d80a0a0bb4004af34d7e3984c8",
    "service_relations": [
      {
        "parent": "",
        "child": "3a2810c20a0a0bb400268337d6e942ca"
      },
      {
        "parent": "3a27f1520a0a0bb400ecd6ff7afcf036",
        "child": "3a5dd3dbc0a8ce0100655f1ec66ed42c"
      },
      {
        "parent": "3a290cc60a0a0bb400000bdb386af1cf",
        "child": "3a307c930a0a0bb400353965d0b8861f"
      },
      {
        "parent": "3a172e820a0a0bb40034228e9f65f1be",
        "child": "3a27d4370a0a0bb4006316812bf45439"
      }
    ]
  }
```

© 2021 ServiceNow, Inc. All rights reserved.  
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.  
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
"parent": "",
"child": "3a172e820a0a0bb40034228e9f65f1be"
},
{
  "parent": "",
  "child": "3a27f1520a0a0bb400ecd6ff7afcf036"
},
{
  "parent": "3a2810c20a0a0bb400268337d6e942ca",
  "child": "3a290cc60a0a0bb400000b6b386af1cf"
}]
},
"cmdb": {
  "relations": [],
  "items": [
    {
      "values": {
        "sys_id": "3a172e820a0a0bb40034228e9f65f1be",
        "name": "PS LoadBal01"
      },
      "className": "cmdb_ci_win_server"
    },
    {
      "values": {
        "sys_id": "3a2810c20a0a0bb400268337d6e942ca",
        "name": "PS Apache03"
      },
      "className": "cmdb_ci_web_server"
    },
    {
      "values": {
        "sys_id": "55b35562c0a8010e01cff22378e0aea9",
        "name": "ny8500-nbxs08"
      },
      "className": "cmdb_ci_netgear"
    },
    {
      "values": {
        "sys_id": "3a27f1520a0a0bb400ecd6ff7afcf036",
        "name": "PS Apache02"
      },
      "className": "cmdb_ci_web_server"
    }
  ]
}
Example: Sample Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/cmdb/app_service/2fce42d80a0a0bb4004af34d7e3984c8/getContent?mode=full'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```
<assigned>2017-04-25 08:00:00</assigned>
<capacity>1.8</capacity>
<category>Do not migrate to asset</category>
<cost>8999.99</cost>
<device_type>SAN</device_type>
<disk_size>250</disk_size>
<install_date>2017-03-06 07:00:00</install_date>
<name>Storage Area Network 001</name>
<sys_id>109562a3c611227500a7b7ff98cc0dc7</sys_id>
</values>
</element>

<element>
<className>cmdb_ci_linux_server</className>
<lookup />
<values>
<asset_tag>P1000207</asset_tag>
<assigned>2017-07-08 08:00:00</assigned>
<category>Do not migrate to asset</category>
<cost>45557.5</cost>
<cpu_count>1</cpu_count>
<cpu_speed>2800</cpu_speed>
<cpu_type>Intel</cpu_type>
<disk_space>40</disk_space>
<install_date>2017-04-20 08:00:00</install_date>
<name>PS LinuxApp02</name>
<os>Linux Red Hat</os>
<os_version>2.6.9-22.0.1.ELsmp</os_version>
<ram>2048</ram>
<sys_id>3a5dd3d630a8ce0100655f1ec66ed4dc</sys_id>
</values>
</element>

<element>
<className>cmdb_ci_web_server</className>
<lookup />
<values>
<name>PS Apache01</name>
<sys_id>3a27d4370a0a0bb4006316812bf45439</sys_id>
<type>Apache</type>
<version>6.0</version>
</values>
</element>
</items>
<relations />
</cmdb>
<service>
  <name>PeopleSoft Portals</name>
  <service_relations>
    <element>
      <child>3a2810c20a0a0bb400268337d6e942ca</child>
      <parent />
    </element>
    <element>
      <child>3a5dd3dbc0a8ce0100655f1ec66ed42c</child>
      <parent>3a27f1520a0a0bb400ecd6ff7afcf036</parent>
    </element>
    <element>
      <child>3a307c930a0a0bb400353965d0b8861f</child>
      <parent>3a290cc60a0a0bb4000000bdb386af1cf</parent>
    </element>
    <element>
      <child>3a27d4370a0a0bb400342288e9f65f1be</child>
      <parent>3a172e820a0a0bb400342288e9f65f1be</parent>
    </element>
    <element>
      <child>3a172e820a0a0bb400342288e9f65f1be</child>
      <parent />
    </element>
    <element>
      <child>3a27f1520a0a0bb400ecd6ff7afcf036</child>
      <parent />
    </element>
  </service_relations>
  <url>/api/now/table/cmdb_ci_service_discovered/2fce42d80a0a0bb4004af34d7e3984c8</url>
</service>
</root>

Application Service - GET /now/cmdb/csdm/app_service/find_service

Returns basic information about a specified application service and its associated business context.

⚠️  Note: This endpoint has been deprecated, use SG Services – POST – /sg_services/app_service/find instead.

URL format

Default URL: /api/now/cmdb/csdm/app_service/find_service
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| name | Required, if **number** is not passed in. Name of the service for which to return information. Located in the Application Service [cmdb_ci_service_auto] table.  
Data type: String |
| number | Required, if **name** is not passed in. Unique number that identifies the application service for which to return information. Located in the Application Service [cmdb_ci_service_auto] table.  
Data type: String |

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <strong>application/json</strong>.</td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or they do not have the app_service_admin role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| error            | Describes an encountered error.  
Data type: Object  
"error": {  
  "details": "String",  
  "message": "String"  
} |
| error.details    | Additional information about the error.  
Data type: String |
| error.message    | Message describing the error.  
Data type: String |
| services         | Array of objects, each object containing details of services associated with the specified application service.  
Data type: Array |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>services.environment</code></td>
<td>Environment in which the application service is deployed, such as &quot;Production&quot; or &quot;Test Lab 2.&quot; This can be any value that makes sense to the implementer. Data type: String Maximum length: 40 Default: null</td>
</tr>
<tr>
<td><code>services.name</code></td>
<td>Required. Name of the application service. Must be a unique name from all other application services. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td><code>services.number</code></td>
<td>Unique number that identifies the application service. This value is initially generated by the endpoint. Data type: String</td>
</tr>
<tr>
<td><code>services.relationships</code></td>
<td>Service types associated with the application service. Data type: Object</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>services.relationships.business_app</td>
<td>List of sys_ids of the business applications related to the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>services.relationships.technical_service_offering</td>
<td>List of sys_ids of the business service offerings related to the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>services.relationships.business_service_offering</td>
<td>List of sys_ids of the technical service offerings related to the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>services.sys_id</td>
<td>Sys_id of the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>services.version</td>
<td>Application service version number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Format: User definable</td>
</tr>
<tr>
<td></td>
<td>Default: null</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

```bash
curl -X GET \
https://instance.servicenow.com/api/now/cmdb/csdm/app_service/find_service?name=Email_East &number=SNSVC0001018 \
-H 'Accept: application/json' \
-u 'username':'password'
```
Application Service - POST /now/cmdb/app_service/create

Creates an application service or update an existing application service.

URL format
Default URL: /api/now/cmdb/app_service/create

Supported request parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
</tbody>
</table>

Request body

The API accepts these JSON elements in the request body. In addition, you can add any field from the Mapped Application Service [cmdb_ci_service_discovered] table to pass data to the application service record. For example, to add content to the Comments field, add "comments": "Passing data to the Comments field" to the JSON request.
Elements accepted in the request body

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the application service you wish to create or update.</td>
</tr>
<tr>
<td>service_relations</td>
<td>Information about CIs and the hierarchy they form. All CIs form pairs with a parent CI and a child CI. The top-level CI, referred to as the entry point of an application service, does not have a parent CI.</td>
</tr>
<tr>
<td>parent</td>
<td>Sys_id of the CI whose outgoing connection joins this CI with the CI at the lower level in the map hierarchy.</td>
</tr>
<tr>
<td>child</td>
<td>Sys_id of the CI whose incoming connection joins this CI with the CI located of the higher level in the map hierarchy.</td>
</tr>
</tbody>
</table>

Response body
The API returns these JSON elements in the response body.

Elements returned in the response body

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>URL of the standard table REST API for showing the sys_id of an application service record. This element appears only if the endpoint is successful. Data type: String</td>
</tr>
<tr>
<td>getContentUrl</td>
<td>REST API path for getting the application service content. This element appears only if the endpoint is successful. Data type: String</td>
</tr>
<tr>
<td>info</td>
<td>Number of CIs added to the application service. This element appears only if the endpoint is successful. Data type: String</td>
</tr>
<tr>
<td>error</td>
<td>Error message details. This element appears only if the endpoint fails. Data type: Object</td>
</tr>
<tr>
<td>message</td>
<td>Error message describing the nature of the failure.</td>
</tr>
</tbody>
</table>
Elements returned in the response body (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This element appears only if the endpoint fails. Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>States &quot;failure&quot; if the REST API fails. This element appears only if the endpoint fails. Data type: String</td>
</tr>
</tbody>
</table>

Example: Sample cURL request

```
curl "http://<instance.servicenow.com>/api/now/cmdb/app_service/create" \  
--request POST  
--header "Accept:application/json" \  
--header "Content-Type:application/json" \  
--data "{  
"name": "myTestAppService",  
"comments": "Passing data to the Comments field",  
"service_relations":  
[  
  {  
    "parent": "",  
    "child": "3a70f789c0a8ce010091b0ea635b982a"  
  },  
  {  
    "parent": "3a70f789c0a8ce010091b0ea635b982a",  
    "child": "b4fd7c8437201000deeabfc8bcbe5dc1"  
  },  
  {  
    "parent": "b4fd7c8437201000deeabfc8bcbe5dc1",  
    "child": "53fdbc8437201000deeabfc8bcbe5d10"  
  ]  
}  
"  
--user 'admin': 'admin'
```

```json
{
  "result": {
    "url": "http://<instance.servicenow.com>/api/now/table/cmdb_ci_service_discovered/b1cff89fc9e19300964fcb1ac3059a5",
    "getContentView": "http://<instance.servicenow.com>/api/now/cmdb/app_service/b1cff89fc9e19300964fcb1ac3059a5/getContent",
    "info": "3 CIs added to service"
  }
}```
Example: Sample Python request

```python
# Need to install requests package for python
import requests

# Set the request parameters
url = 'http://instance.service-now.com/api/now/cmdb/app_service/create

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {"Content-Type":"application/json","Accept":"application/json"}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, data="{
  "name": "myTestAppService",
  "comments": "Passing data to the Comments field",
  "service_relations": [
    {"parent": "",
     "child": "3a70f789c0a8ce010091b0ea635b982a"},
    {"parent": "3a70f789c0a8ce010091b0ea635b982a",
     "child": "b4fd7c8437201000deabf88bcbe5dc1"},
    {"parent": "b4fd7c8437201000deabf88bcbe5dc1",
     "child": "53fdbc8437201000deabf88bcbe5d10"}
  ]
}
"

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<root>
  <result>
    <getContentUrl>/api/now/cmdb/app_service/b1cff89fc9e19300964fcbb1ac3059a5/getContent</getContentUrl>
  </result>
</root>
```
Application Service - POST /now/cmdb/csdm/app_service/register_service

Creates an application service record stub in the Application Service [cmdb_ci_service_auto] table and populates it with the passed in information.

Before storing the record, it validates the passed parameters and the uniqueness of the application service name. It also creates upstream relationships which provide business context to the service by associating the specified business applications, business service offerings, and technical service offerings to the service application. It does not create downstream relationships to infrastructure. To define downstream relationships you must call the Application Service - PUT /now/cmdb/csdm/app_service/{service_sys_id}/populate_service endpoint.

⚠️ Note: This endpoint has been deprecated, use SG Services – POST – /sg_services/app_service/register instead.

URL format

Default URL: /api/now/cmdb/csdm/app_service/register_service

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>
### Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_details</td>
<td>Required. Details of the service associated with the specified application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;basic_details&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;environment&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;version&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>basic_details.environment</td>
<td>Environment in which the application service is deployed, such as &quot;Production&quot; or &quot;Test Lab 2.&quot; This can be any value that makes sense to the implementer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: null</td>
</tr>
<tr>
<td>basic_details.name</td>
<td>Required. Name of the application service. Must be a unique name from all other application services.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>basic_details.version</td>
<td>Application service version number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Format: User definable</td>
</tr>
<tr>
<td></td>
<td>Default: null</td>
</tr>
<tr>
<td>relationships</td>
<td>Upstream (parent) service types to associate with the application service. These relationships are stored in the CI Relationships [cmdb_rel_ci] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
</tbody>
</table>
Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;relationships&quot;:</td>
<td></td>
</tr>
<tr>
<td>&quot;business_app&quot;: [Array],</td>
<td></td>
</tr>
<tr>
<td>&quot;business_service_offering&quot;: [Array],</td>
<td></td>
</tr>
<tr>
<td>&quot;technical_service_offering&quot;: [Array]</td>
<td></td>
</tr>
</tbody>
</table>

| relationships.business_app                | List of sys_ids of the business applications to relate to the application service. |
|                                          | Data type: Array                                                             |
| relationships.business_service_offering  | List of sys_ids of the business service offerings to relate to the application service. |
|                                          | Data type: Array                                                             |
| relationships.technical_service_offering | List of sys_ids of the technical service offerings to relate to the application service. |
|                                          | Data type: Array                                                             |

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Successful. The application service was successfully registered.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or they do not have the app_service_admin role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Describes an encountered error. Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>error.details</td>
<td>Additional information about the error. Data type: String</td>
</tr>
<tr>
<td>error.message</td>
<td>Message describing the error. Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service. This number is generated by the endpoint.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td>service_sys_id</td>
<td>Sys_id of the registered application service.</td>
</tr>
</tbody>
</table>

Example: Sample cURL request

```bash
curl -X POST \
https://instance.servicenow.com/api/now/cmdb/csdm/app_service/register_service \
-H 'Accept: application/json' \
-H 'Content-Type: application/json' \
-u 'username': 'password' \
-d '{
  "basic_details": {
    "environment": "Test Lab 2",
    "name": "Test Lab 2",
    "version": "1.1.1"
  },
  "relationships": {
    "business_app": ["f07c7620db54101037740f95ca96199a"],
    "business_service_offering": ["069c7e20db54101037740f95ca9619bf"]
  }
}'
```

```json
{
  "result": {
    "service_sys_id": "80f21bbadb4d103d3a93c3e296199a",
    "number": "SNSVC0001018"
  }
}
```

Application Service - PUT /now/cmdb/csdm/app_service/{service_sys_id}/populate_service

Populates the specified application service with a specified set of downstream configuration items (CI).

Before calling this endpoint, you must call the POST /now/cmdb/csdm/app_service/register_service endpoint to create the application service.

⚠️ Note: This endpoint has been deprecated, use SG Services – POST – /sg_services/app_service/populate instead.

**URL format**

Default URL: /api/now/cmdb/csdm/app_service/{service_sys_id}/populate_service
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>service_sys_id</td>
<td>Sys_id of the application service to populate with the specified CIs. Located in the Application Service [cmdb_ci_service_auto] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>population_method</td>
<td>CIs to associate with the specified application service. The elements in the populate_method object depend on the type of information being populated; defined by the type parameter. Data type: Object</td>
</tr>
</tbody>
</table>

For **type** = cmdb_group_based

```json
{
  "population_method": {
    "group_id": "String",
    "type": "String"
  }
}
```

For **type** = discovery

**: Note:** To use this type you must activate the Service Mapping (com.snc.service-mapping) plugin.

```json
{
  "population_method": {
```

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>population_method.attributes</td>
<td>List of attributes to associate with the CI, such as host, port, name, and URL. For additional information on entry points, see Entry point attributes.</td>
</tr>
<tr>
<td>population_method.attributes.name</td>
<td>Name of the attribute, such as host, port, name, and URL. Data type: String</td>
</tr>
<tr>
<td>population_method.attributes.value</td>
<td>Attribute value. Data type: String</td>
</tr>
</tbody>
</table>

For `type = tag_list`

위의 사용자 정의 `type`를 사용하기 위해서는 `Service Mapping (com.snc.service-mapping)` 플러그인을 활성화해야 합니다.

```json
{
    "population_method": {
        "tags": [Array],
        "type": "String"
    }
}
```

**Note:** To use this type you must activate the Service Mapping (com.snc.service-mapping) plugin.

**Note:** You must pass all required attributes.

```json
{
    "population_method": {
        "tags": [Array],
        "type": "String"
    }
}
```
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>population_method.entry_point</td>
<td>Required if <code>population_method.type = discovery</code>. Entry point into the application service. For additional information on entry points, see Entry point attributes and Create entry point types for Service Mapping. Data type: String</td>
</tr>
<tr>
<td>population_method.group_id</td>
<td>Required if <code>population_method.type = cmdb_group_based</code>. Sys_id of the group to associate with the CI. Located in the CMDB Group [cmdb_group] table. Data type: String</td>
</tr>
<tr>
<td>population_method.tags</td>
<td>List of tags to associate with the CI. This information is located in the Key Values [cmdb_key_value] table. Data type: Array</td>
</tr>
<tr>
<td></td>
<td><code>{  &quot;tags&quot;: {  &quot;tag&quot;: &quot;String&quot;,  &quot;value&quot;: &quot;String&quot; } }</code></td>
</tr>
<tr>
<td>population_method.tags.tag</td>
<td>Name of the tag. Data type: String</td>
</tr>
<tr>
<td>population_method.tags.value</td>
<td>Tag value. Data type: String</td>
</tr>
<tr>
<td>population_method.type</td>
<td>Required. Type of CI to populate. Valid values:  • cmdb_group_based  • discovery  • tag_list Data type: String</td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or they do not have the app_service_admin role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Describes an encountered error. Data type: Object</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>error.details</td>
<td>Additional information about the error. Data type: String</td>
</tr>
<tr>
<td>error.message</td>
<td>Message describing the error. Data type: String</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

The following code example shows how to populate an application service when the CI type is `cmdb_group_based`.

```cURL
curl -X PUT \
https://instance.servicenow.com/api/now/cmdb/csdm/app_service/f809cb60db94101037740f95ca96190f/populate_service \
-H 'Accept: application/json' \
-H 'Content-Type: application/json' \
-u 'username':'password' \
-d '{
    "population_method": {
        "type": "cmdb_group_based",
        "group_id": "e809cf60db94101037640f95ca961932"
    }
}'
{
    "result": {}
}
```

The following code example shows how to populate a discovery application service.

```cURL
curl -X PUT \
https://instance.servicenow.com/api/now/cmdb/csdm/app_service/f809cb60db94101037740f95ca96190f/populate_service \
-H 'Accept: application/json' \
-H 'Content-Type: application/json' \
-u 'username':'password' \
```
Application Service - PUT /now/cmdb/csdm/app_service/{service_sys_id}/service_details

Updates the specified application service with the passed in basic information.

It also validates the uniqueness of the application service by checking the passed in information against the Identification and Reconciliation engine (IRE) rules defined on the application service class.

⚠️ Note: This endpoint has been deprecated, use SG Services – POST – /sg_services/app_service/update instead.

URL format

Default URL: /api/now/cmdb/csdm/app_service/{service_sys_id}/service_details
### Supported request parameters

#### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>service_sys_id</td>
<td>Sys_id of the application service to updated. Located in the Application Service [cmdb_ci_service_auto] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

#### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

#### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_details</td>
<td>Details of the service associated with the specified application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;basic_details&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;environment&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>basic_details.environment</td>
<td>Environment in which the application service is deployed, such as &quot;Production&quot; or &quot;Test Lab 2.&quot; This can be any value that makes sense to the implementer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: null</td>
</tr>
<tr>
<td>basic_details.name</td>
<td>Required. Name of the application service. Must be a unique name from all other application services.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>relationships</strong></td>
<td>Service types to associate with the application service. Located in the CI</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;relationships&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;business_app&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;business_service_offering&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;technical_service_offering&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><strong>relationships.business_app</strong></td>
<td>List of sys_ids of the business applications to relate to the application</td>
</tr>
<tr>
<td></td>
<td>service.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td><strong>relationships.business_service_offering</strong></td>
<td>List of sys_ids of the business service offerings to relate to the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td><strong>relationships.technical_service_offering</strong></td>
<td>List of sys_ids of the technical service offerings to relate to the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>
Request headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or they do not have the app_service_admin role.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Describes an encountered error.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;details&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>error.details</td>
<td>Additional information about the error.</td>
</tr>
<tr>
<td>error.message</td>
<td>Message describing the error.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

```bash
curl -X PUT \
https://instance.servicenow.com/api/now/cmdb/csdm/app_service/f809cb60db941037740f95ca96190f/service_details \
-H 'Accept: application/json' \
-H 'Content-Type: application/json' \
-u 'username':'password' \
-d '{
  "basic_details": {
    "environment": "Test Lab 2"
  },
  "relationships": {
    "business_app" :["f07c7620db54101037740f95ca96199a"]
  }
}'

{
  "result": {}
}
```

**Attachment API**

The Attachment API allows you to upload and query file attachments.

You can upload or retrieve a single file with each request.

The Attachment API respects any system limitations on uploaded files, such as maximum file size and allowed attachment types. You can control these settings using the properties `com.glide.attachment.max_size`, 1024MB by default, and `glide.attachment.extensions`.

ℹ️ **Note:** The Attachment API accepts all Content-Type values (*/*). Specify the file content type when uploading an attachment. The content type is stored with file metadata allowing other tools to correctly identify and parse the file.

The following video provides more information on the Attachment API:

Web Services Part 5 | How to Make REST Attachment API Requests

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Attachment API role requirements

To create attachments, the user record used to authenticate the HTTP request with ServiceNow must have any roles required to create Attachment [sys_attachment] records. It must also have any roles required to read and write records on the target table, such as the itil role to add attachments to incident records.

By default there is no single role allowing a user to add attachments. You can create a role to explicitly allow adding attachments, then assign this role to the user account being used to make the request.

Attachment - DELETE /now/attachment/{sys_id}

This method deletes the attachment with a specific sys_id value.

URL format

Versioned URL: /api/now/v1/attachment/{sys_id}
Default URL: /api/now/attachment/{sys_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>Sys_id value of the attachment to delete. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes
The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Indicates the request ran successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Example: cURL request

curl "https://instance.servicenow.com/api/now/attachment/615ea769c0a80166001cf5f2367302f5"
  --request DELETE \
  --user 'username':'password'

Example: Python request

# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/attachment/615ea769c0a80166001cf5f2367302f5'
user = 'username'
pwd = 'password'

# Make the HTTP request
response = requests.delete(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',response.json())
    exit()

None

Attachment - GET /now/attachment

Returns the metadata for multiple attachments.

URL format

Versioned URL: api/now/v1/attachment

Default URL: api/now/attachment
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_limit</td>
<td>Limit to be applied on pagination.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Unusually large <code>sysparm_limit</code> values can impact system performance.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: 10000</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Number of records to exclude from the query. Use this parameter to get more records than specified in <code>sysparm_limit</code>. For example, if <code>sysparm_limit</code> is set to 500, but there are additional records you want to query, you can specify a <code>sysparm_offset</code> value of 500 to get the second set of records.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: 0</td>
</tr>
<tr>
<td>sysparm_query</td>
<td>Encoded query. Queries for the <code>Attachment</code> API are relative to the <code>Attachments [sys_attachment]</code> table.</td>
</tr>
<tr>
<td></td>
<td>For example: <code>(sysparm_query=file_name=attachment.doc)</code></td>
</tr>
<tr>
<td></td>
<td>The encoded query provides support for <code>ORDERBY</code>. To sort responses based on certain fields, use the <code>ORDERBY</code> and <code>ORDERBYDESC</code> clauses in <code>sysparm_query</code>. For example, <code>sysparm_query=ORDERBYfile_name^ORDERBYDESCtable_Name</code> orders the results in ascending order by name first, then in descending order by table name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Headers**
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>The content type of the response. For metadata requests, this is the content type of the metadata, not the content type of the attachment files.</td>
</tr>
<tr>
<td>Link</td>
<td>Links to download the attachments.</td>
</tr>
</tbody>
</table>

**Status codes**
The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
</tbody>
</table>
Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Returned information depends on the selected attachments.</td>
</tr>
</tbody>
</table>

Example: cURL request

```
curl "https://instance.servicenow.com/api/now/attachment?sysparm_limit=1" \
  --request GET \
  --header "Accept:application/json" \
  --user "username":"password"
```

```
{"result": [ 
  {
    "table_sys_id": "5054b6f8c0a800060056addcf551ecf8",
    "size_bytes": "462",
    "download_link": "https://instance.service-now.com/api/now/attachment/615ea769c0a80166001cf5f2367302f5/file",
    "sys_updated_on": "2009-05-21 04:12:21",
    "sys_id": "615ea769c0a80166001cf5f2367302f5",
    "image_height": "",
    "sys_created_on": "2009-05-21 04:12:21",
    "file_name": "blocks.swf",
    "sys_created_by": "glide.maint",
    "compressed": "true",
    "average_image_color": "",
    "sys_updated_by": "glide.maint",
    "sys_tags": "",
    "table_name": "content_block_programmatic",
    "image_width": "",
    "sys_mod_count": "0",
    "content_type": "application/x-shockwave-flash",
    "size_compressed": "485"
  }
]}
```
}
]
}

Example: Python request
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/attachment?sysparm_limit=1'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Accept":"application/xml"}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
response.content)
exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
<?xml version="1.0" encoding="UTF-8"?>
<response>
<result>
<table_sys_id>5054b6f8c0a800060056addcf551ecf8</table_sys_id>
<size_bytes>462</size_bytes>

<download_link>https://instance.service-now.com/api/now/attachment/615ea769c0a80166001cf5f
2367302f5/file</download_link>
<sys_updated_on>2009-05-21 04:12:21</sys_updated_on>
<sys_id>615ea769c0a80166001cf5f2367302f5</sys_id>
<image_height />
<sys_created_on>2009-05-21 04:12:21</sys_created_on>
<file_name>blocks.swf</file_name>

5251

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


Attachment - GET /now/attachment/{sys_id}/file

Returns the binary file attachment with a specific sys_id value.

URL format

Versioned URL: /api/now/v1/attachment/{sys_id}/file

Default URL: /api/now/attachment/{sys_id}/file

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>Sys_id of the attachment record from which to return binary data.</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers
<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. For example, use image/jpeg or image/png to accept JPEG or PNG image files exclusively. To allow all image types, specify image/*; to allow any file type, specify <em>/</em>. Default: <em>/</em></td>
</tr>
</tbody>
</table>

Response headers
<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-Attachment-Metadata</td>
<td>Metadata about the returned file, such as size, name, and file type.</td>
</tr>
</tbody>
</table>

Status codes
The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

Status codes
<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
# Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary file attachment</td>
<td></td>
</tr>
</tbody>
</table>

## Example: cURL request

```bash
curl "https://instance.servicenow.com/api/now/attachment/615ea769c0a80166001cf5f2367302f5/file" \
  --request GET \
  --header "Accept:*/*" \
  --user "username":"password"
```

Binary response not shown.

## Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/attachment/615ea769c0a80166001cf5f2367302f5/file'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': '*/*'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

Binary response not shown.
```
Attachment - GET /now/attachment/{sys_id}
Returns the metadata for the attachment file with a specific sys_id value.

**URL format**
Versioned URL: /api/now/v1/attachment/{sys_id}
Default URL: /api/now/attachment/{sys_id}

**Supported request parameters**

<p>| Path parameters |
|-----------------|-----------------|</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>Sys_id of the attachment record for which to retrieve the metadata.</td>
</tr>
</tbody>
</table>

<p>| Query parameters |
|-----------------|-----------------|</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Request body parameters (XML or JSON)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

**Headers**
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

<p>| Request headers |
|-----------------|-----------------|</p>
<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>The content type of the response. For metadata requests, this is the content type of the metadata, not the content type of the attachment files.</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Indicates the specified attachment does not exist, or the current user cannot access it.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metadata specific to the specified attachment.</td>
<td></td>
</tr>
</tbody>
</table>

Example: cURL request

```bash
curl "https://instance.servicenow.com/api/now/attachment/615ea769c0a80166001cf5f2367302f5" \
  --request GET \
  --header "Accept:application/json" \
  --user "username":"password"

{
    "result": {
      "table_sys_id": "5054b6f8c0a800060056addcf551ecf8",
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.service-now.com/api/now/attachment/615ea769c0a80166001cf5f2367302f5'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Accept": "application/xml"}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.content)
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Attachment - POST /now/attachment/file

Uploads a specified binary file as an attachment to a specified record.

⚠️ Note: The file to attach must be specified after the last parameter in the passed-in request parameter list.

**URL format**

Versioned URL: /api/now/v1/attachment/file
Default URL: /api/now/attachment/file
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>encryption_context</td>
<td>Sys_id of an encryption context record. Specify this parameter to allow only users with the specified encryption context to access the attachment. For additional information on encryption context records, see Encryption Support.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: Attached file is not encrypted with any encryption context.</td>
</tr>
<tr>
<td>file_name</td>
<td>Required. Name to give the attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>table_name</td>
<td>Required. Name of the table to attach the file to.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>table_sys_id</td>
<td>Required. Sys_id of the record in the table specified in <code>table_name</code> that you want to attach the file to.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;</td>
<td>Path to the binary file to attach to the specified record.</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Content type of the file to attach, such as image/jpeg or /*. This header is mandatory to post file attachments.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>URL of the new attachment.</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Indicates the query ran successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Indicates that one or more mandatory parameters were missing from the request.</td>
</tr>
<tr>
<td>404</td>
<td>Indicates the record specified by the table_name and table_sys_id parameters does not exist or is not accessible to the current user.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Metadata of the requested attachment. Data type: Object</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>result</td>
<td></td>
</tr>
</tbody>
</table>
  "average_image_color": "String",
  "compressed": "String",
  "content_type": "String",
  "created_by_name": "String",
  "download_link": "String",
  "file_name": "String",
  "image_height": "String",
  "image_width": "String",
  "size_bytes": "String",
  "size_compressed": "String",
  "sys_created_by": "String",
  "sys_created_on": "String",
  "sys_id": "String",
  "sys_mod_count": "String",
  "sys_tags": "String",
  "sys_updated_by": "String",
  "sys_updated_on": "String",
  "table_name": "String",
  "table_sys_id": "String",
  "updated_by_name": "String"
 |

result.average_image_color | If the attachment is an image, the sum of all colors.  
Data type: String  
Unit: RGB or number of pixels.  
result.compressed | Flag that indicates whether the attachment file has been compressed.  
Valid values:  
• true: File has been compressed.  
• false: File has not been compressed.  
Data type: String  
result.content_type | Content-type of the associated attachment file, such as image or jpeg or application/x-shockwave-flash.  
Data type: String
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| result.created_by_name      | Full name of entity that originally created the attachment file.  
Data type: String                                                                 |
| result.download_link        | Download URL of the attachment on the ServiceNow instance.  
Data type: String                                                                 |
| result.file_name            | File name of the attachment.  
Data type: String                                                                 |
| result.image_height         | If an image file, the height of the image.  
Data type: String  
Unit: Pixels                                                              |
| result.image_width          | If an image file, the width of the image.  
Data type: String  
Unit: Pixels                                                              |
| result.size_bytes           | Size of the attachment.  
Data type: String  
Unit: Bytes                                                                |
| result.size_compressed      | Size of the compressed attachment file. If the file is not compressed, empty.  
Data type: String  
Unit: Bytes                                                                |
| result.sys_created_by       | Entity that originally created the attachment file.  
Data type: String                                                              |
| result.sys_created_on       | Date and time that the attachment file was initially saved to the instance.  
Data type: String                                                              |
| result.sys_id               | Sys_id of the attachment file.  
Data type: String                                                              |
<p>| result.sys_mod_count        | Number of times the attachment file has been modified (uploaded to the instance).                                                          |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.sys_tags</td>
<td>Any system tags associated with the attachment file.</td>
</tr>
<tr>
<td>result.sys_updated_by</td>
<td>Entity that last updated the attachment file.</td>
</tr>
<tr>
<td>result.sys_updated_on</td>
<td>Date and time that the attachment file was last updated.</td>
</tr>
<tr>
<td>result.table_name</td>
<td>Name of the table to which the attachment is associated.</td>
</tr>
<tr>
<td>result.table_sys_id</td>
<td>Sys_id of the table associated with the attachment.</td>
</tr>
<tr>
<td>result.updated_by_name</td>
<td>Full name of entity that last updated the attachment file.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl 
  "https://instance.servicenow.com/api/now/attachment/file?table_name=incident&table_sys_id=
d71f7935c0a8016700802b64c67c11c6&file_name=Issue_screenshot" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type: image/jpeg" \
  --user "username":"password" \
  --data-binary "@ location of the file on file system"
```

```json
{
  "result": {
    "table_sys_id": "d71f7935c0a8016700802b64c67c11c6",
    "size_bytes": "36597",
    "download_link": 
  "https://instance.servicenow.com/api/now/attachment/6ea10fe64f411200adf9f8e18110c739/file"
  }
}
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url =
'https://instance.servicenow.com/api/now/attachment/file?table_name=incident&table_sys_id=d71f7935c0a8016700802b64c67c11c66&file_name=Issue_screenshot.jpg'

# Specify the file To send. When specifying files to send make sure you specify the path to the file, in
# this example the file was located in the same directory as the python script being executed.
data = open('Issue_screenshot.jpg', 'rb').read()

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = ("Content-Type":"image/jpeg","Accept":"application/json")

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, data=data)
```
# Check for HTTP codes other than 201
if response.status_code != 201:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "table_sys_id": "d71f7935c0a8016700802b64c67c11c6",
        "size_bytes": "36597",
        "download_link": "https://instance.service-now.com/api/now/attachment/6ea10fe64f411200adf9f8e18110c739/file",
        "sys_updated_on": "2016-01-22 15:14:07",
        "sys_id": "6ea10fe64f411200adf9f8e18110c739",
        "image_height": "",
        "sys_created_on": "2016-01-22 15:14:07",
        "file_name": "Issue_screenshot.jpg",
        "sys_created_by": "admin",
        "compressed": "true",
        "average_image_color": "",
        "sys_updated_by": "admin",
        "sys_tags": "",
        "table_name": "incident",
        "image_width": "",
        "sys_mod_count": "0",
        "content_type": "image/jpeg",
        "size_compressed": "25130"
    }
}

Attachment - POST /now/attachment/upload

Uploads a multipart file attachment.

The multipart POST method does not accept any parameters. You must specify the table name and record sys_id values within the form body. See the cURL example for a sample of a multipart/form-data request.
Important: When using multipart POST, ensure that the file content is contained in the final part of the message only. Earlier parts should contain only metadata such as table name and record sys_id.

**URL format**

Versioned URL: /api/now/v1/attachment/upload

Default URL: /api/now/attachment/upload

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

| **Name** | **Description** |
| None | |

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Content type of the request. Set this value to multipart/form-data when using the multipart POST method.</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>URL of the new attachment.</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Indicates the query ran successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Metadata of the requested attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
</tbody>
</table>

```
"result": {
    "average_image_color": "String",
    "compressed": "String",
    "content_type": "String",
    "created_by_name": "String",
    "download_link": "String",
    "file_name": "String",
    "image_height": "String",
    "image_width": "String",
    "size_bytes": "String",
    "size_compressed": "String"
}
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sys_created_by</code></td>
<td>&quot;String&quot;,</td>
</tr>
<tr>
<td><code>sys_created_on</code></td>
<td>&quot;String&quot;,</td>
</tr>
<tr>
<td><code>sys_id</code></td>
<td>&quot;String&quot;,</td>
</tr>
<tr>
<td><code>sys_mod_count</code></td>
<td>&quot;String&quot;,</td>
</tr>
<tr>
<td><code>sys_tags</code></td>
<td>&quot;String&quot;,</td>
</tr>
<tr>
<td><code>sys_updated_by</code></td>
<td>&quot;String&quot;,</td>
</tr>
<tr>
<td><code>sys_updated_on</code></td>
<td>&quot;String&quot;,</td>
</tr>
<tr>
<td><code>table_name</code></td>
<td>&quot;String&quot;,</td>
</tr>
<tr>
<td><code>table_sys_id</code></td>
<td>&quot;String&quot;,</td>
</tr>
<tr>
<td><code>updated_by_name</code></td>
<td>&quot;String&quot;</td>
</tr>
<tr>
<td><code>result.average_image_color</code></td>
<td>If the attachment is an image, the sum of all colors.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Unit: RGB or number of pixels.</td>
</tr>
<tr>
<td><code>result.compressed</code></td>
<td>Flag that indicates whether the attachment file has been compressed.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: File has been compressed.</td>
</tr>
<tr>
<td></td>
<td>• false: File has not been compressed.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>result.content_type</code></td>
<td>Content-type of the associated attachment file, such as image or jpeg or</td>
</tr>
<tr>
<td></td>
<td>application/x-shockwave-flash.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>result.created_by_name</code></td>
<td>Full name of entity that originally created the attachment file.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>result.download_link</code></td>
<td>Download URL of the attachment on the ServiceNow instance.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>result.file_name</code></td>
<td>File name of the attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>result.image_height</code></td>
<td>If an image file, the height of the image.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.image_width</td>
<td>If an image file, the width of the image.</td>
</tr>
<tr>
<td>result.size_bytes</td>
<td>Size of the attachment.</td>
</tr>
<tr>
<td>result.size_compressed</td>
<td>Size of the compressed attachment file. If the file is not compressed, empty.</td>
</tr>
<tr>
<td>result.sys_created_by</td>
<td>Entity that originally created the attachment file.</td>
</tr>
<tr>
<td>result.sys_created_on</td>
<td>Date and time that the attachment file was initially saved to the instance.</td>
</tr>
<tr>
<td>result.sys_id</td>
<td>Sys_id of the attachment file.</td>
</tr>
<tr>
<td>result.sys_mod_count</td>
<td>Number of times the attachment file has been modified (uploaded to the instance).</td>
</tr>
<tr>
<td>result.sys_tags</td>
<td>Any system tags associated with the attachment file.</td>
</tr>
<tr>
<td>result.sys_updated_by</td>
<td>Entity that last updated the attachment file.</td>
</tr>
<tr>
<td>result.sys_updated_on</td>
<td>Date and time that the attachment file was last updated.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.table_name</td>
<td>Name of the table to which the attachment is associated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.table_sys_id</td>
<td>Sys_id of the table associated with the attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.updated_by_name</td>
<td>Full name of entity that last updated the attachment file.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**POST multipart mandatory values**

When sending a multipart POST request to upload a file attachment, include attachment data in the message body, not in the URL parameters. You must specify these values in the message body:

**Mandatory values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Content-Type of the file, included in the message body for multipart uploads.</td>
</tr>
<tr>
<td></td>
<td><em>Note:</em> The Content-Type must be defined within the file portion of the POST message, not within the form data. See the sample POST multipart message for an example of a multipart message.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>table_name</td>
<td>Name of the table to which you want to attach the file.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>table_sys_id</td>
<td>Sys_id of the record on the specified table to which you want to attach the file.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl "https://instance.servicenow.com/api/now/attachment/upload" \
--request POST \ 
--header "Accept:application/json"\n```
Example: Python request

```python
# This example uses the Python Requests Library and you will need to install requests package for python
# Documentation can be found at http://docs.python-requests.org/en/master/user/quickstart/
import requests
import pprint
import json

# Specify the Endpoint URL replacing instance with your ServiceNow Instance Name
url = 'https://instance.service-now.com/api/now/attachment/upload'
```
# Specify Parameters for File Being Uploaded, the table_name and table_sys_id should be replaced with values that make sense for your use case
payload = {'table_name': 'incident', 'table_sys_id': '81f8915b0b6ba20028927416bf961971'}

# Specify Files To Send and Content Type. When specifying files to send make sure you specify the path to the file, in this example the file was located in the same directory as the python script being executed.
# it is important to specify the correct file type
files = {'file': ('issue_screenshot.JPG', open('issue_screenshot.JPG', 'rb'), 'image/jpg', {'Expires': '0'})}

# Eg. User name="username", Password="password" for this code sample. This will be sent across as basic authentication
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': '*/*'}

# Send the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, files=files, data=payload)

# Check for HTTP codes other than 201
if response.status_code != 201:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Print Response Details
print 'Response Status Code:', response.status_code

print ''
print('Response Payload:
print json.dumps(response.json(), indent=4)

Response Status Code: 201

Response Payload:
{
  "result": {
    "sys_tags": "",
    "sys_updated_by": "admin",
  }
AWA Agent API

Use the AWA (Advanced Work Assignment) Agent API to get or set agent presence and channel availability.

The AWA Agent API is provided within the sn_awa namespace.

It requires the Advanced Work Assignment (com.glide.awa) plugin and the awa_integration_user role.

**AWA Agent - GET /now/awa/agents/{user_id}**

Returns the current agent presence state and channel availability.

**URL format**

Versioned URL: /api/now/{api_version}/awa/agents/{user_id}

Default URL: /api/now/awa/agents/{user_id}
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, <code>v1</code> or <code>v2</code>. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>user_id</td>
<td>Sys_id of the agent. Located in the User <code>[sys_user]</code> table. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>presence.channels.flag</td>
<td>Indicates whether the channel is available. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Channel is available.</td>
</tr>
<tr>
<td></td>
<td>• false: Channel is not available.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>presence.channels.sys_id</td>
<td>Channel sys_id. Located in the Service Channels <code>[awa_service_channel_list]</code> table. Data type: String</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <strong>application/json</strong> or <strong>application/xml</strong>. Default: <strong>application/json</strong></td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: <strong>application/json</strong> or <strong>application/xml</strong>. Default: <strong>application/json</strong></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>presence</td>
<td>Information about an agent’s current presence state and channel. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>{&quot;presence&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;available&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;channels&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>presence.available</td>
<td>Flag that indicates whether the agent is available. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Agent is available.</td>
</tr>
<tr>
<td></td>
<td>• false: Agent is not available.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>presence.channels</td>
<td>List of objects that describe the available channels of communication with the agent. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>{&quot;channels&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;available&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>presence.channels.available</td>
<td>Flag that indicates whether the channel is available. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Channel is available.</td>
</tr>
<tr>
<td></td>
<td>• false: Channel is not available.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>presence.channels.name</td>
<td>Channel name, such as Chat or Phone.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Response body parameters (JSON or XML) (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>presence.channels.sys_id</td>
<td>Channel sys_id. Located in the Service Channels [awa_service_channel_list] table. Data type: String</td>
</tr>
<tr>
<td>presence.name</td>
<td>Name of the agent’s presence state. Possible values: • Available • Unavailable Data type: String</td>
</tr>
<tr>
<td>presence.restrict_update</td>
<td>Flag that indicates whether the user can restrict updates. Possible values: • true: User can restrict updates. • false: User cannot restrict updates. Data type: Boolean</td>
</tr>
<tr>
<td>sys_id</td>
<td>Agent sys_id. Data type: String</td>
</tr>
</tbody>
</table>

### Example: Sample cURL request

```bash
curl -X GET \
https://instance.servicenow.com/api/now/awa/agents/46d44a23a9fe19810012d100c8a80666 \
-H 'Accept: application/json' \
-H 'Content-Type: application/json' \
-u 'username': 'password'\
{
    "result": {
        "presence": {
            "name": "Available",
            "sys_id": "0b10223c57a313005b4a65ef94f970",
```
Example: Sample Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/awa/agents/46d44a23a9fe19810012d100cca80666'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Content-Type": "application/json", "Accept": "application/json"}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers, data="{"presence": 
{"sys_id": "0b10223c57a313005baaaa65ef94f970", 
"channels": [{"sys_id": "0bbdedbb3b892300a2bac9bb34efc445", "available": 
true }]
}

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',response.content)
    exit()

# Decode the JSON response into a dictionary and use the data
print(response.content)
```
AWA Agent - PUT /now/awa/agents/{user_id}
Sets the state of a specified agent’s presence and, if provided, sets the agent’s channel availability for that state.

**URL format**

Versioned URL: /api/now/{api_version}/awa/agents/{user_id}
Default URL: /api/now/awa/agents/{user_id}

**Supported request parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>user_id</td>
<td>Sys_id of the agent. Located in the User [sys_user] table.</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>presence.channels</td>
<td>List of objects the define which channels are available for the agent. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;channels&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;available&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>presence.channels.available</td>
<td>Flag that indicates whether the channel is available. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Channel is available.</td>
</tr>
<tr>
<td></td>
<td>• false: Channel is not available.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>presence.channels.sys_id</td>
<td>Channel sys_id. Located in the Service Channels [awa_service_channel_list] table. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>presence</td>
<td>Information about an agent's current presence state and channel.</td>
</tr>
</tbody>
</table>
### Response body parameters (JSON or XML) (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data type:</strong></td>
<td>Object</td>
</tr>
</tbody>
</table>

```json
"presence": {
    "available": Boolean,
    "channels": [Array],
    "name": "String",
    "sys_id": "String"
}
```

**presence.available**  
Flag that indicates whether the agent is available.  
Possible values:  
- true: Agent is available.  
- false: Agent is not available.  

**Data type:** Boolean

**presence.channels**  
List of objects that describe the available channels of communication with the agent.  

**Data type:** Array

```json
"channels": [
    {
        "available": Boolean,
        "name": "String",
        "sys_id": "String"
    }
]
```

**presence.channels.available**  
Flag that indicates whether the channel is available.  
Possible values:  
- true: Channel is available.  
- false: Channel is not available.  

**Data type:** Boolean

**presence.channels.name**  
Channel name, such as Chat or Phone.  

**Data type:** String

**presence.channels.sys_id**  
Channel sys_id. Located in the Service Channels [awa_service_channel_list] table.
### Response body parameters (JSON or XML) (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>presence.name</td>
<td>Name of the agent's presence state. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Available</td>
</tr>
<tr>
<td></td>
<td>• Unavailable</td>
</tr>
<tr>
<td>presence.restrict_update</td>
<td>Flag that indicates whether the user can restrict updates. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: User can restrict updates.</td>
</tr>
<tr>
<td></td>
<td>• false: User cannot restrict updates.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Agent sys_id.</td>
</tr>
</tbody>
</table>

```
curl -X PUT \
  https://instance.servicenow.com/api/now/awa/agents/46d44a23a9fe19810012d100cca80666 \
  -H 'Accept: application/json' \
  -H 'Content-Type: application/json' \
  -u 'username':'password'\n  -d '{"presence": {\n    "sys_id": "0b10223c57a313005baaa65ef94f970",\n    "channels": [{\n      "sys_id": "0bbdedbb3b892300a2bac9bb34efc445",\n      "available": true
    }]
  }}
```
Example: Sample Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/awa/agents/46d44a23a9fe19810012d100cca80666'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Content-Type":"application/json","Accept":"application/json"}

# Make the HTTP request
response = requests.put(url, auth=(user, pwd), headers=headers, data='{"presence":{
  \"sys_id\": "0b10223c57a313005baaaa65ef94f970",
  \"channels\": [{
    \"sys_id\": "0b10223c57a313005baaaa65ef94f970",
    \"available\": true
  }],
  \"available\": true
}}')

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the JSON response into a dictionary and use the data
print(response.content)
```

AWA Assignment API

Use the AWA Assignment API to manually assign work items to agents.

This API requires the Advanced Work Assignment (com.glide.awa) plugin and awa_manager role.

A work item is a single piece of work handled by an AWA agent from start to finish. For example, one chat or one case is an object that can be routed and assigned to agents. For more information, refer to Advanced Work Assignment.

**AWA Manual Assign – POST /awa/workitems/{work_item_sys_id}/assignments**

Assigns an available work item to an available Advanced Work Assignment agent.

The primary use case for this endpoint is to enable external routing systems to route work items. If Advanced Work Assignment is configured to use external routing, work items in the queue are assigned using external routing and not AWA. The work item task can be assigned by calling this endpoint. For information, refer to Use external routing.

**URL format**

Versioned URL: /now/{api_version}/awa/workitems/{sys_id}/assignments

Default URL: /now/awa/workitems/{sys_id}/assignments
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>work_item_sys_id</td>
<td>Sys_id of the work item to assign to an available agent. Located in the Work Items [awa_work_item] table. The work item must be unassigned and in the Pending Accept or Queued state. For information, refer to Check unassigned task work items. Type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent_sys_id</td>
<td>Required. Sys_id of the available agent to receive the work item. Agents are users with the awa_agent role in the User [sys_user] table. For information on how to determine if an agent is available, refer to Agent Inbox controls. Type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>409</td>
<td>Conflict. The request could not pass due to an error with the provided work item or agent sys_id.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
# Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>success</td>
<td>Flag that indicates whether the manual work item assignment is successful.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Work item assignment successful.</td>
</tr>
<tr>
<td></td>
<td>• false: Work item assignment unsuccessful.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: True</td>
</tr>
<tr>
<td>message</td>
<td>Response message acknowledging successful assignment or exception. Exceptions:</td>
</tr>
<tr>
<td></td>
<td>• &quot;Work item can not be assigned&quot; – Work item provided cannot be assigned because it is in <strong>Accepted</strong> or <strong>Canceled</strong> state. Refer to Check work items and AWA events.</td>
</tr>
<tr>
<td></td>
<td>• &quot;&lt;agent_sys_id&gt; is not a valid agent&quot; – Agent does not have the awa_agent role.</td>
</tr>
<tr>
<td></td>
<td>• &quot;&lt;work_item_sys_id&gt; is not a valid work item&quot; – Provided work item sys_id does not exist.</td>
</tr>
<tr>
<td></td>
<td>• &quot;Work Item is already assigned to &lt;work_item_sys_id&gt;&quot; – Provided work item is assigned to another agent.</td>
</tr>
<tr>
<td></td>
<td>• &quot;Agent is not available&quot; – Agent is not in the <strong>Available</strong> state in AWA. Refer to Agent inbox controls.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

## Example: cURL request

The following example shows how to assign a work item to an available AWA agent.

```bash
curl "https://instance.servicenow.com/api/now/awa/workitems/<work_item_sys_id>/assignments" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json"
```
The result shows that the task has been successfully assigned to the agent. You can verify results in the Assigned to field of the Work Items [awa_work_item] table.

```json
{
  "result": {
    "success": true,
    "message": "Manual assignment successfully requested"
  }
}
```

**AWA Routing API**

Use the AWA (Advanced Work Assignment) API to route a document to a queue.

Requires the Advanced Work Assignment plugin (com.glide.awa) and awa_integration_user role.

**AWA Routing - POST /now/awa/queues/{queue_sys_id}/work_item**

If an active work item exists, routes a document to a queue. If an active work item does not exist for the specified document, creates a work item and set its fields accordingly.

**URL format**

Versioned URL: /api/now/{api_version}/awa/queues/{queue_sys_id}/work_item

Default URL: /api/now/awa/queues/{queue_sys_id}/work_item

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>queue_sys_id</td>
<td>Sys_id of the queue record from the Queue [awa_queue] table. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>document_sys_id</td>
<td>Sys_id of the document to route to the queue.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>document_table</td>
<td>Name of the table associated with the document, such as incident.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>display_name</td>
<td>Name of the document to route by this work item, such as case record. Data type: String</td>
</tr>
<tr>
<td>document_sys_id</td>
<td>Sys_id of the document to route to the queue. Data type: String</td>
</tr>
<tr>
<td>document_table</td>
<td>Name of the table associated with the document. Data type: String</td>
</tr>
<tr>
<td>queue_sys_id</td>
<td>Sys_id of the queue on which to route a document. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the work item assigned to the queue. Data type: String</td>
</tr>
</tbody>
</table>

Example: Sample cURL request

```
curl -X POST  
https://instance.servicenow.com/api/now/awa/queues/339eed3f3b892300a2bac9bb34efc4c3/work_item  
-H 'Content-Type: application/json'  
-H 'Accept: application/json'  
-u 'username':'password'  
-d '{
```
Example: Sample Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/awa/queues/339eed3f3b892300a2bac9bb34efc4c3/work_item'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Content-Type":"application/json","Accept":"application/json"}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers,
                         data="{"document_sys_id":"9c573169c611228700193229fff72400",
                         "document_table": "incident"}
                         ")

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.content)
    exit()

# Decode the JSON response into a dictionary and use the data
print(response.content)
```
Batch API

The Batch API lets you send a single request containing multiple REST API calls, and returns a stream of response payloads.

This REST API enables integrators to:

• Decrease the amount of time required to send API requests by batching them together, and reduce overhead by narrowing authentication, session setup, and round-trip traffic to a single step.
• Create more efficient code for client-side integrations.
• Mix batch item formats and include them in single batch request. For example, one batch can include a Bases64-encoded request in XML format to send to one endpoint and a Base64 encoded request in JSON format to send to a different endpoint.
• Receive a stream of response payloads in return.
• Apply existing ACLs to each API call in the batch.

You can include any API available on the instance in a Batch API call. For performance reasons, avoid including long-running requests and requests that retrieve large amounts of data.

Size and processing limits

Payloads adhere to these size limits:

• Each item in the request: 5 MB. You can change this default by updating the glide.rest.batch.max.inputSize system property. Maximum value: 10 MB.
• Each item in the response: 10 MB. You can change this default by updating the glide.rest.batch.max.outputSize system property or adding the X-BATCHREQUEST-MAX-OUTPUT-SIZE header to your request. The X-BATCHREQUEST-MAX-OUTPUT-SIZE header value cannot exceed the value of the system property.
**Note:** If a request runs longer than 30 seconds, the REST Batch API request timeout transaction quota rule ends the transaction. Increasing the value of this transaction quota rule may result in performance issues.

When a batch request reaches a size or processing limit, the system cancels the transaction and returns unprocessed requests in the **unserviced** JSON array in the response.

**Batch - POST /now/batch**
Sends multiple REST API requests in a single call.

You can include any API available on the instance in a Batch API call. For performance reasons, avoid including long-running requests and requests that retrieve large amounts of data.

**URL format**
Versioned URL: /api/now/{api_version}/batch

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>batch_request_id</td>
<td>ID that identifies the batch of requests. Data type: String</td>
</tr>
</tbody>
</table>
### Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rest_requests</td>
<td>Required. List of request objects to include in the batch request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;rest_requests&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;body&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;exclude_response_headers&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;headers&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;method&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>rest_requests.body</td>
<td>Base64-encoded body of the request. Only applies to methods that require a body, for example POST. Before encoding, the body can be in any format. For example, XML or JSON.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>rest_requests.exclude_response_headers</td>
<td>Flag that indicates whether to exclude response headers from the response.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Response headers are not included in the response.</td>
</tr>
<tr>
<td></td>
<td>• false: Response headers are included in the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>rest_requests.headers</td>
<td>List of request header objects to send to the endpoint.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>
### Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;headers&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>rest_requests.headers.name</td>
<td>Name of the header.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>rest_requests.headers.value</td>
<td>Value of the header.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>rest_requests.id</td>
<td>Required. ID for each item in the batch.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>rest_requests.method</td>
<td>Required. Method to call for the associated endpoint.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>rest_requests.url</td>
<td>Required. Relative path of the endpoint to send the request to. Includes the query parameters.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>
Request headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-BATCHREQUEST-MAX-OUTPUT-SIZE</td>
<td>Size limit for each item in the batch response. Add this header to provide a lower size limit than the default set by the glide.rest.batch.max.outputSize system property. You cannot set a value higher than the value in the system property. Default: 10 MB.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>batch_request_id</td>
<td>ID of the batch that matches the batch_request_id parameter in the request. Data type: String</td>
</tr>
<tr>
<td>serviced_requests</td>
<td>List of response objects from the batch request.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>serviced_requests.body</td>
<td>Base64 encoded body of the response. To get the value of the body, Base64 decode the content of this parameter. Data type: String</td>
</tr>
<tr>
<td>serviced_requests.error_message</td>
<td>If present, the error messages. Data type: String</td>
</tr>
<tr>
<td>serviced_requests.execution_time</td>
<td>Time it took to execute the batch item request. Data type: Number Unit: Milliseconds</td>
</tr>
<tr>
<td>serviced_requests.headers</td>
<td>Headers for the batch item. Data type: Array</td>
</tr>
<tr>
<td>serviced_requests.headers.name</td>
<td>Name of the header. Data type: String</td>
</tr>
<tr>
<td>serviced_requests.headers.value</td>
<td>Value of the header.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>serviced_requests.id</td>
<td>ID of the batch item that matches the rest_requests.id parameter in the request.</td>
</tr>
<tr>
<td>serviced_requests.redirect_url</td>
<td>If present, the redirect URL.</td>
</tr>
<tr>
<td>serviced_requests.status_code</td>
<td>Status code for the batch item.</td>
</tr>
<tr>
<td>serviced_requests.status_text</td>
<td>Text of the status code for the batch item.</td>
</tr>
<tr>
<td>unserviced_requests</td>
<td>IDs of the requests that were not processed because the batch request reached a size or processing limit.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

This batch request sends two GET requests to the instance and one POST request to the incident table. The body of the POST request is Base64 encoded.

```bash
curl "https://instance.servicenow.com/api/now/v1/batch" \
--request POST \
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--user "username":"password"\
--data "{
  "batch_request_id":"1",
  "rest_requests":[
    {
      "id":"11",
      "headers": [
        {
          "name":"Content-Type",
          "value":"application/xml"
        },
        {
          "name":"Accept",
          "value":"application/xml"
        }
      ]
    }
  ]
}"
```

"url":"/api/global/user_role_inheritance",
"method":"GET"
},
{
"id":"12",
"exclude_response_headers":true,
"headers":[
  {
   "name":"Content-Type",
   "value":"application/json"
  },
  {
   "name":"Accept",
   "value":"application/json"
  }
],
"url":"/api/now/table/incident?sysparm_limit=1",
"method":"GET"
},
{
"id":"13",
"exclude_response_headers":true,
"headers":[
  {
   "name":"Content-Type",
   "value":"application/json"
  },
  {
   "name":"Accept",
   "value":"application/json"
  }
],
"url": "/api/now/table/incident",
"method":"POST",
"body": "eyd1cmdlbmN5JzonMScsICdzaG9ydF9kZXNjcm1lwdGlvbic6J015IGNvbXB1dGVyIGhyb2tlJywqJ2NvbW1lbmRzJzonRWxldmF0aW5nIHYvZ2VuY3ksIHRoaXMgaXMgYSBibG9ja2luZyBpc3N1ZSd9"
}*
}
Case API

The Case REST API enables you to retrieve and update Customer Service Management (CSM) case records.

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
In addition, you can generate new social media profile records when creating a case.

The Case API requires the Customer Service plugin (com.sn_customerservice) and is provided within the sn_customerservice namespace.

Users require the csm_ws_integration role for full API access. External users with the sn_customerservice.customer or sn_customerservice.consumer roles can access the case API endpoints.

**Case - GET /sn_customerservice/case/{id}/activities**

Retrieves the activity stream for a specified Customer Service Management (CSM) case.

Use this endpoint to retrieve all entries from the activity stream for a specified record from the Case [sn_customerservice_case] table.

To use this endpoint, users must have the csm_ws_integration, sn_customerservice.customer, or sn_customerservice.consumer role.

**Note:** You can reference all sysparm query parameters using either their full name or their shortened name (without the `sysparm_` prefix). For example, for `sysparm_limit` you can also use `limit`.

**URL format**

Versioned URL: `/api/sn_customerservice/{api_version}/case/{id}/activities`

Default URL: `/api/sn_customerservice/case/{id}/activities`

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td></td>
<td>id</td>
<td>Sys_id or case number of the case to retrieve. Located in the Case [sn_customerservice_case] table. Results will be unpredictable if a specified case number is shared by two or more cases. Data type: String</td>
</tr>
</tbody>
</table>
### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| `sysparm_activity_type` | Filters the type of activities to return from the activity stream. The `entries.element` parameter contains the activity type for each activity. For example:  
  • `sysparm_activity_type=attachment`  
  • `sysparm_activity_type=work_notes`  
  • `sysparm_activity_type=attachment,work_notes`  
  • `sysparm_activity_type=attachment,work_notes,comments`  
  Data type: String  
  Default: Returns all types of activity. |
| `sysparm_limit`         | Maximum number of records to return. Unusually large `sysparm_limit` values can impact system performance. For requests that exceed this number of records, use the `sysparm_offset` parameter to paginate record retrieval.  
  Data type: Number |
| `sysparm_offset`        | Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks.  
  For example, the first time you call this endpoint, `sysparm_offset` is set to "0". To simply page through all available records, use `sysparm_offset=sysparm_offset +sysparm_limit`, until you reach the end of all records.  
  Data type: Number  
  Default: 0 |
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link</td>
<td>REST response data can be split into multiple result sets. Where applicable, the response header contains different links for the first set, previous set, next set, and the last set of records. For example:</td>
</tr>
<tr>
<td></td>
<td>https://&lt;instance name&gt;.service-now.com/api/now/table/cmdb_ci?sysparm_offset=40&amp;sysparm_limit=10000&gt;;rel=&quot;next&quot;,</td>
</tr>
<tr>
<td></td>
<td>https://&lt;instance name&gt;.service-now.com/api/now/table/cmdb_ci?sysparm_offset=40&amp;sysparm_limit=10000&gt;;rel=&quot;prev&quot;,</td>
</tr>
<tr>
<td></td>
<td>https://&lt;instance name&gt;.service-now.com/api/now/table/cmdb_ci?sysparm_offset=0&amp;sysparm_limit=10000&gt;;rel=&quot;first&quot;,</td>
</tr>
<tr>
<td></td>
<td>https://&lt;instance name&gt;.service-now.com/api/now/table/cmdb_ci?sysparm_offset=2780&amp;sysparm_limit=10000&gt;;rel=&quot;last&quot;</td>
</tr>
<tr>
<td>X-Total-Count</td>
<td>Response header showing the total number of records matching the request when the sysparm_limit of sysparm_offset query parameters are specified.</td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>display_value</td>
<td>Case number for the CSM case. Data type: String</td>
</tr>
<tr>
<td>entries</td>
<td>Array of objects in which each object represents an activity stream entry from the CSM case. Data type: Array</td>
</tr>
</tbody>
</table>

```
"entries": [ 
  { 
    "attachment": {Object},
    "contains_code": Boolean,
    "element": "String",
    "field_label": "String",
    "initials": "String",
    "login_name": "String",
    "name": "String",
    "sys_created_on": "String",
    "sys_created_on_adjusted": "String",
    "sys_id": "String",
    "user_sys_id": "String",
    "value": "String"
  }
]
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entries.attachment</td>
<td>Description of a file attached to the CSM case in the activity stream entry.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;attachment&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;average_image_color&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;content_type&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;file_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;image_height&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;image_path&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;image_width&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;path&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;size&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;size_bytes&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;state&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;thumbnail_path&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>entries.attachment.average_image_color</td>
<td>Hexadecimal representation of average color for image attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>entries.attachment.content_type</td>
<td>MIME type for the attachment. Examples include:</td>
</tr>
<tr>
<td></td>
<td>• application/pdf</td>
</tr>
<tr>
<td></td>
<td>• image/jpeg</td>
</tr>
<tr>
<td></td>
<td>• message/rfc822</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>entries.attachment.file_name</td>
<td>File name for the attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>entries.attachment.image_height</td>
<td>Height of image attachment in pixels.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entries.attachment.image_path</td>
<td>Direct download link for image attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>entries.attachment.image_width</td>
<td>Width of image attachment in pixels.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>entries.attachment.path</td>
<td>Direct link to the attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>entries.attachment.size</td>
<td>Size of the attachment.</td>
</tr>
<tr>
<td></td>
<td>Examples:</td>
</tr>
<tr>
<td></td>
<td>• 157.7 KB</td>
</tr>
<tr>
<td></td>
<td>• 2.2 MB</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>entries.attachment.size_bytes</td>
<td>Size of the attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Unit: Bytes</td>
</tr>
<tr>
<td>entries.attachment.state</td>
<td>State of the attachment record in the Attachment [sys_attachment] table.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Available</td>
</tr>
<tr>
<td></td>
<td>• Available Conditionally</td>
</tr>
<tr>
<td></td>
<td>• Not available</td>
</tr>
<tr>
<td></td>
<td>• Pending</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>entries.attachment.sys_id</td>
<td>Sys_id for the attachment.</td>
</tr>
<tr>
<td></td>
<td>Located in the Attachment [sys_attachment] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entries.attachment.thumbnail_path</td>
<td>Direct download link for thumbnail of image attachment. Data type: String</td>
</tr>
</tbody>
</table>
| entries.contains_code | Flag that indicates whether the activity stream entry contains code or not. Possible values:  
• true: The activity stream entry contains code.  
• false: The activity stream entry does not contain code. Data type: Boolean |
<p>| entries.element | Name for the journal field associated with the activity stream entry. Located in Dictionary Entries [sys_dictionary] table. Data type: String |
| entries.field_label | Display name for the journal field associated with the activity stream entry. Data type: String |
| entries.initials | Initials of the user who created the activity stream entry. Data type: String |
| entries.login_name | Full user name for the user who created the activity stream entry. Data type: String |
| entries.name | Full user name for the user who created the activity stream entry. Data type: String |
| entries.sys_created_on | Date and time of creation for the activity stream entry, expressed in GMT timezone. |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entries.sys_created_on_adjusted</td>
<td>Date and time of creation for the activity stream entry, expressed in logged-in user timezone.</td>
</tr>
<tr>
<td>entries.sys_id</td>
<td>Sys_id for the activity stream entry.</td>
</tr>
<tr>
<td>entries.user_sys_id</td>
<td>Sys_id for user who created the activity stream entry. Located in User [sys_user] table.</td>
</tr>
<tr>
<td>entries.value</td>
<td>Value for this journal entry.</td>
</tr>
<tr>
<td>journal_fields</td>
<td>Array of objects in which each object represents a journal field present in the Case table from which activity entries are drawn.</td>
</tr>
<tr>
<td>journal_fields.can_read</td>
<td>Flag that indicates whether the logged-in user can read this journal field or not. Possible values:</td>
</tr>
</tbody>
</table>

Data type: String
Format: YYYY-MM-DD hh:mm:ss
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• true: The logged-in user can read this journal field.</td>
</tr>
<tr>
<td></td>
<td>• false: The logged-in user cannot read this journal field.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>journal_fields.can_write</td>
<td>Flag that indicates whether the logged-in user can write to the journal field or not. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: The logged-in user can write to this journal field.</td>
</tr>
<tr>
<td></td>
<td>• false: The logged-in user cannot write to this journal field.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>journal_fields.color</td>
<td>Color that represents the journal field in case activity stream on Now Platform.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>journal_fields.label</td>
<td>Display name for the journal field.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>journal_fields.name</td>
<td>Name for the journal field. Located in Dictionary Entries [sys_dictionary] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>label</td>
<td>Display label for the table in which the CSM case record appears. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Case</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Case number for the CSM case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>short_description</td>
<td>Short description for the CSM case.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sys_created_on</td>
<td>Date and time of case creation, expressed in GMT timezone. Data type: String Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>sys_created_on_adjusted</td>
<td>Date and time of case creation, expressed in logged-in user timezone. Data type: String Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id for the CSM case. Located in the Case [sn_customerservice_case] table. Data type: String</td>
</tr>
<tr>
<td>table</td>
<td>Name of the table in which the CSM case appears. Possible values: • sn_customerservice_case. Data type: String</td>
</tr>
<tr>
<td>thumbnail_path</td>
<td>Direct link to get thumbnail for attachment (only for attachment of type images.) Data type: String</td>
</tr>
<tr>
<td>user_full_name</td>
<td>Full name for logged-in user. Data type: String</td>
</tr>
<tr>
<td>user_login</td>
<td>User ID for logged-in user. Data type: String</td>
</tr>
<tr>
<td>user_sys_id</td>
<td>Sys_id for logged-in user. Located in User [sys_user] table. Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request
Retrieves the complete activity stream for a CSM case by sys_id.

```bash
curl --request GET
'https://instance.servicenow.com/api/sn_customerservice/case/f352dc9387632300d6b0a7da0acb0b60/activities' \
--request GET \
--header 'Accept: application/json' \
--user 'username': 'password'
```

The response shows a single activity stream entry for the specified case, an **Additional comments** journal entry submitted by the agent.

```json
{
  "result": {
    "display_value": "CS0001401",
    "sys_id": "f352dc9387632300d6b0a7da0acb0b60",
    "short_description": "Laptop is not working properly",
    "number": "CS0001401",
    "entries": [
      {
        "sys_created_on_adjusted": "2020-05-04 14:15:44",
        "sys_id": "600b6e4adb241010d045232d1396196a",
        "login_name": "John Jason",
        "user_sys_id": "da419c1fc312310015519f2974d3ae15",
        "initials": "JJ",
        "sys_created_on": "2020-05-04 21:15:44",
        "contains_code": "false",
        "field_label": "Additional comments",
        "name": "John Jason",
        "value": "Hi, Alex. Can you please send me a photo of the label on your laptop so we can send you the correct replacement?",
        "element": "comments"
      }
    ],
    "user_sys_id": "b1e0989f87232300d6b0a7da0acb0beb",
    "user_full_name": "Mark Johnson",
    "user_login": "admin",
    "label": "Case",
    "table": "sn_customerservice_case",
    "journal_fields": [
      {
        "can_read": true,
        "color": "transparent",
        "can_write": true,
```
Example: Python request
Retrieve the fourth and fifth activity stream entries for a CSM case by case number.

```python
# Install requests package for python
import requests

# Set the API endpoint URL for the request
url = "https://instance.servicenow.com/api/sn_customerservice/case/CS0030245/activities"

# Set the user credentials
user = "username"
pwd = "password"

# Set the query parameters
params = {
    "sysparm_offset": 3,
    "sysparm_limit": 2
}

# Set the HTTP headers
headers = {"Accept": "application/xml"}

# Issue the HTTP request
response = requests.get(url, auth=(user, pwd), params=params, headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    # Code for handling error response
```
The XML response includes the two requested activity stream entries from the CSM case, including an image attachment and an email message attachment.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <display_value>CS0030245</display_value>
    <short_description>Urgent issue, can't send email</short_description>
    <label>Case</label>
    <sys_id>cf866683c303310fb12d9f2974d3ae1f</sys_id>
    <number>CS0030245</number>
    <sys_created_on.adjusted>2020-05-04 17:00:00</sys_created_on.adjusted>
    <entries>
      <sys_created_on.adjusted>2020-05-04 12:38:09</sys_created_on.adjusted>
      <sys_id>8a526e421be010102ebcd56ec4bcb89</sys_id>
      <login_name>Alex Linde</login_name>
      <user.sys_id>a.linde</user.sys_id>
      <attachment>
        <path>8a526e421be010102ebcd56ec4bcb89.iix</path>
        <sys_id>8a526e421be010102ebcd56ec4bcb89</sys_id>
        <thumbnail_path>8a526e421be010102ebcd56ec4bcb89.iix?t=small</thumbnail_path>
        <size_bytes>8298</size_bytes>
        <image_height>183</image_height>
        <content_type>image/jpeg</content_type>
        <size>8.1 KB</size>
        <file_name>screenshot.jpg</file_name>
        <image_path>8a526e421be010102ebcd56ec4bcb89.iix?t=large</image_path>
        <state>available</state>
        <image_width>275</image_width>
        <average_image_color>#9f896d</average_image_color>
      </attachment>
      <initials>AL</initials>
      <sys_created_on>2020-05-04 17:38:09</sys_created_on>
      <name>user@customer</name>
    </entries>
  </result>
</response>
```
Case - GET /sn_customerservice/case/field_values/{field_name}

Retrieves the list of possible field values for a choice or reference field in the Case [sn_customerservice_case] table.

To use this endpoint, users must have the csm_ws_integration, sn_customerservice.customer, or sn_customerservice.consumer role.

Use this endpoint to populate field value lists in case creation forms.

When you request values for a reference field, any reference qualifiers defined for the requested field in the Dictionary Entry [sys_dictionary] table are applied. For more information on reference qualifiers, see Reference qualifiers.

You can use the sysparm_ref_qual_input query parameter to specify field values to use when evaluating JavaScript reference qualifiers. For an example, see the query parameter listing below.

When you request values for a choice field that is dependent on another field, you can use the sysparm_dependent_value query parameter to specify a value for the dependency parent field and restrict the returned field values accordingly. For more details on dependent fields, see Make a field dependent.

Note: You can reference all sysparm query parameters using either their full name or their shortened name (without the sysparm_ prefix). For example, for sysparm_limit you can also use limit.

URL format

Versioned URL: /api/sn_customerservice/{api_version}/case/field_values/ (field_name)

Default URL: /api/sn_customerservice/case/field_values/{field_name}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| api_version     | Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.  
Data type: String |
| field_name      | Name of a choice or reference field in the Case [sn_customerservice_case] table.  
Data type: String |
### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_dependent_value</td>
<td>Value to select in the Case [sn_customerservice_case] table choice field that the requested field is dependent on. Use only when requesting a choice field that is dependent on another field. As an example, suppose you create a custom Channel Instance choice field in the Case table which is dependent on the default Channel choice field. When sysparm_dependent_value is not specified, this endpoint returns the full set of values for your custom field. To instead return only the subset of custom field values that are available when the Phone CSM channel is chosen from the Channel field, specify: sysparm_dependent_value=Phone. Data type: String</td>
</tr>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. Unusually large sysparm_limit values can impact system performance. For requests that exceed this number of records, use the sysparm_offset parameter to paginate record retrieval. Data type: Number</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, sysparm_offset is set to &quot;0&quot;. To simply page through all available records, sysparm_offset=sysparm_offset+sysparm_limit, until you reach the end of all records. Do not pass a negative number in the sysparm_offset parameter. Data type: Number Default: 0</td>
</tr>
<tr>
<td>sysparm_reference_field_columns</td>
<td>Comma-separated list of column names, from the table of the specified reference field, to return in the response. For example, if Contact is the specified reference field, you would include all of all the fields in the Contact table to return. Data type: String</td>
</tr>
</tbody>
</table>
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **sysparm_query**     | Encoded query used to filter the result set. Queries defined using this parameter are appended to any encoded queries produced by reference qualifiers. For example:  

\`\{(sysparm_query=caller_id=javascript:gs.getUserID()^active=true)\`

The encoded query supports `order by`. To sort responses based on certain fields, use the `ORDERBY` and `ORDERBYDESC` clauses in `sysparm_query`. For example,  

\`\{sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory\`

filters all active records and orders the results in ascending order by number first, and then in descending order by category.

If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property `glide.invalid_query.returns_no_rows`. Set this property to true to return no rows on an invalid query.

⚠️ **Note:** The `glide.invalid_query.returns_no_rows` property controls the behavior of all queries across the instance, such as in lists, scripts (`GlideRecord.query()`), and web service APIs.

Data type: String

| **sysparm_ref_qual_input** | Encoded set of field values representing a current object to reference qualifiers that use JavaScript functions. Use when requesting a reference field.

For example, when requesting values from the `Asset` field you can set this query parameter value to return only assets belonging to the specified product instead of assets belonging to the product associated with the specified case:

\`\{sysparm_ref_qual_input=product=5d1b7bb053401010dc25ddeeff7b129d\`

Data type: String

Syntax: `{field1}={value1}^{field2}={value2} […]`
### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Accept** | Data format of the response body. Supported types: `application/json` or `application/xml`.  
**Default:** `application/json` |

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Link** | REST response data can be split into multiple result sets. Where applicable, the response header contains different links for the first set, previous set, next set, and the last set of records.  
**For example:**  
https://<instance name>.service-now.com/api/now/table/cmdb_ci?sysparm_offset=40&sysparm_limit=10000;rel="next",  
https://<instance name>.service-now.com/api/now/table/cmdb_ci?sysparm_offset=40&sysparm_limit=10000;rel="prev",  
https://<instance name>.service-now.com/api/now/table/cmdb_ci?sysparm_offset=0&sysparm_limit=10000;rel="first",  
https://<instance name>.service-now.com/api/now/table/cmdb_ci?sysparm_offset=2780&sysparm_limit=10000;rel="last" |

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).
**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

**Response body parameters (JSON or XML)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Array of objects in which each object describes a single value for the requested field.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: [</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;label&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>result.label</td>
<td>Display value for the field value.</td>
</tr>
<tr>
<td>result.value</td>
<td>Field value.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

Retrieve values for the contact reference field from all cases:

```bash
curl 'https://instance.servicenow.com/api/sn_customerservice/case/field_values/contact' 
--request GET 
--header 'Accept: application/json' 
--user 'username':'password'
```

The response displays labels and values for matching contacts:
Example: Python request
Retrieve values for the priority choice field from all cases:

```python
# Install requests package for python
import requests

# Set the API endpoint URL for the request
url = 'https://instance.servicenow.com/api/sn_customerservice/case/field_values/priority'

# Set the credentials
user = 'username'
pwd = 'password'

# Set the HTTP headers
```

headers = {"Accept": "application/xml"}

# Issue the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

The response shows labels and values for five matching priority field values:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <label>-- None --</label>
    <value/>
  </result>
  <result>
    <label>1 - Critical</label>
    <value>1</value>
  </result>
  <result>
    <label>2 - High</label>
    <value>2</value>
  </result>
  <result>
    <label>3 - Moderate</label>
    <value>3</value>
  </result>
  <result>
    <default_value>true</default_value>
    <label>4 - Low</label>
    <value>4</value>
  </result>
</response>
```

**Case - GET /sn_customerservice/case/{id}/field_values/{field_name}**

Retrieves the list of possible field values for a choice or reference field for a specified record in the Case [sn_customerservice_case] table.
To use this endpoint, users must have the csm_ws_integration, sn_customerservice.customer, or sn_customerservice.consumer role.

Use this endpoint to populate field value lists in case update forms.

When you request values for a reference field, any reference qualifiers defined for the requested field in the Dictionary Entry [sys_dictionary] table are applied, using values from the specified case record. For more information on reference qualifiers, see Reference qualifiers.

You can use the `sysparm_ref_qual_input` query parameter to override values from the specified case when evaluating JavaScript reference qualifiers. For an example, see the query parameter listing below.

When you request values for a choice field that is dependent on another field, this endpoint limits returned field values based on the dependency parent field value selected in the specified case record. For more details on dependent fields, see Make a field dependent.

You can use the `sysparm_dependent_value` query parameter to override the selection from the dependency parent field. For an example, see the query parameter listing below.

**Note:** You can reference all sysparm query parameters using either their full name or their shortened name (without the `sysparm_` prefix). For example, for `sysparm_limit` you can also use `limit`.

### URL format

**Versioned URL:** `/api/sn_customerservice/{api_version}/case/{id}/field_values/{field_name}`

**Default URL:** `/api/sn_customerservice/case/{id}/field_values/{field_name}`

### Supported request parameters

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>field_name</td>
<td>Name of a choice or reference field. Located in the Case [sn_customerservice_case] table. Data type: String</td>
</tr>
</tbody>
</table>
### Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Sys_id of a record. Located in the Case [sn_customerservice_case] table to use when evaluating reference qualifiers.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_dependent_value</td>
<td>Value to select in the Case [sn_customerservice_case] table choice field that the requested field is dependent on. Use only when requesting a choice field that is dependent on another choice field.</td>
</tr>
<tr>
<td></td>
<td>As an example, suppose you create a custom Channel Instance choice field in the Case table which is dependent on the default Channel choice field. When <code>sysparm_dependent_value</code> is not specified, this endpoint returns the full set of values for your custom field. To instead return only the subset of custom field values that are available when the Phone CSM channel is chosen from the Channel field, specify:</td>
</tr>
<tr>
<td></td>
<td><code>sysparm_dependent_value=Phone</code></td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. Unusually large <code>sysparm_limit</code> values can impact system performance. For requests that exceed this number of records, use the <code>sysparm_offset</code> parameter to paginate record retrieval.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks.</td>
</tr>
<tr>
<td></td>
<td>For example, the first time you call this endpoint, <code>sysparm_offset</code> is set to &quot;0&quot;. To simply page through all available records, set <code>sysparm_offset=sysparm_offset+sysparm_limit</code>, until you reach the end of all records.</td>
</tr>
<tr>
<td></td>
<td>Do not pass a negative number in the <code>sysparm_offset</code> parameter.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sysparm_query</td>
<td>Encoded query used to filter the result set. Queries defined using this parameter are appended to any encoded queries produced by reference qualifiers. For example: {(sysparm_query=caller_id=javascript:gs.getUserID()^active=true)} The encoded query supports order by. To sort responses based on certain fields, use the ORDERBY and ORDERBYDESC clauses in sysparm_query. For example, sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory filters all active records and orders the results in ascending order by number first, and then in descending order by category. If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property glide.invalid_query.returns_no_rows. Set this property to true to return no rows on an invalid query.</td>
</tr>
<tr>
<td>sysparm_reference_field_columns</td>
<td>Comma-separated list of column names, from the table specified reference field, to return in the response. For example, if Contact is the specified reference field, you would include a list of all the fields in the Contact table to return.</td>
</tr>
<tr>
<td>sysparm_ref_qual_input</td>
<td>Encoded set of field values representing a current object to reference qualifiers that use JavaScript functions. Use this parameter when requesting a reference field. For example, when requesting values from the Asset field, you can set this query parameter value to return only assets.</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_ref_qual_input=product=5d1b7bb053401010dc25ddeff7b129d</td>
<td>belonging to the specified product instead of assets belonging to the product associated with the specified case:</td>
</tr>
<tr>
<td>Data type: String</td>
<td>Syntax: {field1}={value1}^{field2}={value2}[...]</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link</td>
<td>REST response data can be split into multiple result sets. Where applicable, the response header contains different links for the first set, previous set, next set, and the last set of records. For example:</td>
</tr>
<tr>
<td></td>
<td>https://&lt;instance name&gt;.service-now.com/api/now/table/cmdb_ci?sysparm_offset=40&amp;sysparm_limit=10000&gt;;rel=&quot;next&quot;,</td>
</tr>
<tr>
<td></td>
<td>https://&lt;instance name&gt;.service-now.com/api/now/table/cmdb_ci?sysparm_offset=40&amp;sysparm_limit=10000&gt;;rel=&quot;prev&quot;,</td>
</tr>
</tbody>
</table>
Response headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Array of objects in which each object describes a single value for the requested field. Data type: Array</td>
</tr>
</tbody>
</table>

```
"result": [ 
  { 
    "label": "String", 
    "value": "String" 
  } 
]
```

result.label Display value for the field value.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type: String</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.value</td>
<td>Field value.</td>
<td></td>
</tr>
</tbody>
</table>

**Example: cURL request**

Retrieve the contact reference field value for an existing case:

```bash
curl
  'https://instance.servicenow.com/api/sn_customerservice/case/f352dc9387632300d6b0a7da0acb0b60/field_values/contact' \
  --request GET \
  --header 'Accept: application/json' \
  --user 'username':'password'
```

The response displays the label and value for the contact assigned to the specified case:

```json
{
  "result": [
    {
      "label": "Alex Linde",
      "value": "60beb5e7d7600200e5982cf65e6103ad"
    }
  ]
}
```

**Example: Python request**

Retrieve values for the asset reference field, using the case with sys_id 4ac73c23c322310015519f2974d3aeeee when evaluating reference qualifiers:

```python
# Install requests package for python
import requests

# Set the API endpoint URL for the request
url = '
  'https://instance.servicenow.com/api/sn_customerservice/case/4ac73c23c322310015519f2974d3aeeee/field_values/asset'

# Set the credentials
user = 'username'
pwd = 'password'
```
# Set the HTTP headers
headers = {"Accept": "application/json"}

# Issue the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the JSON response into a dictionary and use the data
print(response.json())

The response shows labels and values for all assets belonging to the product associated with the specified case:

```
{
    "result": [
        {
            "label": "344-9448 - ACME KX Series - KX5000",
            "value": "b8defca16fc331003b3c498f5d3ee4e7"
        }
    ]
}
```

**Case - GET /sn_customerservice/case/{id}**

Retrieves the specified Customer Service Management (CSM) case.

To use this endpoint, users must have the csm_ws_integration, sn_customerservice.customer, or sn_customerservice.consumer role.

⚠️ **Note:** You can reference all sysparm query parameters using either their full name or their shortened name (without the `sysparm_` prefix). For example, for `sysparm_limit` you can also use `limit`.

**URL format**

- Versioned URL: /api/sn_customerservice/{api_version}/case/{id}
- Default URL: /api/sn_customerservice/case/{id}
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>Sys_id or case number for the case to retrieve. Located in the Case [sn_customerservice_case] table. Results will be unpredictable if a specified</td>
</tr>
<tr>
<td></td>
<td>case number is shared by two or more cases.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_display_value</td>
<td>Data retrieval operation for reference and choice fields.</td>
</tr>
<tr>
<td></td>
<td>Based on this value, retrieves the display value and/or the actual value from the database.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Returns the display values for all fields.</td>
</tr>
<tr>
<td></td>
<td>• false: Returns the actual values from the database.</td>
</tr>
<tr>
<td></td>
<td>• all: Returns both actual and display values.</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>

**Note:** There is no preferred method for setting this parameter. However, specifying the display value may cause performance issues since it is not reading directly from the database and may include referencing other fields and records. For more information on display values and actual values, see Table API FAQs (KB0534905).
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
| 404         | Indicates that the request is invalid. Could be due to one of the following reasons:  
• Requested case does not exist.  
• User does not have access to the case. |
Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account</td>
<td>Sys_id of the account record associated with the case. Located in the Account [customer_account] table. Data type: String</td>
</tr>
</tbody>
</table>

Account details. Data type: Object

```json
"account": {
  "account_code": "String",
  "account_parent": "String",
  "account_path": "String",
  "active_escalation": "String",
  "apple_icon": Image,
  "banner_image": Image,
  "banner_image_light": Image,
  "banner_text": "String",
  "city": "String",
  "contact": "String",
  "country": "String",
  "customer": "String",
  "discount": "String",
  "fax_phone": "String",
  "fiscal_year": "String",
  "lat_long_error": "String",
  "latitude": "String",
  "longitude": "String",
  "market_cap": "String",
  "name": "String",
  "notes": "String",
  "num_employees": Number,
  "number": "String",
  "parent": "String",
  "partner": "String",
  "phone": "String",
  "primary": Boolean,
  "primary_contact": "String",
  "profits": Number,
}"
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account.account_code</td>
<td>Unique combination of values that an application uses to identify budget forecasts and budget plans. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>account.account_parent</td>
<td>Sys_id of the parent account of this account. Located in the Account [customer_account] table. Data type: String</td>
</tr>
<tr>
<td>account.account_path</td>
<td>Path from the parent to child accounts in the account hierarchy. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>account.active_escalation</td>
<td>Sys_id of the active escalation associated with the account. Located in the Escalation [sn_customerservice escalation] table. Data type: String</td>
</tr>
<tr>
<td>account.apple_icon</td>
<td>Icon for iPhone home page bookmarks. Data type: Image</td>
</tr>
<tr>
<td>account.banner_image</td>
<td>Banner image that appears on the customer portal. Data type: Image</td>
</tr>
<tr>
<td>account.banner_image_light</td>
<td>Small banner image. Data type: Image</td>
</tr>
<tr>
<td>account.banner_text</td>
<td>Banner text that appears on the customer portal. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>account.city</td>
<td>City in which the company that is associated with this account resides. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>account.contact</td>
<td>Sys_id of a contact record associated with this account. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>account.country</td>
<td>Country in which the company that is associated with this account resides. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| account.customer   | Flag that indicates whether the account is a customer account, as opposed to a partner account. Possible values:  
  • true: Customer account  
  • false: Partner account  
  Data type: Boolean  
  Default: false |
| account.discount   | Discount given to the account on purchases.  
  Data type: Number  
  Maximum length: 15 |
| account.fax_phone  | Primary fax phone number for the company associated with this account.  
  Data type: String  
  Maximum length: 40 |
| account.fiscal_year| Fiscal year for the company associated with the account.  
  Data type: String |
| account.lat_long_error | Difference in the actual location as compared to latitude and longitude information.  
  Data type: String  
  Maximum length: 1,000 |
| account.latitude   | Latitude of the company associated with this account.  
  Data type: Number (floating point number)  
  Maximum length: 40 |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account.longitude</td>
<td>Longitude of the company associated with this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (floating point number)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.manufacturer</td>
<td>Flag that indicates whether the company associated with this account manufactures goods.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Manufactures goods</td>
</tr>
<tr>
<td></td>
<td>• false: Does not manufacture goods</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>account.market_cap</td>
<td>Market value of the associated company's publicly traded stock shares.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (currency)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 20</td>
</tr>
<tr>
<td>account.name</td>
<td>Name of the company associated with this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 80</td>
</tr>
<tr>
<td>account.notes</td>
<td>Additional information about the company.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>account.num_employees</td>
<td>Number of people employed by the company.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>account.number</td>
<td>Number that identifies this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.parent</td>
<td>Sys_id of the parent account of this account. Located in the Company [core_company] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>account.partner</td>
<td>Flag that indicates whether the account is a partner account or a customer account.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Partner account</td>
</tr>
<tr>
<td></td>
<td>• false: Customer account</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>account.phone</td>
<td>Primary phone number for the company.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>account.primary</td>
<td>Flag that indicates whether the account is a primary account.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Primary account</td>
</tr>
<tr>
<td></td>
<td>• false: Secondary account</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>account.primary_contact</td>
<td>Sys_id of the primary contact for the account. Located in the Contact [customer_contact] table.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>account.profits</strong></td>
<td>Profit information entered for this account. Data type: Number (Currency) Maximum length: 40</td>
</tr>
</tbody>
</table>
| **account.publicly_traded**| Flag that indicates whether the company associated with this account is publicly traded on the stock exchange. Possible values:  
  • true: Publicly traded  
  • false: Private company  
  Data type: Boolean |
| **account.rank_tier**       | Type of account. Possible values:  
  • blacklist  
  • strategic  
  • tactical  
  • valued  
  • other  
  Data type: String Maximum length: 40 |
| **account.registration_code** | Unique code that customers use when requesting a login on the customer portal. This provides a method for validating the customer on the company before granting access.  
  Data type: String Maximum length: 40 |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account.revenue_per_year</td>
<td>Revenue produced by the company associated with this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Currency)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 20</td>
</tr>
<tr>
<td>account.state</td>
<td>State in which the company resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.stock_price</td>
<td>Price of the company stock.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.stock_symbol</td>
<td>Stock symbol of the company.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.street</td>
<td>Street address of the company.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>account.sys_class_name</td>
<td>Table that contains the associated account record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>account.sys_created_by</td>
<td>User that originally created the account.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.sys_created_on</td>
<td>Date and time that the account was originally created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>account.sys_id</td>
<td>Sys_id for the account record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>account.sys_mod_count</td>
<td>Number of times the account information has been updated. Data type: Number (Integer)</td>
</tr>
<tr>
<td>account.sys_updated_by</td>
<td>User that last modified the account information. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>account.sys_updated_on</td>
<td>Date and time the account information was last updated. Data type: String</td>
</tr>
<tr>
<td>account.theme</td>
<td>Sys_id of the customer portal theme used by this account. Located in the Theme [sys_ui_theme] table. Data type: String</td>
</tr>
<tr>
<td>account.vendor</td>
<td>Flag that indicates whether the company associated with the account is a vendor. Possible values: • true: Vendor • false: Not a vendor Data type: Boolean Default: false</td>
</tr>
<tr>
<td>account.vendor_manager</td>
<td>List of sys_ids of the vendor managers for the account. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>account.vendor_type</td>
<td>List of sys_ids of the type of vendor such as, applications, hardware, services, or software. Located in the Vendor Type [vendor_type] table.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>account.website</td>
<td>URL of the website for the company.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 1,024</td>
</tr>
<tr>
<td>account.zip</td>
<td>Zip code of the company.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>active</td>
<td>Flag that indicates whether the case is open and active.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Case is active</td>
</tr>
<tr>
<td></td>
<td>• false: Case is closed</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td>active_account_escalation</td>
<td>Sys_id of the active account escalation record associated with the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>active_escalation</td>
<td>Sys_id of the active account escalation record associated with the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>activity_due</td>
<td>Date for which the associated case is expected to be completed.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>additional_assignee_list</td>
<td>List of the sys_ids of the persons (other than primary assignee) that have been assigned to the account. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>Maximum: 4,000</td>
</tr>
<tr>
<td>approval</td>
<td>String that describes the type of approval required. Possible values: • approved • cancelled • duplicate • not_required • not requested • rejected • requested</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum characters: 40</td>
</tr>
<tr>
<td></td>
<td>Default: not requested</td>
</tr>
<tr>
<td>approval_history</td>
<td>List of all approvals associated with the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum characters: 4,000</td>
</tr>
<tr>
<td>approval_set</td>
<td>Date and time that the associated action was approved.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>asset</td>
<td>Sys_id of the asset record associated with the case. Located in the Asset [alm_asset] table.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>assigned_to</td>
<td>Sys_id of the person assigned to the case. Located in the <code>sys_user</code> table.</td>
</tr>
<tr>
<td>assignment_group</td>
<td>Sys_id of the customer service agent group assigned to the case. Located in the <code>sys_user_group</code> table.</td>
</tr>
<tr>
<td>business_duration</td>
<td>Length in calendar work hours, work days, and work weeks that it took to complete the case.</td>
</tr>
<tr>
<td>business_impact</td>
<td>Impact of the issue on the associated customer. Maximum characters: 4,000</td>
</tr>
<tr>
<td>business_service</td>
<td>Sys_id of the service record associated with the case. Located in the <code>cmdb_ci_service</code> table.</td>
</tr>
<tr>
<td>case</td>
<td>Case short description and case number. Maximum length: 300</td>
</tr>
<tr>
<td>category</td>
<td>Case category.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Question</td>
</tr>
<tr>
<td></td>
<td>• 1: Issue</td>
</tr>
<tr>
<td></td>
<td>• 2: Feature</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 1</td>
</tr>
<tr>
<td>cause</td>
<td>Details about the cause of the problem.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>caused_by</td>
<td>Sys_id of the change request that caused the case to be created.</td>
</tr>
<tr>
<td></td>
<td>Located in the Change Request [change_request] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>change</td>
<td>Sys_id of the change request that caused the case to be created.</td>
</tr>
<tr>
<td></td>
<td>Located in the Change Request [change_request] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>child_case_creation_progress</td>
<td>Flag that indicates whether the case is a child case that was created from a major case.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Child case created from a major case</td>
</tr>
<tr>
<td></td>
<td>• false: Not a child case</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>closed_at</td>
<td>Date and time that the case was closed.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>closed_by</td>
<td>Sys_id of the user that closed the case. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>close_notes</td>
<td>Notes made when the case was closed. Data type: String</td>
</tr>
<tr>
<td>cmdb_ci</td>
<td>Sys_id of the configuration item associated with the case. Located in the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>comments</td>
<td>Additional comments about the case. Data type: String</td>
</tr>
<tr>
<td>comments_and_work_notes</td>
<td>Comments and work notes entered for the case. Data type: String</td>
</tr>
<tr>
<td>company</td>
<td>Sys_id of the company associated with the case. Located in the Company [core_company] table. Data type: String</td>
</tr>
<tr>
<td>consumer</td>
<td>Business-to-consumer cases only. Sys_id of the person to contact with regards to this case. Located in the Consumer [csm_consumer] table. Data type: String</td>
</tr>
<tr>
<td>consumer</td>
<td>Array of parameters for business-to-consumer cases.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| consumer.active       | Flag that indicates whether the consumer is active. Possible values:  
  • true: Consumer active  
  • false: Consumer de-activated  
  Data type: Boolean  
  Default: true |
| consumer.business_phone | Business phone number of the consumer.  
  Data type: String  
  Maximum length: 40 |
| consumer.city         | City in which the consumer resides.  
  Data type: String  
  Maximum length: 100 |
| consumer.country      | Country in which the consumer resides.  
  Data type: String  
  Maximum length: 40 |
| consumer.date_format  | Format in which to display dates. Valid values:  
  • dd-mm-yyyy  
  • dd/mm/yyyy  
  • dd.mm.yyyy  
  • mm-dd-yyyy  
  • yyyy-mm-dd  
  Data type: String  
  Maximum length: 40 |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Default: blank (system date format)</td>
</tr>
<tr>
<td>consumer.email</td>
<td>Email address of the consumer. Data type: String. Maximum length: 100</td>
</tr>
<tr>
<td>consumer.fax</td>
<td>Fax number of the consumer. Data type: String. Maximum length: 40</td>
</tr>
<tr>
<td>consumer.first_name</td>
<td>Consumer first name. Data type: String. Maximum length: 50</td>
</tr>
<tr>
<td>consumer.gender</td>
<td>Gender of the consumer. Data type: String. Maximum length: 40</td>
</tr>
<tr>
<td>consumer.home_phone</td>
<td>Home phone number of the consumer. Data type: String. Maximum length: 40</td>
</tr>
<tr>
<td>consumer.household</td>
<td>Sys_id of the record that describes the household characteristics. Located in the Household [csm_household] table. Data type: String</td>
</tr>
<tr>
<td>consumer.last_name</td>
<td>Consumer last name. Data type: String. Maximum length: 50</td>
</tr>
<tr>
<td>consumer.middle_name</td>
<td>Consumer middle name. Data type: String. Maximum length: 50</td>
</tr>
<tr>
<td>consumer.mobile_phone</td>
<td>Consumer mobile phone.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consumer.name</td>
<td>Consumer full name; first_name + middle_name + last_name.</td>
</tr>
<tr>
<td>consumer.notes</td>
<td>Notes on consumer.</td>
</tr>
<tr>
<td>consumer.notification</td>
<td>Indicates whether the consumer should receive notifications.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Disabled</td>
</tr>
<tr>
<td></td>
<td>• 2: Enabled</td>
</tr>
<tr>
<td>consumer.number</td>
<td>Unique number associated with the consumer.</td>
</tr>
<tr>
<td>consumer.photo</td>
<td>Photo of the consumer.</td>
</tr>
<tr>
<td>consumer.preferred_language</td>
<td>Consumer primary language.</td>
</tr>
<tr>
<td>consumer.prefix</td>
<td>Consumer name prefix such as Dr., Mr., Mrs., or Ms.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| consumer.primary      | Flag that indicates whether the consumer is the primary consumer. Possible values:  
|                       | • true: Primary consumer  
|                       | • false: Not primary consumer  
|                       | Data type: Boolean  
|                       | Default: false |
| consumer.state        | State in which the consumer resides. Data type: String  
|                       | Maximum length: 100 |
| consumer.street       | Consumer street address. Data type: String  
|                       | Maximum length: 255 |
| consumer.suffix       | Consumer name suffix such as Jr., Sr., or II. Data type: String |
| consumer.sys_created_by | User that created the consumer record. Data type: String  
|                       | Maximum length: 40 |
| consumer.sys_created_on | Date and time the consumer record was originally created. Data type: String |
| consumer.sys_domain   | ServiceNow domain in which the consumer information resides. Data type: String  
<p>| consumer.sys_id       | Unique identifier for the consumer. Data type: String |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>consumer.sys_mod_count</td>
<td>Number of times that the associated consumer information has been modified.</td>
</tr>
<tr>
<td></td>
<td>Data type: Integer</td>
</tr>
<tr>
<td>consumer.sys_updated_by</td>
<td>User that last updated the consumer information.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.sys_updated_on</td>
<td>Date and time when the consumer information was last updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>consumer.time_format</td>
<td>Format in which to display time.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• hh.mm.ss a: hh.mm.ss</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>• HH.mm.ss: hh.mm.ss</td>
</tr>
<tr>
<td></td>
<td>• HH:mm:ss: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system time format)</td>
</tr>
<tr>
<td>consumer.time_zone</td>
<td>Consumer time zone, such as Canada/Central or US/Eastern.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.title</td>
<td>Consumer business title such as Manager, Software Developer, or Contractor.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 60</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consumer.user</td>
<td>Sys_id of the consumer user. Located in the Consumer [csm_consumer_user] table. Data type: String</td>
</tr>
<tr>
<td>consumer.zip</td>
<td>Consumer zip code. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>contact</td>
<td>Business-to-business based cases only. Sys_id of the person to contact regarding this case. Located in the Contact [customer_contact] table. Data type: String</td>
</tr>
<tr>
<td>contact</td>
<td>Array of contact parameters for business-to-business cases.</td>
</tr>
<tr>
<td>contact.account</td>
<td>Sys_id of the account record to which the contact is associated; Account [customer_account] table. Data type: String</td>
</tr>
</tbody>
</table>
| contact.active   | Flag that indicates whether the contact is active within the system. Possible values:  
  • true: Contact is active  
  • false: Contact is inactive  
  Data type: Boolean Default: true |
| contact.agent_status | Status of the agent. Possible values:  
  • Off work  
  • On break |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• On route</td>
</tr>
<tr>
<td></td>
<td>• On site</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.building</td>
<td>Sys_id of the record that describes the building in which the contact resides; Building [cmn_building] table.</td>
</tr>
<tr>
<td>contact.calendar_integration</td>
<td>Calendar application that the contact uses.</td>
</tr>
<tr>
<td></td>
<td>• 1: Outlook</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 1</td>
</tr>
<tr>
<td>contact.city</td>
<td>City in which the contact resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.company</td>
<td>Sys_id of the company to which the contact is associated; Company [core_company] table.</td>
</tr>
<tr>
<td>contact.cost_center</td>
<td>Sys_id of the cost center associated with the contact; Cost Center [cmn_cost_center] table.</td>
</tr>
<tr>
<td>contact.country</td>
<td>Country code of the country in which the contact resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 3</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| contact.date_format           | Format in which to display dates to contacts.  
Valid values:  
• dd/mm/yyyy  
• dd-mm-yyyy  
• dd.mm.yyyy  
• mm-dd-yyyy  
• yyyy-mm-dd  
Data type: String  
Maximum length: 40  
Default: blank (system date format) |
| contact.default_perspective   | Sys_id of the default perspective for the contact. Located in the Menu List [sys_perspective] table.  
Data type: String |
| contact.department           | Sys_id of the department associated with the contact. Located in the Department [cmn_department] table.  
Data type: String |
| contact.edu_status            | Education status of the associated contact.  
Data type: String  
Maximum length: 40  
Default: faculty |
| contact.email                 | Contact email address.  
Data type: String |
| contact.employee_number       | Contact employee number.  
Data type: String |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contact.enable_multifactor_authn</td>
<td>Flag that indicates whether multifactor authorization is required for the contact to log in to the service portal. Possible values: true: Multifactor authorization enabled, false: Multifactor authorization disabled. Data type: Boolean Default: false</td>
</tr>
<tr>
<td>contact.failed_attempts</td>
<td>Number of failed log in attempts. Data type: Number (Integer)</td>
</tr>
<tr>
<td>contact.first_name</td>
<td>Contact first name. Data type: String Maximum length: 50</td>
</tr>
<tr>
<td>contact.gender</td>
<td>Contact gender. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>contact.geolocation_tracked</td>
<td>Flag that indicates whether the contact location is obtained through geotracking. Possible values: true: Contact location obtained through geotracking, false: Contact location not obtained through geotracking. Data type: Boolean Default value: false</td>
</tr>
<tr>
<td>contact.home_phone</td>
<td>Contact home phone. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.internal_integration_user</td>
<td>Flag that indicates whether the contact is an internal integration user.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Internal integration user</td>
</tr>
<tr>
<td></td>
<td>• false: Other type of user</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>contact.introduction</td>
<td>Introduction</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.last_login</td>
<td>Date on which the contact last logged into the system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date)</td>
</tr>
<tr>
<td>contact.last_login_device</td>
<td>Date the consumer used the last time they logged in to the system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.last_login_time</td>
<td>Date and time the contact last logged into the system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date/time)</td>
</tr>
<tr>
<td>contact.last_name</td>
<td>Contact last name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>contact.last_position_update</td>
<td>Date and time the last position was updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date/time)</td>
</tr>
<tr>
<td>contact.latitude</td>
<td>Latitude coordinate of the contact.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.ldap_server</td>
<td>Sys_id of the LDAP server used by the contact to last log in to the system; LDAP Server [ldap_server_config] table. Data type: String.</td>
</tr>
<tr>
<td>contact.location</td>
<td>Sys_id of the record that describes the location of the contact; Location [cmn_location] table. Data type: String.</td>
</tr>
<tr>
<td>contact.locked_out</td>
<td>Flag that indicates if the contact is locked-out. Possible values: true: Contact locked-out, false: Contact not locked-out. Data type: Boolean.</td>
</tr>
<tr>
<td>contact.longitude</td>
<td>Longitude coordinate of the contact. Data type: Number (Floating point). Maximum length: 40.</td>
</tr>
<tr>
<td>contact.manager</td>
<td>Sys_id of the record that describes the direct supervisor of the contact; User [sys_user] table. Data type: String.</td>
</tr>
<tr>
<td>contact.middle_name</td>
<td>Contact middle name. Data type: Number (Floating point).</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.mobile_phone</td>
<td>Contact mobile phone number. Data type: String. Maximum length: 40</td>
</tr>
<tr>
<td>contact.name</td>
<td>Contact full name. Data type: String. Maximum length: 151</td>
</tr>
<tr>
<td>contact.notification</td>
<td>Indicates whether the contact should receive notifications. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Disabled</td>
</tr>
<tr>
<td></td>
<td>• 2: Enabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer). Default: 2</td>
</tr>
<tr>
<td>contact.on_schedule</td>
<td>Indicates the timeliness of dispatched service personnel. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• Ahead: Ahead of schedule</td>
</tr>
<tr>
<td></td>
<td>• behind_less30: Behind schedule, but less than 30 minutes</td>
</tr>
<tr>
<td></td>
<td>• behind_30to60: Behind schedule between 30 and 60 minutes</td>
</tr>
<tr>
<td></td>
<td>• behind_more60: Behind schedule more than 60 minutes</td>
</tr>
<tr>
<td></td>
<td>• on_time: On schedule</td>
</tr>
<tr>
<td></td>
<td>Data type: String. Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.phone</td>
<td>Contact business phone number. Data type: String, Maximum length: 40</td>
</tr>
<tr>
<td>contact.photo</td>
<td>Photo image of the contact. Data type: Image</td>
</tr>
<tr>
<td>contact.preferred_language</td>
<td>Country code of the contact's primary language. Data type: String, Maximum length: 3</td>
</tr>
<tr>
<td>contact.roles</td>
<td>List of user roles associated with the contact. Data type: String, Maximum length: 40</td>
</tr>
<tr>
<td>contact.schedule</td>
<td>Sys_id of the record that describes the work schedule for the associated contact. Schedule [cmn_schedule]. Data type: String</td>
</tr>
<tr>
<td>contact.source</td>
<td>Source of the contact. Data type: String, Maximum length: 255</td>
</tr>
<tr>
<td>contact.state</td>
<td>State in which the contact resides. Data type: String, Maximum length: 40</td>
</tr>
<tr>
<td>contact.street</td>
<td>Contact street address. Data type: String, Maximum length: 255</td>
</tr>
<tr>
<td>contact.sys_class_name</td>
<td>Table that contains the record. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>contact.sys_created_by</td>
<td>User that originally created the associated contact record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.sys_created_on</td>
<td>Data and time the associated contact was originally created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date/Time)</td>
</tr>
<tr>
<td>contact.sys_domain</td>
<td>ServiceNow instance domain of the associated contact record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>contact.sys_domain_path</td>
<td>Contact record domain path.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td></td>
<td>Default: / (global)</td>
</tr>
<tr>
<td>contact.sys_id</td>
<td>Unique identifier for the associated contact record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>contact.sys_mod_count</td>
<td>Number of times that the associated contact record has been modified.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td>contact.sys_tags</td>
<td>System tags for the content.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>contact.sys_updated_by</td>
<td>User that last updated the associated contact information.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.sys_updated_on</td>
<td>Data and time the associated contact information was updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date/Time)</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.time_format</td>
<td>Format in which to display time. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>• HH:mm:ss: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: Blank (system time format)</td>
</tr>
<tr>
<td>contact.time_sheet_policy</td>
<td>Sys_id of the record that contains the time sheet policy for the associated contact; Time Sheet Policy [time_sheet_policy]</td>
</tr>
<tr>
<td>contact.time_zone</td>
<td>Time zone in which the contact resides, such as Canada/Central or US/Eastern.</td>
</tr>
<tr>
<td>contact.title</td>
<td>Contact business title such as Manager, Software Developer, or Contractor.</td>
</tr>
<tr>
<td>contact.user_name</td>
<td>Contact user ID.</td>
</tr>
<tr>
<td>contact.vip</td>
<td>Flag that indicates whether the associated contact has VIP status.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.web_service_access_only</td>
<td>Flag that indicates whether the contact can only access services through the web. Possible values: Data type: Boolean Default: false</td>
</tr>
<tr>
<td>contact.zip</td>
<td>Contact zip code. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>contact_local_time</td>
<td>Contact local time. Data type: String Maximum length: 70</td>
</tr>
<tr>
<td>contact_time_zone</td>
<td>Time zone of the contact associated with the case. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>contact_type</td>
<td>Method in which the case was initially reported. Possible values: chat email phone</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contract</td>
<td>Sys_id of the contract associated with the case. Located in the Contract [ast_contract] table. This contract contains information about the type of support that is provided to the company associated to the case. A contract can include a company and contact, specific assets that are covered, and multiple service entitlements and SLAs. Data type: String</td>
</tr>
</tbody>
</table>
| correlation_display | Correlation display.  
Data type: String  
Maximum length: 100 |
| correlation_id   | Correlation identifier.  
Data type: String  
Maximum length: 100 |
Data type: String |
Data type: String |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| description | Detailed description of the problem associated with the case. | Data type: String  

Maximum length: 4,000 |
| due_date   | Date that the case is due to be closed.                                                                                                     |
Entitlements define the type of support that a customer receives as well as the supported communication channels. An entitlement can be associated with a product, an asset, an account, or a contract. |
| escalation | Current escalation level.  
Possible values:  
• 0: Normal  
• 1: Moderate  
• 2: High  
• 3: Overdue                                                                                                                                            |
| expected_start | Date and time when work is scheduled to begin on the case.                                                                                   |
|              | Data type: String  

Default: 0 |
<table>
<thead>
<tr>
<th><strong>Element</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
</table>
| first_response_time | Date and time when the first action was taken on the case.  
Data type: String                                                                                                                                       |
| follow_the_sun   | Flag that indicates whether the case should be handed off for global follow-up.  
If a customer enters additional comments on a Priority 1 - Critical or a Priority 2 - High case, or if the case is escalated, the flag is automatically set to true.  
Possible values:  
• true: Case should be handed-off for global follow-up  
• false: Case should not be handed-off for global follow-up  
Data type: Boolean  
Default: false                                                                                     |
| follow_up        | Date and time of the next follow up action.  
Data type: String                                                                                                                                       |
| group_list       | List of sys_ids of the group records associated with the case. Located in the Group [sys_user_group] table.  
Data type: Array                                                                                                                                       |
| impact           | Impact on customer.  
Possible values:  
• 1: High  
• 2: Medium  
• 3: Low  
Data type: Number (Integer)                                                                                                                                  |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| knowledge          | Flag that indicates if the knowledge base article is available for the specified issue. Possible values:  
|                    | • true: Knowledge base article is available for this issue  
|                    | • false: Knowledge base article is not available for this issue  
|                    | Data type: Boolean  
| Default: false     |                                                                            |
| location           | Sys_id of the record describing the company location located in the Location [cmn_location] table.  
|                    | Data type: String                                                        |
|                    |                                                                            |
| made_sla           | Flag that indicates whether the case was resolved in alignment with the associated service level agreement. Possible values:  
|                    | • true: Case was resolved in alignment with SLA  
|                    | • false: Case was not resolved according to the SLA  
|                    | Data type: Boolean  
<p>| Default: true      |                                                                            |
| major_case_state   | Current state of the major case. Possible values:                         |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• accepted: Initial state when a manager creates a major case or when a manager promotes a candidate case.</td>
</tr>
<tr>
<td></td>
<td>• canceled: Case is canceled.</td>
</tr>
<tr>
<td></td>
<td>• proposed: Initial state when an agent or manager creates or proposes a candidate case.</td>
</tr>
<tr>
<td></td>
<td>• rejected: Manager rejected a candidate case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>needs_attention</td>
<td>Flag that indicates whether the case needs attention.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Case needs additional attention</td>
</tr>
<tr>
<td></td>
<td>• false: Case does not need additional attention</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>notes_to_comments</td>
<td>Flag that indicates whether to add the resolution notes to the comments.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Resolutions notes, when added, are also added to comments</td>
</tr>
<tr>
<td></td>
<td>• false: Resolution notes in comments are not required</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| notify         | Method to use to notify contact/consumer. Possible values:  
|                | • 1: Do not notify  
|                | • 2: Send email  
|                | • 3: Telephone  
|                | Data type: Number (Integer)  
|                | Default: 1  |
| number         | Case number.  
|                | Data type: String  
|                | Maximum length: 40  |
| opened_at      | Date and time that the case was opened.  
|                | Data type: String  |
| opened_by      | Sys_id of the person that initially opened the case. Located in the User [sys_user] table.  
|                | Data type: String  |
| order          | Order of the case.  
|                | Data type: Number (Integer)  |
| parent         | Sys_id of the parent case to which this case (child) is associated. Located in the Task [task] table.  
|                | Data type: String  |
| partner        | Sys_id of the partner associated with the case. Located in the Account [customer_account] table.  
|                | Data type: String  |
| partner_contact| Sys_id of the partner contact associated with the case.  
<p>|                | Data type: String  |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| priority          | Priority of the case. Possible values:  
- 1: Critical  
- 2: High  
- 3: Moderate  
- 4: Low  
Data type: Number (Integer)  
Default: 4 |
| probable_cause    | Possible cause of the issue associated with the case.  
Data type: String  
Maximum length: 4,000 |
| problem           | Sys_id of the issue that the customer is encountering. Located in the Problem [problem] table.  
Data type: String |
| product           | Sys_id of the product model of the asset associated to the case. Located in the Product Model [cmdb_model] table. A model is a specific version or configuration of an asset (for example, Apple Mac Book Pro).  
Data type: String |
| reassignment_count| Number of times that the case was reassigned to a person that is responsible for moving the case forward.  
Data type: Number (Integer) |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>recipient_list</td>
<td>Sys_id of the record that contains the list of recipients for communications about this case. Located in the Recipients [sn_publications_recipients_list] table.</td>
</tr>
<tr>
<td>rejection_goto</td>
<td>Sys_id of the task to execute if the case is rejected. Located in the Task [task] table.</td>
</tr>
<tr>
<td>resolution_code</td>
<td>Resolution state for the case, such as &quot;Solved - Fixed by Support/Guidance provided&quot;.</td>
</tr>
<tr>
<td>resolved_at</td>
<td>Date and time that the case was resolved.</td>
</tr>
<tr>
<td>resolved_by</td>
<td>Sys_id of the person that resolved the case. Located in the [sys_user] table.</td>
</tr>
<tr>
<td>result</td>
<td>Array of objects in which each object describes a single value for the requested field.</td>
</tr>
<tr>
<td>result.label</td>
<td>Display value for the field.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.value</td>
<td>Field value.</td>
</tr>
<tr>
<td>short_description</td>
<td>Concise description of the case.</td>
</tr>
<tr>
<td>skills</td>
<td>List of the unique identifiers (sys_id) of the skills needed to complete the case. Located in the Skill [cmn_skill] table.</td>
</tr>
<tr>
<td>sla_due</td>
<td>Date/time at which the case must be closed based on the associated service level agreement.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile</td>
<td>Array of objects that contain details for a specific social media profile, such as Facebook or YouTube.</td>
</tr>
</tbody>
</table>

```
{sn_app_cs_social_social_profile: [
  {
    "account": "String",
    "channel": [Array],
    "contact": {Object},
    "consumer": {Object},
    "created_on": "String",
    "profile": "String",
    "profile_url": "String",
    "social_id": "String",
    "source": "String",
    "sys_created_by": "String",
    "sys_created_on": "String",
    "sys_id": "String",
    "sys_mod_count": Number
  }
]}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.account</td>
<td>Unique identifier of the record containing account information for the social media profile. Located in the Account [customer_account] table. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel</td>
<td>Details on the associated social media profile channel. Data type: Object</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.name</td>
<td>Name of the social media channel. Data type: String Maximum length: 100</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_class_name</td>
<td>Table that contains the social media channel record. Data type: String Maximum length: 80</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_created_by</td>
<td>Person that created the social media channel.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_created_on</td>
<td>Date and time the social media profile was created.</td>
</tr>
<tr>
<td></td>
<td>Data type: Date/time</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_id</td>
<td>Unique identifier of the associated social media profile channel.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_mod_count</td>
<td>Number of times that information was modified for the associated social media</td>
</tr>
<tr>
<td></td>
<td>profile channel.</td>
</tr>
<tr>
<td></td>
<td>Data type: Integer</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_name</td>
<td>System name of channel.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_package</td>
<td>Unique identifier of the record that contains information about the package</td>
</tr>
<tr>
<td></td>
<td>associated with the profile; Package [sys_package] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Reference</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_policy</td>
<td>System protection policy.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• protected</td>
</tr>
<tr>
<td></td>
<td>• read</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_scope</td>
<td>Unique identifier of the record that contains information about the scope of the social profile. Application [sys_scope] table. Data type: Reference</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_tags</td>
<td>System tags associated with the channel. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_update_name</td>
<td>Name of the person that last updated the social media profile channel. Data type: String Maximum length: 250</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_updated_by</td>
<td>User that last updated the social media profile channel. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_updated_on</td>
<td>Date and time the social media profile channel was last updated. Data type: Date/time Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer</td>
<td>For business-to-consumer cases. Details about the consumer associated with the case. Data type: Object</td>
</tr>
</tbody>
</table>

```
"consumer": {
    "active": Boolean,
    "business_phone": "String",
    "city": "String",
    "country": "String",
    "date_format": "String",
    "email": "String",
    "fax": "String",
    "first_name": "String",
    "gender": "String",
    "last_name": "String",
    "phone": "String",
    "state": "String",
    "street1": "String",
    "street2": "String",
    "timezone": "String",
    "zip": "String"
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;home_phone&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;household&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;last_name&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;middle_name&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;mobile_phone&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;name&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;notes&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;notification&quot;: Number,</td>
<td></td>
</tr>
<tr>
<td>&quot;number&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;photo&quot;: Image,</td>
<td></td>
</tr>
<tr>
<td>&quot;preferred_language&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;prefix&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;primary&quot;: Boolean,</td>
<td></td>
</tr>
<tr>
<td>&quot;state&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;street&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;suffix&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_created_by&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_created_on&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_domain&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_mod_count&quot;: Number,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_tags&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_updated_by&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_updated_on&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;time_format&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;time_zone&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;title&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;user&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;zip&quot;: &quot;String&quot;</td>
<td></td>
</tr>
</tbody>
</table>

**sn_app_cs_social_social_profile.consumer.active**

Flag that indicates whether the consumer is active.

Possible values:
- true: Consumer active
- false: Consumer de-activated

Data type: Boolean
Default: true
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| sn_app_cs_social_social_profile.consumer.business_phone                | Business phone number of the consumer.  
Data type: String  
Maximum length: 40 |
| sn_app_cs_social_social_profile.consumer.city                          | City in which the consumer resides.  
Data type: String  
Maximum length: 100 |
| sn_app_cs_social_social_profile.consumer.country                       | Country in which the consumer resides.  
Data type: String  
Maximum length: 40 |
| sn_app_cs_social_social_profile.consumer.date_format                   | Format in which to display dates.  
Valid values:  
• dd-mm-yyyy  
• dd/mm/yyyy  
• dd.mm.yyyy  
• mm-dd-yyyy  
• yyyy-mm-dd  
Data type: String  
Maximum length: 40  
Default: blank (system date format) |
| sn_app_cs_social_social_profile.consumer.email                         | Email address of the consumer.  
Data type: String  
Maximum length: 100 |
| sn_app_cs_social_social_profile.consumer.fax                           | Fax number of the consumer.  
Data type: String  
Maximum length: 40 |
<p>| sn_app_cs_social_social_profile.consumer.first_name                    | Consumer first name.                                                                                                                                 |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.consumer.gender</td>
<td>Gender of the consumer.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.home_phone</td>
<td>Home phone number of the consumer.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.household</td>
<td>Sys_id of the record that describes the household characteristics. Located in the Household [csm_household] table.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.last_name</td>
<td>Consumer last name.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.middle_name</td>
<td>Consumer middle name.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.mobile_phone</td>
<td>Consumer mobile phone number.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.name</td>
<td>Consumer full name; first_name +middle_name+last_name.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.notes</td>
<td>Notes on consumer.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.notification</td>
<td>Indicates whether the consumer should receive notifications. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Disabled</td>
</tr>
<tr>
<td></td>
<td>• 2: Enabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Integer</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: 2</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.number</td>
<td>Unique number associated with the consumer. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.photo</td>
<td>Photo of the consumer. Data type: Image</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.preferred_language</td>
<td>Consumer primary language. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 3</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.prefix</td>
<td>Consumer name prefix such as, Dr., Mr., Mrs., or Ms. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.primary</td>
<td>Flag that indicates whether the consumer is the primary consumer. Possible</td>
</tr>
<tr>
<td></td>
<td>values:</td>
</tr>
<tr>
<td></td>
<td>• true: Primary consumer</td>
</tr>
<tr>
<td></td>
<td>• false: Not primary consumer</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| `sn_app_cs_social_social_profile.consumer.state` | State in which the consumer resides.  
Data type: String  
Maximum length: 100 |
| `sn_app_cs_social_social_profile.consumer.street` | Consumer street address.  
Data type: String  
Maximum length: 255 |
| `sn_app_cs_social_social_profile.consumer.suffix` | Consumer name suffix such as Jr., Sr., or II.  
Data type: String |
| `sn_app_cs_social_social_profile.consumer.sys_created_by` | User that created the consumer record.  
Data type: String  
Maximum length: 40 |
| `sn_app_cs_social_social_profile.consumer.sys_created_on` | Date and time the consumer record was originally created.  
Data type: String |
| `sn_app_cs_social_social_profile.consumer.sys_domain` | ServiceNow domain in which the consumer information resides.  
Data type: String |
| `sn_app_cs_social_social_profile.consumer.sys_id` | Unique identifier for the consumer.  
Data type: String |
| `sn_app_cs_social_social_profile.consumer.sys_mod_count` | Number of times that the associated consumer information has been modified.  
Data type: Integer |
| `sn_app_cs_social_social_profile.consumer.sys_tags` | Consumer system tags |
| `sn_app_cs_social_social_profile.consumer.sys_updated_by` | User that last updated the consumer information.  
Data type: String  
Maximum length: 40 |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_updated_on</td>
<td>Date and time when the consumer information was last updated. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.time_format</td>
<td>Format in which to display time. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• hh.mm.ss a: hh.mm.ss</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>• HH.mm.ss: hh.mm.ss</td>
</tr>
<tr>
<td></td>
<td>• HH:mm:ss: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system time format)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.time_zone</td>
<td>Consumer time zone, such as Canada/Central or US/Eastern. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.title</td>
<td>Consumer business title, such as Manager, Software Developer, or Contractor. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 60</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.user</td>
<td>Sys_id of the consumer user. Located in the Consumer User [csm_consumer_user] table. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.zip</td>
<td>Consumer zip code. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact</td>
<td>Unique identifier of the record containing details about the social media profile that belongs to the contact associated with the case. Located in the [customer_contact] table. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.account</td>
<td>Sys_id of the account record to which the contact is associated; Account [customer_account] table. Data type: String</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.contact.active | Flag that indicates whether the contact is active within the system. Possible values:  
• true: Contact is active  
• false: Contact is inactive  
Data type: Boolean  
Default: true |
| sn_app_cs_social_social_profile.contact.agent_status | Status of the agent. Possible values:  
• Off work  
• On break  
• On route  
• On site  
Data type: String  
Maximum length: 40 |
<p>| sn_app_cs_social_social_profile.contact.building | Sys_id of the record that describes the building |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>the contact resides; Building [cmn_building] table.</td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.contact.calendar_integration | Calendar application that the contact uses.  
  • 1: Outlook | Data type: Number (Integer)  
  Default: 1 |
| sn_app_cs_social_social_profile.contact.city | City in which the contact resides. | Data type: String  
  Maximum length: 40 |
| sn_app_cs_social_social_profile.contact.company | Sys_id of the company to which the contact is associated; Company [core_company] table. | Data type: String |
| sn_app_cs_social_social_profile.contact.cost_center | Sys_id of the cost center associated with the contact; Cost Center [cmn_cost_center] table. | Data type: String |
| sn_app_cs_social_social_profile.contact.country | Country code of the country in which the contact resides. | Data type: String  
  Maximum length: 3 |
| sn_app_cs_social_social_profile.contact.date_format | Format in which to display dates to contacts. | Valid values:  
  • dd/mm/yyyy  
  • dd-mm-yyyy  
  • dd.mm.yyyy |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.contact.default_perspective</td>
<td>Sys_id of the default perspective for the contact. Located in the Menu List [sys_perspective] table. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.department</td>
<td>Sys_id of the department associated with the contact. Located in the Department [cmn_department] table. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.edu_status</td>
<td>Education status of the associated contact. Data type: String Maximum length: 40 Default: faculty</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.email</td>
<td>Contact email address. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.employee_number</td>
<td>Contact employee number. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.enable_multifactor_authn</td>
<td>Flag that indicates whether multifactor authorization is required for the contact to log in to the service portal. Possible values:</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.failed_attempts</td>
<td>Number of failed log in attempts.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.first_name</td>
<td>Contact first name.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.gender</td>
<td>Contact gender.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.geolocation_tracked</td>
<td>Flag that indicates whether contact location is obtained through geotracking. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Contact location obtained through geotracking</td>
</tr>
<tr>
<td></td>
<td>• false: Contact location not obtained through geotracking</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.home_phone</td>
<td>Contact home phone number.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.internal_integration_user</td>
<td>Flag that indicates whether contact is an internal integration user.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.contact.introduction | Possible values:  
• true: Internal integration user  
• false: Other type of user  
Data type: Boolean  
Default: false |
| sn_app_cs_social_social_profile.contact.last_login | Introduction  
Data type: String  
Maximum length: 40 |
| sn_app_cs_social_social_profile.contact.last_login_device | Date on which the consumer logged into the system.  
Data type: String (Date) |
| sn_app_cs_social_social_profile.contact.last_login_time | Device the consumer used the last time they logged in to the system.  
Data type: String  
Maximum length: 40 |
| sn_app_cs_social_social_profile.contact.last_name | Date and time the contact logged in to the system.  
Data type: String (Date/t ime) |
| sn_app_cs_social_social_profile.contact.last_position_update | Contact last name.  
Data type: String  
Maximum length: 50 |
| sn_app_cs_social_social_profile.contact.last_position_device | Date and time the last position was updated.  
Data type: String (Date/t ime) |
| sn_app_cs_social_social_profile.contact.latitude | Latitude coordinate of contact.  
Data type: Number (Floating point)  
Maximum length: 40 |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.contact.ldap_server</td>
<td>Sys_id of the LDAP server used by the contact to last log in to the system; LDAP Server [ldap_server_config] table. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.location</td>
<td>Sys_id of the record that describes the location of the contact; Location [cmn_location] table. Data type: String</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.contact.locked_out | Flag that indicates if the contact is locked-out. Possible values:  
  • true: Contact locked-out  
  • false: Contact not locked-out Data type: Boolean Default: false |
<p>| sn_app_cs_social_social_profile.contact.longitude | Longitude coordinate of the contact. Data type: Number (Floating point) Maximum length: 40 |
| sn_app_cs_social_social_profile.contact.manager | Sys_id of the record that describes the direct supervisor of the contact; User [sys_user] table. Data type: String |
| sn_app_cs_social_social_profile.contact.middle_name | Contact middle name. Data type: Number (Floating point) Maximum length: 50 |
| sn_app_cs_social_social_profile.contact.mobile_phone | Contact mobile phone number. Data type: String |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.contact.name</td>
<td>Contact full name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.notification</td>
<td>Indicates whether the contact should receive notifications.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Disabled</td>
</tr>
<tr>
<td></td>
<td>• 2: Enabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 2</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.on_schedule</td>
<td>Indicates the timeliness of dispatched service personnel.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• Ahead: Ahead of schedule</td>
</tr>
<tr>
<td></td>
<td>• behind_less30: Behind schedule, but less than 30 minutes.</td>
</tr>
<tr>
<td></td>
<td>• behind_30to60: Behind schedule between 30 and 60 minutes.</td>
</tr>
<tr>
<td></td>
<td>• behind_more60: Behind schedule more than 60 minutes.</td>
</tr>
<tr>
<td></td>
<td>• on_time: On schedule</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.phone</td>
<td>Contact business phone number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.photo</td>
<td>Photo image of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: Image</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.preferred_language</td>
<td>Country code of the contact's primary language.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 3</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.roles</td>
<td>List of user roles associated with the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.schedule</td>
<td>Sys_id of the record that describes the work schedule for the associated contact.</td>
</tr>
<tr>
<td></td>
<td>Schedule [cmn_schedule]</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.source</td>
<td>Source of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.state</td>
<td>State in which the contact resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.street</td>
<td>Contact street address.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_class_name</td>
<td>Table that contains the contact record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 80</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_created_by</td>
<td>User that originally created the associated contact record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_created_on</td>
<td>Data and time the associated contact was originally created. Data type: String (Date/time)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_domain</td>
<td>ServiceNow instance of the associated contact record. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_domain_path</td>
<td>Contact record domain path. Data type: String Maximum length: 255 Default: / (global)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_id</td>
<td>Unique identifier for the associated contact record. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_mod_count</td>
<td>Number of times that the associated contact record has been modified. Data type: Number (Integer)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_tags</td>
<td>Contact system tags.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_updated_by</td>
<td>User that last updated the associated contact information. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_updated_on</td>
<td>Data and time the associated contact information was updated. Data type: String (Date/time)</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.contact.time_format | Format in which to display time. Valid values:  
  - hh.mm.ss a: hh:mm:ss (12 hour)  
  - hh:mm:ss a: hh:mm:ss (12 hour) |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sn_app_cs_social_social_profile.contact.time_sheet_policy</code></td>
<td>Sys_id of the record that contains the time sheet policy for the associated contact; Time Sheet Policy [time_sheet_policy] table. Data type: String</td>
</tr>
<tr>
<td><code>sn_app_cs_social_social_profile.contact.time_zone</code></td>
<td>Time zone in which the contact resides, such as Canada/Central or US/Eastern. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td><code>sn_app_cs_social_social_profile.contact.title</code></td>
<td>Contact business title such as Manager, Software Developer, or Contractor. Data type: String Maximum length: 60</td>
</tr>
<tr>
<td><code>sn_app_cs_social_social_profile.contact.user_name</code></td>
<td>Contact user ID. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td><code>sn_app_cs_social_social_profile.contact.vip</code></td>
<td>Flag that indicates whether the associated contact has VIP status. Possible values: true: VIP false: Not VIP Data type: Boolean</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.contact.web_service_access_only | Flag that indicates whether contact can only access through the web. Possible values: 
  - true: Web access only
  - false: Access through all available methods
  Data type: Boolean
  Default: false |
| sn_app_cs_social_social_profile.contact.zip | Contact zip code.
  Data type: String
  Maximum length: 40 |
| sn_app_cs_social_social_profile.created_on | Date and time the associated social media profile was initially created. |
| sn_app_cs_social_social_profile.profile | Social profile.
  Data type: String
  Maximum length: 255 |
| sn_app_cs_social_social_profile.profile_url | URL to use to access the social media profile.
  Data type: String
  Maximum length: 1,024 |
| sn_app_cs_social_social_profile.social_id | Unique social media account provider identifier for the associated social media account.
  Data type: String
  Maximum length: 100 |
| sn_app_cs_social_social_profile.source | Source of the social profile.
  Data type: String |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.sys_created_by</td>
<td>User that initially created the social media profile. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_created_on</td>
<td>Date and time the social profile was initially created. Data type: String (Date)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_id</td>
<td>Unique identifier for the social media profile. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_mod_count</td>
<td>Number of times that information was modified for the associated social media profile. Data type: Number (Integer)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_tags</td>
<td>Profile system tags.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_updated_by</td>
<td>User that initially created the social media profile. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_updated_on</td>
<td>User that initially created the social media profile. Data type: String Maximum length: 40</td>
</tr>
</tbody>
</table>
| state | Current state of the case. Possible values:  
  - 1: New  
  - 3: Closed  
  - 6: Resolved  
  - 10: Open  
  - 18: Awaiting Info |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subcategory</td>
<td>Case subcategory. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Question</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer) Default: 1</td>
</tr>
<tr>
<td>support_manager</td>
<td>Sys_id of the CSM manager assigned to the case. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sync_driver</td>
<td>Flag that indicates whether there is driver synchronization. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Driver is synchronized</td>
</tr>
<tr>
<td></td>
<td>• false: Driver is not synchronized</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>sys_class_name</td>
<td>Table that contains the case record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 80</td>
</tr>
<tr>
<td>sys_created_by</td>
<td>Person that initially opened the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sys_created_on</td>
<td>Date and time when the case was initially created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_domain</td>
<td>Domain associated with the case.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sys_domain_path</td>
<td>Domain path. Data type: String, Maximum length: 255, Default: /</td>
</tr>
<tr>
<td>sys_id</td>
<td>Unique identifier for the case. Data type: String, Maximum length: 32</td>
</tr>
<tr>
<td>sys_mod_count</td>
<td>Number of updates to the case since it was initially created. Data type: Number (Integer)</td>
</tr>
<tr>
<td>sys_tags</td>
<td>System tags.</td>
</tr>
<tr>
<td>sys_updated_by</td>
<td>Person that last updated the case. Data type: String, Maximum length: 40</td>
</tr>
<tr>
<td>sys_updated_on</td>
<td>Date and time when the case was last updated. Data type: String</td>
</tr>
<tr>
<td>time_worked</td>
<td>Total amount of time worked on the case. Data type: String</td>
</tr>
<tr>
<td>upon_approval</td>
<td>Action to take if the case is approved. Possible values: do_nothing, proceed</td>
</tr>
</tbody>
</table>

Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>upon_reject</td>
<td>Action to take if the case is rejected. Possible values: • cancel • goto Data type: String Maximum length: 40 Default: cancel</td>
</tr>
<tr>
<td>urgency</td>
<td>Urgency of the case. Possible values: • 1: High • 2: Medium • 3: Low Data type: Number (Integer) Default: 3</td>
</tr>
<tr>
<td>user_input</td>
<td>Additional user input. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>variable_pool</td>
<td>Grouping of variables.</td>
</tr>
<tr>
<td>variables</td>
<td>Name-value pairs of variables associated with the case. Data type: String Maximum length: 40</td>
</tr>
</tbody>
</table>
| watch_list    | List of sys_ids of the users who receive notifications about the case when additional comments are added or if the state of the case is changed to Resolved.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>or Closed. Located in the [sys_user] table.</td>
<td>Data type: Array</td>
</tr>
<tr>
<td>wf_activity</td>
<td>Sys_id of the workflow activity record associated with the case. Located in the Workflow Activity [wf_activity] table.</td>
</tr>
<tr>
<td>work_end</td>
<td>Date and time work ended on the case.</td>
</tr>
<tr>
<td>work_notes</td>
<td>Information about how to resolve the case, or steps taken to resolve it.</td>
</tr>
<tr>
<td>Maximum length: 4,000</td>
<td></td>
</tr>
<tr>
<td>work_notes_list</td>
<td>List of sys_ids of the internal users who receive notifications about this case when work notes are added. Located in the [sys_user] table.</td>
</tr>
<tr>
<td>work_start</td>
<td>Date and time that work started on the case.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl 
  "https://instance.servicenow.com/api/sn_customerservice/case/ba325dbe64f01300964fae39b050e9a6" \ 
--request GET \ 
--header "Accept:application/json" \ 
--user "username":"password"

{ 
 "result": {
```
"close_notes": "",
"sys_id": "ba325dbe64f01300964fae39b050e9a6",
"contact_type": "web",
"resolution_code": "",
"probable_cause": "",
"urgency": "3",
"company": "",
"major_case_state": "accepted",
"consumer": "",
"activity_due": "",
"comments": "",
"approval": "not requested",
"due_date": "",
"sys_mod_count": "2",
"sys_tags": "",
"active_account_escalation": "",
"location": "",
"category": "1",
"account": ""
}

Example: Python request
This request retrieves a case, with sysparm_display_value set to "all" to retrieve display and literal field values.

```python
# Install requests package for python
import requests

# Set the API endpoint URL for the request
url = 'https://instance.servicenow.com/api/sn_customerservice/v1/case/09ba659a0b412300c9c86c9037673af4'

# Set the credentials
user = 'username'
pwd = 'password'

# Set the query parameters
params = {'sysparm_display_value': 'all'}

# Set the HTTP headers
headers = {'Accept': 'application/xml'}
```
# Issue the HTTP request
response = requests.get(url, auth=(user, pwd), params=params, headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
    response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

Because `sysparm_display_value` is set to "all", the response includes literal values and display values for each field. Removing this parameter or setting its value to "false" retrieves literal database values.

```xml
<response>
  <result>
    <parent>
      <display_value/>
      <name>parent</name>
      <label>Parent</label>
      <type>reference</type>
      <value>null</value>
    </parent>
    <caused_by>
      <display_value/>
      <name>caused_by</name>
      <label>Caused by Change</label>
      <type>reference</type>
      <value>null</value>
    </caused_by>
    <watch_list>
      <display_value/>
      <name>watch_list</name>
      <label>Watch list</label>
      <type>glide_list</type>
      <value>null</value>
    </watch_list>
    <active_escalation>
      <display_value/>
      <name>active_escalation</name>
      <label>Active escalation</label>
  </result>
</response>
```
<type>reference</type>
  <value>null</value>
</active_escalation>
<upon_reject>
  <display_value>Cancel all future Tasks</display_value>
  <name>upon_reject</name>
  <label>Upon reject</label>
  <type>string</type>
  <value>cancel</value>
</upon_reject>
<sys_updated_on>
  <display_value>2020-03-17 08:51:51</display_value>
  <name>sys_updated_on</name>
  <label>Updated</label>
  <type>glide_date_time</type>
  <value>2020-03-17 15:51:51</value>
</sys_updated_on>
<support_manager>
  <display_value/>
  <name>support_manager</name>
  <label>Support Manager</label>
  <type>reference</type>
  <value>null</value>
</support_manager>
<approval_history>
  <display_value/>
  <name>approval_history</name>
  <label>Approval history</label>
  <type>journal</type>
  <value/>
</approval_history>
<skills>
  <display_value/>
  <name>skills</name>
  <label>Skills</label>
  <type>glide_list</type>
  <value>null</value>
</skills>
<number>
  <display_value>CS0009003</display_value>
  <name>number</name>
  <label>Number</label>
  <type>string</type>
  <value>CS0009003</value>
</number>
<display_value/>
<name>order</name>
<label>Order</label>
$type>integer</type>
$value>null</value>
</order>

<assigned_on>
<display_value/>
<name>assigned_on</name>
<label>Assigned on</label>
$type>glide_date_time</type>
$value/>
</assigned_on>

<cmdb_ci>
<display_value/>
<name>cmdb_ci</name>
<label>Configuration item</label>
$type>reference</type>
$value>null</value>
</cmdb_ci>

<delivery_plan>
<display_value/>
<name>delivery_plan</name>
<label>Delivery plan</label>
$type>reference</type>
$value>null</value>
</delivery_plan>

<impact>
<display_value>3 - Low</display_value>
<name>impact</name>
<label>Impact</label>
$type>integer</type>
$value>3</value>
</impact>

<contract>
<display_value/>
<name>contract</name>
<label>Contract</label>
$type>reference</type>
$value>null</value>
</contract>

<active>
<display_value>true</display_value>
<name>active</name>

<value/>
</first_response_time>
<business_duration>
<display_value/>
<name>business_duration</name>
<label>Business duration</label>
<type>glide_duration</type>
[value/>
</business_duration>
<group_list>
<display_value/>
<name>group_list</name>
<label>Group list</label>
<type>glide_list</type>
[value>null</value>
</group_list>
<child_case_creation_progress>
<display_value>false</display_value>
<name>child_case_creation_progress</name>
<label>Child Case creation progress</label>
<type>boolean</type>
[value>null</value>
</child_case_creation_progress>
<sync_driver>
<display_value>false</display_value>
<name>sync_driver</name>
<label>Sync Driver</label>
<type>boolean</type>
[value>0</value>
</sync_driver>
<internal_contact>
<display_value/>
<name>internal_contact</name>
<label>Internal Contact</label>
<type>reference</type>
[value>null</value>
</internal_contact>
<approval_set>
<display_value/>
<name>approval_set</name>
<label>Approval set</label>
<type>glide_date_time</type>
[value/>
</approval_set>
<wf_activity>
  <display_value/>
  <name>wf_activity</name>
  <label>Workflow activity</label>
  <type>reference</type>
  <value>null</value>
</wf_activity>

<needs_attention>
  <display_value>false</display_value>
  <name>needs_attention</name>
  <label>Needs attention</label>
  <type>boolean</type>
  <value>0</value>
</needs_attention>

<universal_request>
  <display_value/>
  <name>universal_request</name>
  <label>Universal Request</label>
  <type>reference</type>
  <value>null</value>
</universal_request>

<short_description>
  <display_value>I have a pending change request</display_value>
  <name>short_description</name>
  <label>Short description</label>
  <type>string</type>
  <value>I have a pending change request</value>
</short_description>

<correlation_display>
  <display_value/>
  <name>correlation_display</name>
  <label>Correlation display</label>
  <type>string</type>
  <value>null</value>
</correlation_display>

<delivery_task>
  <display_value/>
  <name>delivery_task</name>
  <label>Delivery task</label>
  <type>reference</type>
  <value>null</value>
</delivery_task>

<work_start>
  <display_value/>
</work_start>
<type>sys_class_name</type>
<value>sn_customerservice_case</value></sys_class_name>
<service_offering>
<display_value/>
<name>service_offering</name>
<label>Service offering</label>
<type>reference</type>
<value>null</value></service_offering>
<closed_by>
<display_value/>
<name>closed_by</name>
<label>Closed by</label>
<type>reference</type>
<value>null</value></closed_by>
<follow_up>
<display_value/>
<name>follow_up</name>
<label>Follow up</label>
<type>glide_date_time</type>
<value/>
</follow_up>
<contact_local_time>
<display_value/>
<name>contact_local_time</name>
<label>Contact Local Time</label>
<type>string</type>
<value>null</value></contact_local_time>
<sn_app_cs_social_social_profile>null</sn_app_cs_social_social_profile>
<reassignment_count>
<display_value>1</display_value>
<name>reassignment_count</name>
<label>Reassignment count</label>
<type>integer</type>
<value>1</value></reassignment_count>
<contact_time_zone>
<display_value>null</display_value>
<name>contact_time_zone</name>
<label>Contact time zone</label>
<type>choice</type>
<type>boolean</type>
<value>0</value>
</web_service_access_only>
<notification>
<display_value>Enable</display_value>
<name>notification</name>
<label>Notification</label>
<type>integer</type>
<value>2</value>
</notification>
<sys_updated_by>
<display_value>admin</display_value>
<name>sys_updated_by</name>
<label>Updated by</label>
<type>string</type>
<value>admin</value>
</sys_updated_by>
<enable_multifactor_authn>
<display_value>false</display_value>
<name>enable_multifactor_authn</name>
<label>Enable Multifactor Authentication</label>
<type>boolean</type>
<value>0</value>
</enable_multifactor_authn>
<sys_created_on>
<display_value>2018-08-15 13:45:45</display_value>
<name>sys_created_on</name>
<label>Created</label>
<type>glide_date_time</type>
<value>2018-08-15 20:45:45</value>
</sys_created_on>
<sys_domain>
<display_value>global</display_value>
<name>sys_domain</name>
<label>Domain</label>
<type>domain_id</type>
<value>global</value>
</sys_domain>
<agent_status>
<display_value>null</display_value>
<name>agent_status</name>
<label>Work agent status</label>
<type>choice</type>
<value>null</value>
</agent_status>
</agent_status>
</state>
<display_value/>
</state>
<name>state</name>
<label>State / Province</label>
<type>string</type>
</state>
</vip>
<display_value>false</display_value>
</vip>
<name>vip</name>
<label>VIP</label>
<type>boolean</type>
</vip>
</sys_created_by>
<display_value>admin</display_value>
</sys_created_by>
<name>sys_created_by</name>
<label>Created by</label>
<type>string</type>
</sys_created_by>
<longitude>
<display_value/>
</longitude>
<name>longitude</name>
<label>Longitude</label>
<type>float</type>
</longitude>
</zip>
<display_value/>
</zip>
<name>zip</name>
<label>Zip / Postal code</label>
<type>string</type>
</zip>
</home_phone>
<display_value/>
</home_phone>
<name>home_phone</name>
<label>Home phone</label>
<type>ph_number</type>
</home_phone>
</time_format>
<display_value>null</display_value>
<name>time_format</name>
<label>Time format</label>
?type>string</type>
<value>null</value>
</time_format>

<last_login>
<display_value/>
<name>last_login</name>
<label>Last login</label>
?type>glide_date</type>
<value/>
</last_login>

<default_perspective>
<display_value/>
<name>default_perspective</name>
<label>Default perspective</label>
?type>reference</type>
<value>null</value>
</default_perspective>
<geolocation_tracked>
<display_value>false</display_value>
<name>geolocation_tracked</name>
<label>Geolocation tracked</label>
?type>boolean</type>
<value>0</value>
</geolocation_tracked>

<active>
<display_value>true</display_value>
<name>active</name>
<label>Active</label>
?type>boolean</type>
<value>1</value>
</active>
<time_sheet_policy>
<display_value/>
<name>time_sheet_policy</name>
<label>Time sheet policy</label>
?type>reference</type>
<value>null</value>
</time_sheet_policy>
<sys_domain_path>
<display_value>/</display_value>
<name>sys_domain_path</name>
</sys_domain_path>
<label>Domain Path</label>
<type>domain_path</type>
[value>/</value>
</sys_domain_path>

<transaction_log>
<display_value/>
<name>transaction_log</name>
<label>Transaction log</label>
?type>string</type>
[value>null</value>
</transaction_log>

<phone>
<display_value/>
<name>phone</name>
<label>Business phone</label>
?type>ph_number</type>
[value>null</value>
</phone>

<cost_center>
<display_value/>
<name>cost_center</name>
<label>Cost center</label>
?type>reference</type>
[value>null</value>
</cost_center>

<name>
<display_value>Amy Chen</display_value>
<name>name</name>
<label>Name</label>
?type>string</type>
[value>Amy Chen</value>
</name>

<employee_number>
<display_value/>
<name>employee_number</name>
<label>Employee number</label>
?type>string</type>
[value>null</value>
</employee_number>

<gender>
<display_value>null</display_value>
<name>gender</name>
<label>Gender</label>
?type>string</type>
<value>null</value>
</gender>
<city>
  <display_value/>
  <name>city</name>
  <label>City</label>
  <type>string</type>
  <value>null</value>
</city>
<hr_integration_source>
  <display_value/>
  <name>hr_integration_source</name>
  <label>HR Integration source</label>
  <type>reference</type>
  <value>null</value>
</hr_integration_source>
<user_name>
  <display_value/>
  <name>user_name</name>
  <label>User ID</label>
  <type>string</type>
  <value>null</value>
</user_name>
<failed_attempts>
  <display_value/>
  <name>failed_attempts</name>
  <label>Failed login attempts</label>
  <type>integer</type>
  <value>null</value>
</failed_attempts>
<edu_status>
  <display_value>Faculty</display_value>
  <name>edu_status</name>
  <label>EDU Status</label>
  <type>string</type>
  <value>faculty</value>
</edu_status>
<latitude>
  <display_value/>
  <name>latitude</name>
  <label>Latitude</label>
  <type>float</type>
  <value>null</value>
</latitude>
<roles>
  <display_value/>
  <name>roles</name>
  <label>Roles</label>
  <type>user_roles</type>
  <value>null</value>
</roles>

<title>
  <display_value/>
  <name>title</name>
  <label>Title</label>
  <type>string</type>
  <value>null</value>
</title>

<sys_class_name>
  <display_value>Contact</display_value>
  <name>sys_class_name</name>
  <label>Class</label>
  <type>sys_class_name</type>
  <value>customer_contact</value>
</sys_class_name>

<sys_id>
  <displayValue>b88d14843bc02300bfe04d72f3efc4cd</displayValue>
  <label>Sys ID</label>
  <type>GUID</type>
  <value>b88d14843bc02300bfe04d72f3efc4cd</value>
</sys_id>

<internal_integration_user>
  <display_value>false</display_value>
  <name>internal_integration_user</name>
  <label>Internal Integration User</label>
  <type>boolean</type>
  <value>0</value>
</internal_integration_user>

<ldap_server>
  <display_value/>
  <name>ldap_server</name>
  <label>LDAP server</label>
  <type>reference</type>
  <value>null</value>
</ldap_server>

<mobile_phone>
  <display_value/>
  <name>mobile_phone</name>
</mobile_phone>
<label>Mobile phone</label>
<type>ph_number</type>
,value>null</value>
</mobile_phone>

<street>
<display_value/>
<name>street</name>
<label>Street</label>
?type>multi_two_lines</type>
,value>null</value>
</street>

<company>
<display_value>Advances Super Computing</display_value>
<name>company</name>
<label>Company</label>
?type>reference</type>
,value>dc5dd0843bc02300bfe04d72f3efc41e</value>
</company>

<department>
<display_value/>
<name>department</name>
<label>Department</label>
?type>reference</type>
,value>null</value>
</department>

<first_name>
<display_value>Amy</display_value>
<name>first_name</name>
<label>First name</label>
?type>string</type>
,value>Amy</value>
</first_name>

<preferred_language>
<display_value>null</display_value>
<name>preferred_language</name>
<label>Language</label>
?type>string</type>
,value>null</value>
</preferred_language>

<introduction>
<display_value>null</display_value>
<name>introduction</name>
<label>Prefix</label>
?type>string</type>
<value>null</value>
</introduction>
<email>
<display_value>amy.chen@mailinator.com</display_value>
<name>email</name>
<label>Email</label>
<type>email</type>
<value>amy.chen@mailinator.com</value>
</email>
<manager>
<display_value/>
<name>manager</name>
<label>Manager</label>
<type>reference</type>
<value>null</value>
</manager>
<locked_out>
<display_value>false</display_value>
<name>locked_out</name>
<label>Locked out</label>
<type>boolean</type>
<value>0</value>
</locked_out>
<sys_mod_count>
<display_value>0</display_value>
<name>sys_mod_count</name>
<label>Updates</label>
<type>integer</type>
<value>0</value>
</sys_mod_count>
<last_name>
<display_value>Chen</display_value>
<name>last_name</name>
<label>Last name</label>
<type>string</type>
<value>Chen</value>
</last_name>
<photo>
<display_value/>
<name>photo</name>
<label>Photo</label>
<type>user_image</type>
<value>null</value>
</photo>
<correlation_id>
    <label>Correlation ID</label>
    <type>string</type>
    <value>null</value>
</correlation_id>

date_format
    <display_value>null</display_value>
    <label>Date format</label>
    <type>string</type>
    <value>null</value>
</date_format>

<location>
    <display_value/>
    <label>Location</label>
    <type>reference</type>
    <value>null</value>
</location>

<account>
    <display_value>Advances Super Computing</display_value>
    <label>Account</label>
    <type>reference</type>
    <value>dc5dd0843bc02300bfe04d72f3efc41e</value>
</account>

<sys_domain>
    <display_value>global</display_value>
    <label>Domain</label>
    <type>domain_id</type>
    <value>global</value>
</sys_domain>

<route_reason>
    <display_value/>
    <label>Route reason</label>
    <type>integer</type>
    <value>null</value>
</route_reason>

closed_at
    <display_value/>
    <label>Closed at</label>
</closed_at>
<label>Closed</label>
<type>glide_date_time</type>
<value/>
</closed_at>
<follow_the_sun>
<display_value>false</display_value>
<name>follow_the_sun</name>
<label>Follow the sun</label>
<type>boolean</type>
<value>0</value>
</follow_the_sun>
<entitlement>
<display_value/>
<name>entitlement</name>
<label>Entitlement</label>
<type>reference</type>
<value>null</value>
</entitlement>
<business_service>
<display_value/>
<name>business_service</name>
<label>Service</label>
<type>reference</type>
<value>null</value>
</business_service>
<business_impact>
<display_value/>
<name>business_impact</name>
<label>Business impact</label>
<type>string</type>
<value>null</value>
</business_impact>
<time_worked>
<display_value/>
<name>time_worked</name>
<label>Time worked</label>
<type>timer</type>
<value/>
</time_worked>
<expected_start>
<display_value/>
<name>expected_start</name>
<label>Expected start</label>
<type>glide_date_time</type>
<value/>
</expected_start>
<value/>
</expected_start>

<opened_at>
<display_value>2018-10-11 19:11:28</display_value>

<name>opened_at</name>
<label>Opened</label>
<type>glide_date_time

RIEND

</opened_at>

<work_end>
<display_value/>

<name>work_end</name>
<label>Actual end</label>
<type>glide_date_time

<value/>

</work_end>

<resolved_at>
<display_value/>

<name>resolved_at</name>
<label>Resolved</label>
<type>glide_date_time

<value/>

</resolved_at>

<case_report>
<display_value/>

<name>case_report</name>
<label>Case Report</label>
<type>reference

<value>b7baa59a0b412300c9c86c9037673a64</value>

</case_report>

<subcategory>
<display_value>Question</display_value>

<name>subcategory</name>
<label>Subcategory</label>
<type>integer

<value>0</value>

</subcategory>

<work_notes>
<display_value>2018-10-11 12:12:43 - System Administrator (Work notes)

Change

[code]@&gt;CHG0040007;&lt;/a&gt;

;[/code] has been associated with the Case

</display_value>

<name>work_notes</name>
<label>Work notes</label>

</work_notes>
</close_notes>
<auto_close>
  <display_value>false</display_value>
  <name>auto_close</name>
  <label>Auto close</label>
  <type>boolean</type>
  <value>null</value>
</auto_close>
=sys_id>
  <displayValue>09ba659a0b412300c9c86c9037673af4</displayValue>
  <label>Sys ID</label>
  <type>GUID</type>
  <value>09ba659a0b412300c9c86c9037673af4</value>
</sys_id>
<sn_esign_esignature_configuration>
  <display_value/>
  <name>sn_esign_esignature_configuration</name>
  <label>E-signature template</label>
  <type>reference</type>
  <value>null</value>
</sn_esign_esignature_configuration>
<contact_type>
  <display_value>Web</display_value>
  <name>contact_type</name>
  <label>Channel</label>
  <type>string</type>
  <value>web</value>
</contact_type>
<resolution_code>
  <display_value>null</display_value>
  <name>resolution_code</name>
  <label>Resolution code</label>
  <type>string</type>
  <value>null</value>
</resolution_code>
<probable_cause>
  <display_value/>
  <name>probable_cause</name>
  <label>Probable cause</label>
  <type>string</type>
  <value>null</value>
</probable_cause>
<urgency>
  <display_value>3 - Low</display_value>
</urgency>
<name>urgency</name>
<label>Urgency</label>
?type>integer</type>
<value>3</value>
</urgency>

<company>
&display_value/>
<label>Company</label>
?type>reference</type>
<value>null</value>
</company>

<major_case_state>
&display_value>null</display_value>
<label>Major case state</label>
?type>string</type>
<value>null</value>
</major_case_state>

<consumer>null</consumer>

<activity_due>
&display_value>UNKNOWN</display_value>
<label>Activity due</label>
?type>due_date</type>
</activity_due>

'action_status>
&display_value>Blocked internally</display_value>
<label>Action status</label>
?type>integer</type>
<value>1</value>
</'
<label>Approval</label>
    <type>string</type>
    <value>not requested</value>
</approval>
<due_date>
    <display_value/>
    <name>due_date</name>
    <label>Due date</label>
    <type>glide_date_time</type>
    <value/>
</due_date>
<sys_mod_count>
    <display_value>9</display_value>
    <name>sys_mod_count</name>
    <label>Updates</label>
    <type>integer</type>
    <value>9</value>
</sys_mod_count>
<sys_tags>
    <display_value/>
    <name>sys_tags</name>
    <label>Tags</label>
    <type>related_tags</type>
    <value/>
</sys_tags>
<active_account_escalation>
    <display_value/>
    <name>active_account_escalation</name>
    <label>Active account escalation</label>
    <type>reference</type>
    <value>null</value>
</active_account_escalation>
<location>
    <display_value/>
    <name>location</name>
    <label>Location</label>
    <type>reference</type>
    <value>null</value>
</location>
<category>
    <display_value>Issue</display_value>
    <name>category</name>
    <label>Category</label>
    <type>integer</type>
<value>1</value>
</category>

<incident>
<display_value/>
<name>incident</name>
<label>Incident</label>
?type>reference</type>
<value>null</value>
</incident>

<account>
<banner_image_light>
<display_value/>
<name>banner_image_light</name>
<label>UI16 Banner Image</label>
?type>user_image</type>
<value>null</value>
</banner_image_light>

<country>
<display_value>USA</display_value>
<name>country</name>
<label>Country</label>
?type>string</type>
<value>USA</value>
</country>

<parent>
<display_value/>
<name>parent</name>
<label>Parent</label>
?type>reference</type>
<value>null</value>
</parent>

<notes>
<display_value/>
<name>notes</name>
<label>Notes</label>
?type>string</type>
<value>null</value>
</notes>

<stock_symbol>
<display_value/>
<name>stock_symbol</name>
<label>Stock symbol</label>
?type>string</type>
<value>null</value>
</stock_symbol>
<display_value/>
<name>fiscal_year</name>
<label>Fiscal year</label>
&type>glide_date</type>
</fiscal_year>
<sys_created_on>
<display_value>2018-08-15 13:44:55</display_value>
</sys_created_on>
<contact>
<display_value/>
</contact>
<stock_price>
<display_value/>
</stock_price>
<state>
<display_value>CA</display_value>
</state>
<banner_image>
<display_value/>
</banner_image>
<sys_created_by>
<display_value>admin</display_value>
</sys_created_by>
<label>Created by</label>
<type>string</type>
[value>admin</value>
</sys_created_by>

<longitude>
<display_value>-121.9529992</display_value>
<label>Longitude</label>
<type>float</type>
[value>-121.9529992</value>
</longitude>

<zip>
<display_value>95050</display_value>
<label>Zip / Postal code</label>
<type>string</type>
[value>95050</value>
</zip>

<profits>
<display_value>$0.00</display_value>
<label>Profits</label>
<type>currency</type>
[value>0</value>
</profits>

<phone>
<display_value>(408) 839-2810</display_value>
<label>Phone</label>
<type>ph_number</type>
[value>(408) 839-2810</value>
</phone>

<fax_phone>
<display_value/>
<label>Fax phone</label>
<type>ph_number</type>
[value>null</value>
</fax_phone>

<name>
<display_value>Advances Super Computing</display_value>
</name>

<value>Advances Super Computing</value>
</name>
<banner_text>
<display_value/>
<name>banner_text</name>
<label>Banner text</label>
<type>string</type>
<value>null</value>
</banner_text>
<account_code>
<display_value>~~~~30</display_value>
<name>account_code</name>
<label>Account Code</label>
<type>string</type>
<value>~~~~30</value>
</account_code>
<primary>
<display_value>false</display_value>
<name>primary</name>
<label>Primary</label>
<type>boolean</type>
<value>0</value>
</primary>
<city>
<display_value>Santa Clara</display_value>
<name>city</name>
<label>City</label>
<type>string</type>
<value>Santa Clara</value>
</city>
<latitude>
<display_value>37.3539663</display_value>
<name>latitude</name>
<label>Latitude</label>
<type>float</type>
<value>37.3539663</value>
</latitude>
<sys_class_name>
<display_value>Account</display_value>
<name>sys_class_name</name>
<label>Class</label>
<type>sys_class_name</type>
<value>customer_account</value>
</sys_class_name>

5434

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


<manufacturer>
  <display_value>false</display_value>
  <name>manufacturer</name>
  <label>Manufacturer</label>
  <type>boolean</type>
  <value>0</value>
</manufacturer>

<account_parent>
  <display_value/>
  <name>account_parent</name>
  <label>Parent Account</label>
  <type>reference</type>
  <value>null</value>
</account_parent>

/sys_id>
  <displayValue>dc5dd0843bc02300bfe04d72f3efc41e</displayValue>
  <label>Sys ID</label>
  <type>GUID</type>
  <value>dc5dd0843bc02300bfe04d72f3efc41e</value>
</sys_id>

<market_cap>
  <display_value>$0.00</display_value>
  <name>market_cap</name>
  <label>Market cap</label>
  <type>currency</type>
  <value>0</value>
</market_cap>

<num_employees>
  <display_value/>
  <name>num_employees</name>
  <label>Number of employees</label>
  <type>integer</type>
  <value>null</value>
</num_employees>

<rank_tier>
  <display_value>null</display_value>
  <name>rank_tier</name>
  <label>Rank tier</label>
  <type>string</type>
  <value>null</value>
</rank_tier>

<street>
  <display_value/>
  <name>street</name>
</street>
<value>0</value>
</revenue_per_year>

<publicly_traded>
  <display_value>false</display_value>
  <name>publicly_traded</name>
  <label>Publicly traded</label>
  <type>boolean</type>
  <value>0</value>
</publicly_traded>

<sys_mod_count>
  <display_value>2</display_value>
  <name>sys_mod_count</name>
  <label>Updates</label>
  <type>integer</type>
  <value>2</value>
</sys_mod_count>

<sys_tags>
  <display_value/>
  <name>sys_tags</name>
  <label>Tags</label>
  <type>related_tags</type>
  <value/>
</sys_tags>

<partner>
  <display_value>false</display_value>
  <name>partner</name>
  <label>Partner</label>
  <type>boolean</type>
  <value>0</value>
</partner>

<registration_code>
  <display_value/>
  <name>registration_code</name>
  <label>Registration Code</label>
  <type>string</type>
  <value>null</value>
</registration_code>

<vendor_manager>
  <display_value/>
  <name>vendor_manager</name>
  <label>Vendor manager</label>
  <type>glide_list</type>
  <value>null</value>
</vendor_manager>
<account_path>
    <display_value>~30</display_value>
    <name>account_path</name>
    <label>Account Path</label>
    <type>string</type>
    <value>~30</value>
</account_path>

<primary_contact>
    <display_value>Amy Chen</display_value>
    <name>primary_contact</name>
    <label>Primary Contact</label>
    <type>reference</type>
    <value>b88d14843bc02300bfe04d72f3efc4cd</value>
</primary_contact>

<customer>
    <display_value>true</display_value>
    <name>customer</name>
    <label>Customer</label>
    <type>boolean</type>
    <value>1</value>
</customer>

</account>
</result>
</response>

**Case - GET /sn_customerservice/case**

Retrieves a specified set of Customer Service Management (CSM) cases.

To use this endpoint, users must have the csm_ws_integration, sn_customerservice.customer, or sn_customerservice.consumer role.

**Note:** You can reference all sysparm query parameters using either their full name or their shortened name (without the `sysparm_` prefix). For example, for `sysparm_limit` you can also use `limit`.

**URL format**

Versioned URL: /api/sn_customerservice/{api_version}/case

Default URL: /api/sn_customerservice/case
**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, <code>v1</code> or <code>v2</code>. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_display_value</td>
<td>Data retrieval operation for reference and choice fields. Based on this value, retrieves the display value and/or the actual value from the database.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Returns the display values for all fields</td>
</tr>
<tr>
<td></td>
<td>• false: Returns the actual values from the database</td>
</tr>
<tr>
<td></td>
<td>• all: Returns both actual and display values</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> There is no preferred method for setting this parameter. However, specifying the display value may cause performance issues since it is not reading directly from the database and may include referencing other fields and records. For more information on display values and actual values, see Table API FAQs (KB0534905).</td>
</tr>
<tr>
<td>sysparm_fields</td>
<td>Comma-separated list of fields to return in the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. Unusually large <code>sysparm_limit</code> values can impact system performance. For requests that exceed this number of records, use the <code>sysparm_offset</code> parameter to paginate record retrieval.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, <code>sysparm_offset</code> is set to &quot;0&quot;. To simply page through all available records, use <code>sysparm_offset=sysparm_offset+sysparm_limit</code>, until you reach the end of all records. Do not pass a negative number in the <code>sysparm_offset</code> parameter. Data type: Number Default: 0</td>
</tr>
<tr>
<td>sysparm_query</td>
<td>Encoded query used to filter the result set. For example: <code>{sysparm_query=caller_id=javascript:gs.getUserID()^active=true}</code> The encoded query supports order by. To sort responses based on certain fields, use the ORDERBY and ORDERBYDESC clauses in <code>sysparm_query</code>. For example, <code>sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory</code> filters all active records and orders the results in ascending order by number first, and then in descending order by category. If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property <code>glide.invalid_query.returns_no_rows</code>. Set this property to true to return no rows on an invalid query. Data type: String</td>
</tr>
<tr>
<td>sysparm_reference_fields</td>
<td>Whether or not to retrieve fields from reference tables such as account, contact, consumer, and sn_app_cs_social_social_profile. By default, these fields are returned if the user has read access to them, but this adds...</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>performance overhead. When retrieving a large number of case records, this overhead can be eliminated by setting sysparm_reference_fields=false.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>Default: true</td>
<td></td>
</tr>
</tbody>
</table>

Note: When using this endpoint, you can omit the sysparm_ prefix for this query parameter and specify it as reference_fields.

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link</td>
<td>REST response data can be split into multiple result sets. Where applicable, the response header contains different links for the first set, previous set, next set, and the last set of records. For example:</td>
</tr>
</tbody>
</table>
Response headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>https://&lt;instance name&gt;.service-now.com/api/now/table/cmdb_ci?sysparm_offset=40&amp;sysparm_limit=10000&gt;;rel=&quot;next&quot;</td>
<td></td>
</tr>
<tr>
<td>https://&lt;instance name&gt;.service-now.com/api/now/table/cmdb_ci?sysparm_offset=40&amp;sysparm_limit=10000&gt;;rel=&quot;prev&quot;</td>
<td></td>
</tr>
<tr>
<td>https://&lt;instance name&gt;.service-now.com/api/now/table/cmdb_ci?sysparm_offset=0&amp;sysparm_limit=10000&gt;;rel=&quot;first&quot;</td>
<td></td>
</tr>
<tr>
<td>https://&lt;instance name&gt;.service-now.com/api/now/table/cmdb_ci?sysparm_offset=2780&amp;sysparm_limit=10000&gt;;rel=&quot;last&quot;</td>
<td></td>
</tr>
<tr>
<td>X-Total-Count</td>
<td>Response header showing the total number of records matching the request when the sysparm_limit or sysparm_offset query parameters are specified.</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account</td>
<td>Sys_id of the account record associated with the case. Located in the Account [customer_account] table. Data type: String</td>
</tr>
<tr>
<td>account</td>
<td>Account details.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Data type</strong>: Object</td>
<td></td>
</tr>
</tbody>
</table>

```json
"account": {
  "account_code": "String",
  "account_parent": "String",
  "account_path": "String",
  "active_escalation": "String",
  "apple_icon": Image,
  "banner_image": Image,
  "banner_image_light": Image,
  "banner_text": "String",
  "city": "String",
  "contact": "String",
  "country": "String",
  "customer": "String",
  "discount": "String",
  "fax_phone": "String",
  "fiscal_year": "String",
  "lat_long_error": "String",
  "latitude": "String",
  "longitude": "String",
  "market_cap": "String",
  "name": "String",
  "notes": "String",
  "num_employees": Number,
  "number": "String",
  "parent": "String",
  "partner": "String",
  "phone": "String",
  "primary": Boolean,
  "primary_contact": "String",
  "profits": Number,
  "publicly_traded": Boolean,
  "rank_tier": "String",
  "registration_code": "String",
  "revenue_per_year": Number,
  "state": "String",
  "stock_price": "String",
  "stock_symbol": "String",
  "street": "String",
  "sys_class_name": "String",
  "sys_created_by": "String",
  "sys_created_on": "String"
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account.account_code</td>
<td>Unique combination of values that an application uses to identify budget forecasts and budget plans. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>account.account_parent</td>
<td>Sys_id of the parent account of this account. Located in the Account [customer_account] table. Data type: String</td>
</tr>
<tr>
<td>account.account_path</td>
<td>Path from the parent to child accounts in the account hierarchy. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>account.active_escalation</td>
<td>Sys_id of the active escalation associated with the account. Located in the Escalation [sn_customerservice_escalation] table. Data type: String</td>
</tr>
<tr>
<td>account.apple_icon</td>
<td>Icon for iPhone home page bookmarks. Data type: Image</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>account.banner_image</td>
<td>Banner image that appears on the customer portal.</td>
</tr>
<tr>
<td></td>
<td>Data type: Image</td>
</tr>
<tr>
<td>account.banner_image_light</td>
<td>Small banner image.</td>
</tr>
<tr>
<td></td>
<td>Data type: Image</td>
</tr>
<tr>
<td>account.banner_text</td>
<td>Banner text that appears on the customer portal.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>account.city</td>
<td>City in which the company that is associated with this account resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>account.contact</td>
<td>Sys_id of a contact record associated with this account.</td>
</tr>
<tr>
<td></td>
<td>Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>account.country</td>
<td>Country in which the company that is associated with this account resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: USA</td>
</tr>
<tr>
<td>account.customer</td>
<td>Flag that indicates whether the account is a customer account or a partner account.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Customer account</td>
</tr>
<tr>
<td></td>
<td>• false: Partner account</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account.discount</td>
<td>Discount given to the account on purchases.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 15</td>
</tr>
<tr>
<td>account.fax_phone</td>
<td>Primary fax phone number for the company associated with this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.fiscal_year</td>
<td>Fiscal year for the company associated with this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>account.lat_long_error</td>
<td>Difference in the actual location as compared to latitude and longitude information.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 1,000</td>
</tr>
<tr>
<td>account.latitude</td>
<td>Latitude of the company associated with this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (floating point number)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.longitude</td>
<td>Longitude of the company associated with this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (floating point number)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.manufacturer</td>
<td>Flag that indicates whether the company associated with this account is a manufacturer</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
|                         | • true: Manufactures goods  
• false: Does not manufacture goods  
Data type: Boolean  
Default: false                                                                                                                                 |
| account.market_cap      | Market value of the associated company's publicly traded stock shares.  
Data type: Number (currency)  
Maximum length: 20                                                                                                                             |
| account.name            | Name of the company associated with this account.  
Data type: String  
Maximum length: 80                                                                                                                           |
| account.notes           | Additional information about the company.  
Data type: String  
Maximum length: 4,000                                                                                                                         |
| account.num_employees   | Number of people employed by the company.  
Data type: Number (Integer)  
Maximum length: 40                                                                                                                         |
| account.number          | Number that identifies this account.  
Data type: String  
Maximum length: 40                                                                                                                         |
| account.parent          | Sys_id of the parent account of this account. Located in the Company [core_company] table.  
Data type: String                                                                                                                            |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| account.partner          | Flag that indicates whether the account is a partner account or a customer account.  
                          | Possible values:  
                          | • true: Partner account  
                          | • false: Customer account  
                          | Data type: Boolean  
                          | Default: false                                                   |
| account.phone            | Primary phone number for the company.  
                          | Data type: String                                                                                                                        |
| account.primary          | Flag that indicates whether this is a primary account.  
                          | Possible values:  
                          | • true: Primary account  
                          | • false: Secondary account  
                          | Data type: Boolean  
                          | Default: false                                                   |
| account.primary_contact  | Sys_id of the primary contact for the account. Located in the Contact [customer_contact] table.  
                          | Data type: String                                                                                                                        |
| account.profits          | Profit information entered for this account.  
                          | Data type: Number (Currency)  
                          | Maximum length: 40                                                          |
| account.publicly_traded  | Flag that indicates whether the company associated with the account is publicly traded on the stock exchange. |

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Publicly traded</td>
</tr>
<tr>
<td></td>
<td>• false: Private company</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>account.rank_tier</td>
<td>Type of account.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• blacklist</td>
</tr>
<tr>
<td></td>
<td>• strategic</td>
</tr>
<tr>
<td></td>
<td>• tactical</td>
</tr>
<tr>
<td></td>
<td>• valued</td>
</tr>
<tr>
<td></td>
<td>• other</td>
</tr>
<tr>
<td>Data type: String</td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.registration_code</td>
<td>Unique code that customers use when requesting a login on the customer portal. This code provides a method for validating the customer on the company before granting access.</td>
</tr>
<tr>
<td>Data type: String</td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.revenue_per_year</td>
<td>Revenue produced by the company associated with this account.</td>
</tr>
<tr>
<td>Data type: Number (Currency)</td>
<td>Maximum length: 20</td>
</tr>
<tr>
<td>account.state</td>
<td>State in which the company resides.</td>
</tr>
<tr>
<td>Data type: String</td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>account.stock_price</td>
<td>Price of the company stock.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.stock_symbol</td>
<td>Stock symbol of the company.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.street</td>
<td>Street address of the company.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>account.sys_class_name</td>
<td>Table that contains the associated account record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>account.sys_created_by</td>
<td>User that originally created the account.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.sys_created_on</td>
<td>Date and time that the account was originally created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>account.sys_id</td>
<td>Sys_id for the account</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>account.sys_mod_count</td>
<td>Number of times the account information has been updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td>account.sys_updated_by</td>
<td>User that last modified the account information.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.sys_updated_on</td>
<td>Date and time the account information was last updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account.theme</td>
<td>Sys_id of the customer portal theme used by this account. Located in the Theme [sys_ui_theme] table. Data type: String</td>
</tr>
</tbody>
</table>
| account.vendor        | Flag that indicates whether the company associated with this account is a vendor. Possible values: 
  • true: Vendor 
  • false: Not a vendor Data type: Boolean Default: false |
<p>| account.vendor_manager | List of sys_ids of the vendor managers for the account. Located in the User [sys_user] table. Data type: String |
| account.vendor_type   | List of sys_ids of the type of vendor such as, applications, hardware, services, or software. Located in the Vendor Type [vendor_type] table. Data type: String |
| account.website       | URL of the website for the company. Data type: String Maximum length: 1,024 |
| account.zip           | Zip code of the company. Data type: String Maximum length: 40 |
| active                | Flag that indicates whether the case is open and active. |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Case is active</td>
</tr>
<tr>
<td></td>
<td>• false: Case is closed</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td>Default: true</td>
</tr>
<tr>
<td>active_account_escalation</td>
<td>Sys_id of the active account escalation record associated with the case. Located in the Escalation [sn_customerservice_escalation] table.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>active_escalation</td>
<td>Sys_id of the active account escalation record associated with the case. Located in the Escalation [sn_customerservice_escalation] table.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>activity_due</td>
<td>Date for which the associated case is expected to be completed.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>additional_assignee_list</td>
<td>List of the sys_ids of the persons (other than primary assignee) that have been assigned to the account. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td>Maximum: 4,000</td>
</tr>
<tr>
<td>approval</td>
<td>String that describes the approval required.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>• approved</td>
</tr>
<tr>
<td></td>
<td>• cancelled</td>
</tr>
<tr>
<td></td>
<td>• duplicate</td>
</tr>
<tr>
<td></td>
<td>• not_required</td>
</tr>
<tr>
<td></td>
<td>• not_requested</td>
</tr>
<tr>
<td></td>
<td>• rejected</td>
</tr>
<tr>
<td></td>
<td>• requested</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum characters:</td>
</tr>
<tr>
<td></td>
<td>Default: not requested</td>
</tr>
<tr>
<td>approval_history</td>
<td>List of all approvals associated with the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum characters:</td>
</tr>
<tr>
<td>approval_set</td>
<td>Date and time that the associated action was approved.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>asset</td>
<td>Sys_id of the asset record associated with the case.</td>
</tr>
<tr>
<td></td>
<td>Located in the Asset [alm_asset] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>assigned_to</td>
<td>Sys_id of the person assigned to the case.</td>
</tr>
<tr>
<td></td>
<td>Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>assignment_group</td>
<td>Sys_id of the customer service agent group assigned to the case.</td>
</tr>
<tr>
<td></td>
<td>Located in the Group [sys_user_group] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>business_duration</td>
<td>Length in calendar work hours, work days, and work weeks that it took to complete the case. Data type: String</td>
</tr>
<tr>
<td>business_impact</td>
<td>Impact of the issue on the associated customer. Data type: String Maximum characters:</td>
</tr>
<tr>
<td>business_service</td>
<td>Sys_id of the service record associated with the case. Located in the Service [cmdb_ci_service] table. Data type: String</td>
</tr>
<tr>
<td>case</td>
<td>Case short description and case number. Data type: String Maximum length: 300</td>
</tr>
<tr>
<td>category</td>
<td>Case category. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Question</td>
</tr>
<tr>
<td></td>
<td>• 1: Issue</td>
</tr>
<tr>
<td></td>
<td>• 2: Feature Data type: Number (Integer) Default: 1</td>
</tr>
<tr>
<td>cause</td>
<td>Details about the cause of the problem.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>caused_by</td>
<td>Sys_id of the change request that caused the case to be created. Located in the Change Request [change_request] table.</td>
</tr>
<tr>
<td>change</td>
<td>Sys_id of the change request that caused the case to be created. Located in the Change Request [change_request] table.</td>
</tr>
<tr>
<td>child_case_creation_progress</td>
<td>Flag that indicates whether the case is a child case that was created from a major case. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Child case created from a major case</td>
</tr>
<tr>
<td></td>
<td>• false: Not a child case</td>
</tr>
<tr>
<td>closed_at</td>
<td>Date and time that the case was closed.</td>
</tr>
<tr>
<td>closed_by</td>
<td>Sys_id of the user that closed the case. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td>close_notes</td>
<td>Notes made when the case was closed.</td>
</tr>
<tr>
<td>cmdb_ci</td>
<td>Sys_id of the configuration item associated with the case.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>comments_and_work_notes</td>
<td>Comments and work notes entered for the case. Data type: String. Maximum length: 4,000.</td>
</tr>
<tr>
<td>consumer</td>
<td>Business-to-consumer cases only. Sys_id of the person to contact with regards to this case. Located in the Consumer [csm_consumer] table.</td>
</tr>
<tr>
<td>consumer</td>
<td>Array of parameters for business-to-consumer cases.</td>
</tr>
<tr>
<td>consumer.active</td>
<td>Flag that indicates whether the consumer is active. Possible values: true: Consumer active; false: Consumer de-activated. Data type: Boolean. Default: true.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consumer.business_phone</td>
<td>Business phone number of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.city</td>
<td>City in which the consumer resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>consumer.country</td>
<td>Country in which the consumer resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.date_format</td>
<td>Format in which to display dates.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• dd-mm-yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd/mm/yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd.mm.yyyy</td>
</tr>
<tr>
<td></td>
<td>• mm-dd-yyyy</td>
</tr>
<tr>
<td></td>
<td>• yyyy-mm-dd</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system date format)</td>
</tr>
<tr>
<td>consumer.email</td>
<td>Email address of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>consumer.fax</td>
<td>Fax number of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.first_name</td>
<td>Consumer first name.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consumer.gender</td>
<td>Gender of the consumer.</td>
</tr>
<tr>
<td>consumer.home_phone</td>
<td>Home phone number of the consumer.</td>
</tr>
<tr>
<td>consumer.household</td>
<td>Sys_id of the record that describes the household characteristics. Located Household [csm_household] table.</td>
</tr>
<tr>
<td>consumer.last_name</td>
<td>Consumer last name.</td>
</tr>
<tr>
<td>consumer.middle_name</td>
<td>Consumer middle name.</td>
</tr>
<tr>
<td>consumer.mobile_phone</td>
<td>Consumer mobile phone.</td>
</tr>
<tr>
<td>consumer.name</td>
<td>Consumer full name: first_name +middle_name+last_name.</td>
</tr>
<tr>
<td>consumer.notes</td>
<td>Notes on consumer.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consumer.notification</td>
<td>Indicates whether the consumer should receive notifications.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Disabled</td>
</tr>
<tr>
<td></td>
<td>• 2: Enabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Integer</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: 2</td>
</tr>
<tr>
<td>consumer.number</td>
<td>Unique number associated with the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.photo</td>
<td>Photo of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: Image</td>
</tr>
<tr>
<td>consumer.preferred_language</td>
<td>Consumer primary language.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 3</td>
</tr>
<tr>
<td>consumer.prefix</td>
<td>Consumer name prefix such as, Dr., Mr., Mrs., or Ms.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.primary</td>
<td>Flag that indicates whether the consumer is the primary consumer.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Primary consumer</td>
</tr>
<tr>
<td></td>
<td>• false: Not primary consumer</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consumer.state</td>
<td>State in which the consumer resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>consumer.street</td>
<td>Consumer street address.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>consumer.suffix</td>
<td>Consumer name suffix such as Jr., Sr., or II.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>consumer.sys_created_by</td>
<td>User that created the consumer record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.sys_created_on</td>
<td>Date and time the consumer record was originally created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>consumer.sys_domain</td>
<td>ServiceNow domain in which the consumer information resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>consumer.sys_id</td>
<td>Unique identifier for the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>consumer.sys_mod_count</td>
<td>Number of times that the associated consumer information has been modified.</td>
</tr>
<tr>
<td></td>
<td>Data type: Integer</td>
</tr>
<tr>
<td>consumer.sys_updated_by</td>
<td>User that last updated the consumer information.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consumer.sys_updated_on</td>
<td>Date and time when the consumer information was last updated. Data type: String</td>
</tr>
<tr>
<td>consumer.time_format</td>
<td>Format in which to display time. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• hh.mm.ss a: hh.mm.ss</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>• HH.mm.ss: hh.mm.ss</td>
</tr>
<tr>
<td></td>
<td>• HH:mm:ss: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system time format)</td>
</tr>
<tr>
<td>consumer.time_zone</td>
<td>Consumer time zone, such as Canada/Central or US/Eastern. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.title</td>
<td>Consumer business title such as Manager, Software Developer, or Contractor. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 60</td>
</tr>
<tr>
<td>consumer.user</td>
<td>Sys_id of the consumer. Located in the Consumer User [csm_consumer_user] table. Data type: String</td>
</tr>
<tr>
<td>consumer.zip</td>
<td>Consumer zip code. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact</td>
<td>Business-to-business based cases only. Sys_id of the person to contact regarding this case. Located in the Customer [customer_contact] table. Data type: String</td>
</tr>
<tr>
<td>contact</td>
<td>Array of contact parameters for business-to-business cases.</td>
</tr>
<tr>
<td>contact.account</td>
<td>Sys_id of the account record to which the contact is associated; Account [customer_account] table. Data type: String</td>
</tr>
<tr>
<td>contact.active</td>
<td>Flag that indicates whether the contact is active within the system. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Contact is active</td>
</tr>
<tr>
<td></td>
<td>• false: Contact is inactive</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td>contact.agent_status</td>
<td>Status of the agent. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Off work</td>
</tr>
<tr>
<td></td>
<td>• On break</td>
</tr>
<tr>
<td></td>
<td>• On route</td>
</tr>
<tr>
<td></td>
<td>• On site</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.building</td>
<td>Sys_id of the record that describes the building</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>the contact resides: Building</td>
<td>Describes the building where the contact resides. Data type: String</td>
</tr>
<tr>
<td>contact.calendar_integration</td>
<td>Calendar application that the contact uses. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Outlook</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 1</td>
</tr>
<tr>
<td>contact.city</td>
<td>City in which the contact resides. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.company</td>
<td>Sys_id of the company with which the contact is associated. Data type: String</td>
</tr>
<tr>
<td>contact.cost_center</td>
<td>Sys_id of the cost center associated with the contact. Data type: String</td>
</tr>
<tr>
<td>contact.country</td>
<td>Country code of the country in which the contact resides. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 3</td>
</tr>
<tr>
<td>contact.date_format</td>
<td>Format in which to display dates to contacts. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• dd/mm/yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd-mm-yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd.mm.yyyy</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>mm-dd-yyyy</td>
<td>Data type: String</td>
</tr>
<tr>
<td>yyy-mm-dd</td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system date format)</td>
</tr>
<tr>
<td>contact.default_perspective</td>
<td>Sys_id of the default perspective for the contact. Located in the Menu List [sys_perspective] table.</td>
</tr>
<tr>
<td>contact.department</td>
<td>Sys_id of the department associated with the contact. Located in the Department [cmn_department] table.</td>
</tr>
<tr>
<td>contact.edu_status</td>
<td>Education status of the associated contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: faculty</td>
</tr>
<tr>
<td>contact.email</td>
<td>Contact email address.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>contact.employee_number</td>
<td>Contact employee number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>contact.enable_multifactor_authn</td>
<td>Flag that indicates whether multifactor authorization is required for the contact to log in to the service portal.</td>
</tr>
<tr>
<td></td>
<td>Possible values: multipurpose, faculty, student, staff, guest, alumni, alumni, education, other, unknown</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>• true: Multifactor auth</td>
<td>enabled</td>
</tr>
<tr>
<td>• false: Multifactor auth</td>
<td>disabled</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td>Default: false</td>
</tr>
<tr>
<td>contact.failed_attempts</td>
<td>Number of failed log in attempts</td>
</tr>
<tr>
<td>Data type: Number (Integer)</td>
<td></td>
</tr>
<tr>
<td>contact.first_name</td>
<td>Contact first name.</td>
</tr>
<tr>
<td>Data type: String</td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>contact.gender</td>
<td>Contact gender.</td>
</tr>
<tr>
<td>Data type: String</td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.geolocation_tracked</td>
<td>Flag that indicates whether the contact location is obtained through geotracking.</td>
</tr>
<tr>
<td>Possible values:</td>
<td>• true: Contact location obtained through geotracking</td>
</tr>
<tr>
<td></td>
<td>• false: Contact location not obtained through geotracking</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td>Default value: false</td>
</tr>
<tr>
<td>contact.home_phone</td>
<td>Contact home phone number.</td>
</tr>
<tr>
<td>Data type: String</td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.internal_integration_user</td>
<td>Flag that indicates whether the contact is an internal integration user.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.introduction</td>
<td>Introduction</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.last_login</td>
<td>Date on which the contact last logged into the system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date)</td>
</tr>
<tr>
<td>contact.last_login_device</td>
<td>Device the consumer used the last time they logged in to the system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.last_login_time</td>
<td>Date and time the contact last logged in to the system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date/time)</td>
</tr>
<tr>
<td>contact.last_name</td>
<td>Contact last name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>contact.last_position_update</td>
<td>Date and time the last position was updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date/time)</td>
</tr>
<tr>
<td>contact.latitude</td>
<td>Latitude coordinate of contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Floating point)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.ldap_server</td>
<td>Sys_id of the LDAP server used by the contact to last log in to the system; LDAP Server [ldap_server_config] table. Data type: String</td>
</tr>
<tr>
<td>contact.location</td>
<td>Sys_id of the record that describes the location of the contact; Location [cmn_location] table. Data type: String</td>
</tr>
<tr>
<td>contact.locked_out</td>
<td>Flag that indicates if the contact is locked-out. Possible values: • true: Contact locked-out • false: Contact not locked-out Data type: Boolean Default: false</td>
</tr>
<tr>
<td>contact.longitude</td>
<td>Longitude coordinate of the contact. Data type: Number (Floating point) Maximum length: 40</td>
</tr>
<tr>
<td>contact.manager</td>
<td>Sys_id of the record that describes the direct supervisor of the contact; User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>contact.middle_name</td>
<td>Contact middle name. Data type: Number (Floating point) Maximum length: 50</td>
</tr>
<tr>
<td>contact.mobile_phone</td>
<td>Contact mobile phone number. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.name</td>
<td>Contact full name. Contact phone number. Data type: String. Maximum length: 151.</td>
</tr>
</tbody>
</table>
| contact.notification | Indicates whether the contact should receive notifications. Valid values:  
• 1: Disabled
• 2: Enabled  
Data type: Number (Integer). Default: 2 |
| contact.on_schedule | Indicates the timeliness of dispatched service personnel. Valid values:  
• Ahead: Ahead of schedule  
• behind_less30: Behind schedule, but less than 30 minutes.  
• behind_30to60: Behind schedule between 30 and 60 minutes.  
• behind_more60: Behind schedule more than 60 minutes.  
• on_time: On schedule  
Data type: String. Maximum length: 40 |
<p>| contact.phone    | Contact business phone. Data type: String. Maximum length: 40 |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contact.photo</td>
<td>Photo image of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: Image</td>
</tr>
<tr>
<td>contact.preferred_language</td>
<td>Country code of the contact's primary language.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 3</td>
</tr>
<tr>
<td>contact.roles</td>
<td>List of user roles associated with the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.schedule</td>
<td>Sys_id of the record that describes the work schedule for the associated contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>contact.source</td>
<td>Source of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>contact.state</td>
<td>State in which the contact resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.street</td>
<td>Contact street address</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>contact.sys_class_name</td>
<td>Table that contains the contact record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 80</td>
</tr>
<tr>
<td>contact.sys_created_by</td>
<td>User that originally created the associated contact record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.sys_created_on</td>
<td>Maximum length: 40 Data and time the associated contact was originally created. Data type: String (Date/time)</td>
</tr>
<tr>
<td>contact.sys_domain</td>
<td>ServiceNow instance of the associated contact record. Data type: String</td>
</tr>
<tr>
<td>contact.sys_domain_path</td>
<td>Contact record domain path. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255 Default: / (global)</td>
</tr>
<tr>
<td>contact.sys_id</td>
<td>Unique identifier for the associated contact record. Data type: String</td>
</tr>
<tr>
<td>contact.sys_mod_count</td>
<td>Number of times that the associated contact record has been modified. Data type: Number (Integer)</td>
</tr>
<tr>
<td>contact.sys_tags</td>
<td>System tags for the contact. Data type: String</td>
</tr>
<tr>
<td>contact.sys_updated_by</td>
<td>User that last updated the associated contact information. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.sys_updated_on</td>
<td>Data and time the associated contact information was last updated. Data type: String (Date/time)</td>
</tr>
<tr>
<td>contact.time_format</td>
<td>Format in which to display time. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.time_sheet_policy</td>
<td>Sys_id of the record that contains the time sheet policy for the associated contact; Time Sheet Policy [time_sheet_policy]</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.web_service_access_only</td>
<td>Flag that indicates whether the contact can only access through the web. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Web access only</td>
</tr>
<tr>
<td></td>
<td>• false: Access through all available methods</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>contact.zip</td>
<td>Contact zip code.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact_local_time</td>
<td>Contact local time.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 70</td>
</tr>
<tr>
<td>contact_time_zone</td>
<td>Time zone of the contact associated with the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact_type</td>
<td>Method in which the case was initially reported. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• chat</td>
</tr>
<tr>
<td></td>
<td>• email</td>
</tr>
<tr>
<td></td>
<td>• phone</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contract</td>
<td>Sys_id of the contract associated with the case. Located in the Contract [ast_contract] table. This contract contains information about the type of support that is provided to the company associated to the case. A contract can include a company and contact, specific assets that are covered, and multiple service entitlements and SLAs. Data type: String</td>
</tr>
<tr>
<td>correlation_display</td>
<td>Correlation display. Data type: String Maximum length: 100</td>
</tr>
<tr>
<td>correlation_id</td>
<td>Correlation identifier. Data type: String Maximum length: 100</td>
</tr>
<tr>
<td>delivery_plan</td>
<td>Sys_id of the parent execution plan for this case. Located in the Execution Plan [sc_cat_item_delivery_plan] table. Data type: String</td>
</tr>
<tr>
<td>delivery_task</td>
<td>Sys_id of the execution task. Located in the Execution Task [sc_cat_item_delivery_task] table. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>description</td>
<td>Detailed description of the problem associated with the case. Data type: String. Maximum length: 4,000.</td>
</tr>
<tr>
<td>due_date</td>
<td>Date that the case is due to be closed. Data type: String.</td>
</tr>
<tr>
<td>entitlement</td>
<td>Sys_id of the entitlement record associated with the case. Located in the Entitlement [service_entitlement] table. Entitlements define the type of support that a customer receives as well as the communication channels an entitlement can be associated with a product, an asset, an account, or a contract. Data type: String.</td>
</tr>
</tbody>
</table>
| escalation | Current escalation level. Possible values:  
  - 0: Normal  
  - 1: Moderate  
  - 2: High  
  - 3: Overdue  
  Data type: Number (Integer). Default: 0. |
<p>| expected_start | Date and time when work is scheduled to begin on the case. Data type: String. |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>first_response_time</td>
<td>Date and time when the first action was taken on the case. Data type: String</td>
</tr>
</tbody>
</table>
| follow_the_sun    | Flag that indicates whether the case should be handed-off for global follow-up. If a customer enters additional comments on a Priority 1 - Critical or a Priority 2 - High case, or if the case is escalated, it is automatically set to true. Possible values:  
  - true: Case should be handed-off for global follow-up  
  - false: Case should not be handed-off for global follow-up  
  Data type: Boolean  
  Default: false                                                                 |
| follow_up         | Date and time of the next follow-up action.  
  Data type: String                                                                                                                       |
| group_list        | List of sys_ids of the group records associated with the case. Located in the [sys_user_group] table. Data type: Array                                                                                          |
| impact            | Impact on customer. Possible values:  
  - 1: High  
  - 2: Medium  
  - 3: Low  
  Data type: Number (Integer)                                                                                                              |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>knowledge</td>
<td>Flag that indicates if there is a knowledge base article available for the specified issue. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Knowledge base article is available for this issue</td>
</tr>
<tr>
<td></td>
<td>• false: Knowledge base article is not available for this issue</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>accepted</td>
<td>• accepted: Initial state when a manager creates a major case or when a manager promotes a candidate case.</td>
</tr>
<tr>
<td>canceled</td>
<td>• canceled: Case is canceled.</td>
</tr>
<tr>
<td>proposed</td>
<td>• proposed: Initial state when an agent or manager creates or proposes a candidate case.</td>
</tr>
<tr>
<td>rejected</td>
<td>• rejected: Manager rejects a candidate case.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Maximum length: 40</td>
<td></td>
</tr>
<tr>
<td>needs_attention</td>
<td>Flag that indicates whether the case needs attention.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Case needs additional attention</td>
</tr>
<tr>
<td></td>
<td>• false: Case does not need additional attention</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>Default: false</td>
<td></td>
</tr>
<tr>
<td>notes_to_comments</td>
<td>Flag that indicates whether to add the resolution notes to the comments.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Resolutions notes, when added, are also added to the comments</td>
</tr>
<tr>
<td></td>
<td>• false: Resolution notes in comments are not required</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| notify        | Method to use to notify contact/consumer. Possible values:  
• 1: Do not notify  
• 2: Send email  
• 3: Telephone  
Data type: Number (Integer)  
Default: 1 |
| number        | Case number.  
Data type: String  
Maximum length: 40 |
| opened_at     | Date and time that the case was opened.  
Data type: String |
| opened_by     | Sys_id of the person that initially opened the case. Located in the User [sys_user] table.  
Data type: String |
| order         | Order of the case.  
Data type: Number (Integer) |
| parent        | Sys_id of the parent case to which this case (child) is associated. Located in the Task [task] table.  
Data type: String |
| partner       | Sys_id of the partner associated with the case. Located in the Account [customer_account] table.  
Data type: String |
| partner_contact | Sys_id of the partner contact associated with the case.  
Data type: String |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>priority</td>
<td>Priority of the case. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Critical</td>
</tr>
<tr>
<td></td>
<td>• 2: High</td>
</tr>
<tr>
<td></td>
<td>• 3: Moderate</td>
</tr>
<tr>
<td></td>
<td>• 4: Low</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer) Default: 4</td>
</tr>
<tr>
<td>probable_cause</td>
<td>Possible cause of the issue associated with the case.</td>
</tr>
<tr>
<td>problem</td>
<td>Sys_id of the issue that the customer is encountering.</td>
</tr>
<tr>
<td>product</td>
<td>Sys_id of the product model of the asset associated with the issue.</td>
</tr>
<tr>
<td>reassignment_count</td>
<td>Number of times that the case was reassigned to a person who is responsible for moving the case forward.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>recipient_list</td>
<td>Sys_id of the record that contains the list of recipients for communications about this case. Located in the Recipients List [sn_publications_recipients_list] table. Data type: String</td>
</tr>
<tr>
<td>rejection_goto</td>
<td>Sys_id of the task to execute if the case is rejected. Located in the Task [task] table. Data type: String</td>
</tr>
<tr>
<td>resolution_code</td>
<td>Resolution state for the case, such as &quot;Solved - Fixed by Support/Guidance provided&quot;. Maximum length: 40</td>
</tr>
<tr>
<td>resolved_at</td>
<td>Date and time that the case was resolved. Data type: String</td>
</tr>
<tr>
<td>resolved_by</td>
<td>Sys_id of the person that resolved the case. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>result</td>
<td>Array of objects in which each object describes a single value for the requested field. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"result": [
  {
    "label": "String",
    "value": "String"
  }
]
```

result.label | Display value for the field.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.value</td>
<td>Field value. Data type: String</td>
</tr>
<tr>
<td>short_description</td>
<td>Concise description of the case. Data type: String, Maximum length: 160</td>
</tr>
<tr>
<td>skills</td>
<td>List of the unique identifiers (sys_id) of the skills needed to complete the case. Located in the Skill [cmn_skill] table. Data type: String</td>
</tr>
<tr>
<td>sla_due</td>
<td>Date/time at which the case must be closed based on the associated service level agreement. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile</td>
<td>Array of objects that contain details for a specific social media profile, such as Facebook or YouTube. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"sn_app_cs_social_social_profile": [
  {
    "account": "String",
    "channel": [Array],
    "contact": {Object},
    "consumer": {Object},
    "created_on": "String",
    "profile": "String",
    "profile_url": "String",
    "social_id": "String",
    "source": "String",
    "sys_created_by": "String",
    "sys_created_on": "String",
    "sys_id": "String",
    "sys_mod_count": Number
  }
]```
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.account</td>
<td>Unique identifier of the record containing account information for the social media profile. Located in the Account [customer_account] table. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel</td>
<td>Details on the associated social media profile channel. Data type: Object</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.name</td>
<td>Name of the social media channel. Data type: String Maximum length: 100</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_class_name</td>
<td>Table that contains the social media channel record Data type: String Maximum length: 80</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.channel.sys_created_by | Person that created the social media channel.  
Data type: String  
Maximum length: 40 |
| sn_app_cs_social_social_profile.channel.sys_created_on | Date and time the social media profile was created.  
Data type: Date/time  
Maximum length: 40 |
| sn_app_cs_social_social_profile.channel.sys_id | Unique identifier of the associated social media channel.  
Data type: String |
| sn_app_cs_social_social_profile.channel.sys_mod_count | Number of times that information was modified for the associated social media profile channel.  
Data type: Integer |
| sn_app_cs_social_social_profile.channel.sys_name | System name of channel.  
Data type: String  
Maximum length: 255 |
| sn_app_cs_social_social_profile.channel.sys_package | Unique identifier of the record that contains information about the package associated with the profile; Package [sys_package] table.  
Data type: Reference |
| sn_app_cs_social_social_profile.channel.sys_policy | System protection policy.  
Possible values:  
• protected  
• read  
Data type: String  
Maximum length: 40 |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_scope</td>
<td>Unique identifier of the record that contains information about the scope of the social profile. Application [sys_scope] table. Data type: Reference</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_tags</td>
<td>System tags associated with the channel. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_update_name</td>
<td>Name of the person that last updated the social media profile channel. Data type: String Maximum length: 250</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_updated_by</td>
<td>User that last updated the social media profile channel. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_updated_on</td>
<td>Date and time the social media profile channel was last updated. Data type: Date/time Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer</td>
<td>For business-to-consumer cases. Details about the consumer associated with the case. Data type: Object</td>
</tr>
</tbody>
</table>

```json
"consumer": {
  "active": Boolean,
  "business_phone": "String",
  "city": "String",
  "country": "String",
  "date_format": "String",
  "email": "String",
  "fax": "String",
  "first_name": "String",
  "gender": "String",
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;home_phone&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;household&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;last_name&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;middle_name&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;mobile_phone&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;name&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;notes&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;notification&quot;: Number,</td>
<td></td>
</tr>
<tr>
<td>&quot;number&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;photo&quot;: Image,</td>
<td></td>
</tr>
<tr>
<td>&quot;preferred_language&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;prefix&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;primary&quot;: Boolean,</td>
<td></td>
</tr>
<tr>
<td>&quot;state&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;street&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;suffix&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_created_by&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_created_on&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_domain&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_mod_count&quot;: Number,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_tags&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_updated_by&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_updated_on&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;time_format&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;time_zone&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;title&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;user&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;zip&quot;: &quot;String&quot;</td>
<td></td>
</tr>
</tbody>
</table>

sn_app_cs_social_social_profile.consumer.active

Flag that indicates whether the consumer is active.

Possible values:

- true: Consumer active
- false: Consumer de-activated

Data type: Boolean

Default: true
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sn_app_cs_social_social_profile.consumer.business_phone</code></td>
<td>Business phone number of the consumer. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td><code>sn_app_cs_social_social_profile.consumer.city</code></td>
<td>City in which the consumer resides. Data type: String Maximum length: 100</td>
</tr>
<tr>
<td><code>sn_app_cs_social_social_profile.consumer.country</code></td>
<td>Country in which the consumer resides. Data type: String Maximum length: 40</td>
</tr>
</tbody>
</table>
| `sn_app_cs_social_social_profile.consumer.date_format` | Format in which to display dates. Valid values:  
- `dd-mm-yyyy`  
- `dd/mm/yyyy`  
- `dd.mm.yyyy`  
- `mm-dd-yyyy`  
- `yyyy-mm-dd`  
Data type: String Maximum length: 40 Default: blank (system date format) |
<p>| <code>sn_app_cs_social_social_profile.consumer.email</code> | Email address of the consumer. Data type: String Maximum length: 100 |
| <code>sn_app_cs_social_social_profile.consumer.fax</code> | Fax number of the consumer. Data type: String Maximum length: 40 |
| <code>sn_app_cs_social_social_profile.consumer.first_name</code> | Consumer first name. |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.consumer.gender</td>
<td>Gender of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.home_phone</td>
<td>Home phone number of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.household</td>
<td>Sys_id of the record that describes the household characteristics.</td>
</tr>
<tr>
<td></td>
<td>Located in the Household [csm_household] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.last_name</td>
<td>Consumer last name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.middle_name</td>
<td>Consumer middle name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.mobile_phone</td>
<td>Consumer mobile phone.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.name</td>
<td>Consumer full name; first_name +middle_name+last_name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 152</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.notes</td>
<td>Notes on consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.consumer.notification | Indicates whether the consumer should receive notifications. Valid values:  
- 1: Disabled  
- 2: Enabled  
Data type: Integer  
Maximum length: 40  
Default: 2 |
| sn_app_cs_social_social_profile.consumer.number | Unique number associated with the consumer.  
Data type: String  
Maximum length: 40 |
| sn_app_cs_social_social_profile.consumer.photo | Photo of the consumer.  
Data type: Image |
| sn_app_cs_social_social_profile.consumer.preferred_language | Consumer primary language.  
Data type: String  
Maximum length: 3 |
| sn_app_cs_social_social_profile.consumer.prefix | Consumer name prefix such as, Dr., Mr., Mrs., or Ms.  
Data type: String  
Maximum length: 40 |
| sn_app_cs_social_social_profile.consumer.primary | Flag that indicates whether the consumer is the primary consumer. Possible values:  
- true: Primary consumer  
- false: Not primary consumer  
Data type: Boolean  
Default: false |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.consumer.state</td>
<td>State in which the consumer resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.street</td>
<td>Consumer street address.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.suffix</td>
<td>Consumer name suffix such as Jr., Sr., or II.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_created_by</td>
<td>User that created the consumer record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_created_on</td>
<td>Date and time the consumer record was originally created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_domain</td>
<td>ServiceNow domain in which the consumer information resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_id</td>
<td>Unique identifier for the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_mod_count</td>
<td>Number of times that the associated consumer information has been modified.</td>
</tr>
<tr>
<td></td>
<td>Data type: Integer</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_tags</td>
<td>Consumer system tags.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_updated_by</td>
<td>User that last updated the consumer information.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_updated_on</td>
<td>Date and time when the consumer information was last updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.time_format</td>
<td>Format in which to display time.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• hh.mm.ss a: hh.mm.ss</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>• HH.mm.ss: hh.mm.ss</td>
</tr>
<tr>
<td></td>
<td>• HH:mm:ss: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system time format)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.time_zone</td>
<td>Consumer time zone, such as Canada/Central or US/Eastern.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.title</td>
<td>Consumer business title such as Manager, Software Developer, Contractor.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 60</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.user</td>
<td>Sys_id of the consumer user. Located in the Consumer User [csm_consumer_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.zip</td>
<td>Consumer zip code.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact</td>
<td>For business-to-business cases. Unique identifier of the record containing details about the social media profile that belongs to the contact associated with the case. Located in the [customer_contact] table. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.account</td>
<td>Sys_id of the account record to which the contact is associated; Account [customer_account] table. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.active</td>
<td>Flag that indicates whether the contact is active within the system. Possible values: • true: Contact is active • false: Contact is inactive Data type: Boolean Default: true</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.agent_status</td>
<td>Status of the agent. Possible values: • Off work • On break • On route • On site Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.building</td>
<td>Sys_id of the record that describes the building</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>the contact resides; Building [cmn_building] table.</td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.contact.calendar_integration | Calendar application that the contact uses.  
• 1: Outlook | Data type: Number (Integer)  
Default: 1 |
| sn_app_cs_social_social_profile.contact.city | City in which the contact resides. | Data type: String  
Maximum length: 40 |
| sn_app_cs_social_social_profile.contact.company | Sys_id of the company to which the contact is associated; Company [core_company] table. | Data type: String |
| sn_app_cs_social_social_profile.contact.cost_center | Sys_id of the cost center associated with the contact; Cost Center [cmn_cost_center] table. | Data type: String |
| sn_app_cs_social_social_profile.contact.country | Country code of the country in which the contact resides. | Data type: String  
Maximum length: 3 |
| sn_app_cs_social_social_profile.contact.date_format | Format in which to display dates to contacts.  
Valid values:  
• dd/mm/yyyy  
• dd-mm-yyyy  
• dd.mm.yyyy |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| mm-dd-yyyy                                                             | • mm-dd-yyyy  
| yyy-mm-dd                                                              | Data type: String  
| Maximum length: 40  
| Default: blank (system format)                                        |                                                                                                                                                                                                            |
| sn_app_cs_social_social_profile.contact.default_perspective            | Sys_id of the default perspective for the contact. Located in the Menu List [sys_perspective] table.  
| Data type: String                                                    |                                                                                                                                                                                                            |
| sn_app_cs_social_social_profile.contact.department                    | Sys_id of the department associated with the contact. Located in the Department [cmn_department] table.  
| Data type: String                                                    |                                                                                                                                                                                                            |
| sn_app_cs_social_social_profile.contact.edu_status                     | Education status of the associated contact.  
| Data type: String                                                    | Maximum length: 40  
| Default: faculty                                                     |                                                                                                                                                                                                            |
| sn_app_cs_social_social_profile.contact.email                         | Contact email address.  
| Data type: String                                                    |                                                                                                                                                                                                            |
| sn_app_cs_social_social_profile.contact.employee_number               | Contact employee number.  
| Data type: String                                                    |                                                                                                                                                                                                            |
| sn_app_cs_social_social_profile.contact.enable_multifactor_authn      | Flag that indicates whether multifactor authorization is required for the contact to log in to the service portal.  
<p>| Possible values:                                                     |                                                                                                                                                                                                            |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• true: Multifactor authorization enabled</td>
<td>• false: Multifactor authorization disabled</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td>Default: false</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.failed_attempts</td>
<td>Number of failed log in attempts.</td>
</tr>
<tr>
<td>Data type: Number (Integer)</td>
<td></td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.first_name</td>
<td>Contact first name.</td>
</tr>
<tr>
<td>Data type: String</td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.gender</td>
<td>Contact gender.</td>
</tr>
<tr>
<td>Data type: String</td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.geolocation_tracked</td>
<td>Flag that indicates whether the contact location is obtained through geotracking.</td>
</tr>
<tr>
<td>Possible values:</td>
<td>• true: Contact location obtained through geotracking</td>
</tr>
<tr>
<td></td>
<td>• false: Contact location not obtained through geotracking</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td>Default value: false</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.home_phone</td>
<td>Contact home phone number.</td>
</tr>
<tr>
<td>Data type: String</td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.internal_integration_user</td>
<td>Flag that indicates whether the contact is an internal integration user.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.contact.introduction</td>
<td>Introduction  &lt;br&gt; Data type: String  &lt;br&gt; Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.last_login</td>
<td>Date on which the contact last logged into the system.  &lt;br&gt; Data type: String (Date)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.last_login_device</td>
<td>Device the consumer used the last time they logged in to the system.  &lt;br&gt; Data type: String &lt;br&gt; Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.last_login_time</td>
<td>Date and time the contact last logged in to the system.  &lt;br&gt; Data type: String (Date/time)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.last_name</td>
<td>Contact last name.  &lt;br&gt; Data type: String  &lt;br&gt; Maximum length: 50</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.last_position_update</td>
<td>Date and time the last position was updated.  &lt;br&gt; Data type: String (Date/time)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.latitude</td>
<td>Latitude coordinate of contact.  &lt;br&gt; Data type: Number (Floating point)  &lt;br&gt; Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.ldap_server</td>
<td>Sys_id of the LDAP server used by the contact to last log in to the system; LDAP Server [ldap_server_config] table. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.location</td>
<td>Sys_id of the record that describes the location of the contact; Location [cmn_location] table. Data type: String</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.contact.locked_out | Flag that indicates if the contact is locked-out. Possible values:  
• true: Contact locked-out  
• false: Contact not locked-out  
Data type: Boolean  
Default: false |
| sn_app_cs_social_social_profile.contact.longitude | Longitude coordinate of the contact. Data type: Number (Floating point)  
Maximum length: 40 |
| sn_app_cs_social_social_profile.contact.manager | Sys_id of the record that describes the direct supervisor of the contact; User [sys_user] table. Data type: String |
| sn_app_cs_social_social_profile.contact.middle_name | Contact middle name. Data type: Number (Floating point)  
Maximum length: 50 |
<p>| sn_app_cs_social_social_profile.contact.mobile_phone | Contact mobile phone number. Data type: String |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| sn_app_cs_social_social_profile.contact.name | Contact full name.  
Data type: String  
Maximum length: 151 |
| sn_app_cs_social_social_profile.contact.notification | Indicates whether the contact should receive notifications.  
Valid values:  
• 1: Disabled  
• 2: Enabled  
Data type: Number (Integer)  
Default: 2 |
| sn_app_cs_social_social_profile.contact.on_schedule | Indicates the timeliness of dispatched service personnel.  
Valid values:  
• Ahead: Ahead of schedule  
• behind_less30: Behind schedule, but less than 30 minutes  
• behind_30to60: Behind schedule between 30 and 60 minutes  
• behind_more60: Behind schedule more than 60 minutes  
• on_time: On schedule  
Data type: String  
Maximum length: 40 |
| sn_app_cs_social_social_profile.contact.phone | Contact business phone number.  
Data type: String  
Maximum length: 40 |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.contact.photo</td>
<td>Photo image of the contact. Data type: Image</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.preferred_language</td>
<td>Country code of the contact’s primary language. Data type: String Maximum length: 3</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.roles</td>
<td>List of user roles associated with the contact. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.schedule</td>
<td>Sys_id of the record that describes the work schedule for the associated contact. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.source</td>
<td>Source of the contact. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.state</td>
<td>State in which the contact resides. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.street</td>
<td>Contact street address. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_class_name</td>
<td>Table that contains the contact record. Data type: String Maximum length: 80</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_created_by</td>
<td>User that originally created the associated contact record. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_created_on</td>
<td>Data and time the associated contact was originally created. Data type: String (Date/Time)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_domain</td>
<td>ServiceNow instance of the associated contact. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_domain_path</td>
<td>Contact record domain path. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255 Default: / (global)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_id</td>
<td>Unique identifier for the associated contact record. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_mod_count</td>
<td>Number of times that the associated contact record has been modified. Data type: Number (Integer)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_tags</td>
<td>Contact system tags.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_updated_by</td>
<td>User that last updated the associated contact information. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_updated_on</td>
<td>Data and time the associated contact information was updated. Data type: String (Date/Time)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.time_format</td>
<td>Format in which to display time. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.time_sheet_policy</td>
<td>Sys_id of the record that contains the time sheet policy for the associated contact; Time Sheet Policy [time_sheet_policy] table.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.time_zone</td>
<td>Time zone in which the contact resides, such as Canada/Central or US/Eastern.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.title</td>
<td>Contact business title such as Manager, Software Developer, or Contractor.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.user_name</td>
<td>Contact user ID.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.vip</td>
<td>Flag that indicates whether the associated contact has VIP status. Possible values: true: VIP; false: Not VIP.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.contact.web_service_access_only | Flag that indicates whether the contact can only access through the web. Possible values:  
  - true: Web access only  
  - false: Access through all available methods  
  Data type: Boolean  
  Default: false |
| sn_app_cs_social_social_profile.contact.zip | Contact zip code.  
  Data type: String  
  Maximum length: 40 |
| sn_app_cs_social_social_profile.created_on | Date and time the associated social media profile was initially created. |
| sn_app_cs_social_social_profile.profile | Social profile.  
  Data type: String  
  Maximum length: 255 |
| sn_app_cs_social_social_profile.profile_url | URL to use to access the social media profile.  
  Data type: String  
  Maximum length: 1,024 |
| sn_app_cs_social_social_profile.social_id | Unique social media account provider identifier for the associated social media account.  
  Data type: String  
  Maximum length: 100 |
| sn_app_cs_social_social_profile.source | Source of the social profile  
  Data type: String |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.sys_created_by</td>
<td>User that initially created the social media profile.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_created_on</td>
<td>Date and time the social media profile was initially created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_id</td>
<td>Unique identifier for the social media profile.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_mod_count</td>
<td>Number of times that information was modified for the associated social media profile.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_tags</td>
<td>Profile system tags.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_updated_by</td>
<td>User that initially created the social media profile.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_updated_on</td>
<td>User that initially created the social media profile.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>state</td>
<td>Current state of the case.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: New</td>
</tr>
<tr>
<td></td>
<td>• 3: Closed</td>
</tr>
<tr>
<td></td>
<td>• 6: Resolved</td>
</tr>
<tr>
<td></td>
<td>• 10: Open</td>
</tr>
<tr>
<td></td>
<td>• 18: Awaiting Info</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------------------------------------</td>
</tr>
</tbody>
</table>
| Data type: Number (Integer)  
Default: 1  |
| subcategory         | Case subcategory. Possible values:  
• 0: Question  |
| Data type: Number (Integer)  
Default: 0  |
| support_manager     | Sys_id of the CSM manager assigned to the case. Located in the User [sys_user] table.  |
| Data type: String    |
| sync_driver         | Flag that indicates whether there is driver synchronization. Possible values:  
• true: Driver is synchronized  
• false: Driver is not synchronized  |
| Data type: Boolean   |
| sys_class_name      | Table that contains the case record.  |
| Data type: String    
Maximum length: 80  |
| sys_created_by      | Person that initially opened the case.  |
| Data type: String    
Maximum length: 40  |
| sys_created_on      | Date and time when the case was initially created.  |
| Data type: String    |
| sys_domain          | Domain associated with the case.  |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| sys_domain_path | Domain path.  
Data type: String  
Maximum length: 255  
Default: / |
| sys_id        | Unique identifier for the case.  
Data type: String  
Maximum length: 32 |
| sys_mod_count  | Number of updates to the case since it was initially created.  
Data type: Number (Integer) |
| sys_tags       | System tags. |
| sys_updated_by | Person that last updated the case.  
Data type: String  
Maximum length: 40 |
| sys_updated_on | Date and time when the case was last updated.  
Data type: String |
| time_worked   | Total amount of time worked on the case.  
Data type: String |
| upon_approval  | Action to take if the case is approved.  
Possible values:  
• do_nothing  
• proceed  
Data type: String |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>upon_reject</td>
<td>Action to take if the case is rejected. Possible values: • cancel • goto Data type: String Maximum length: 40 Default: cancel</td>
</tr>
<tr>
<td>urgency</td>
<td>Urgency of the case. Possible values: • 1: High • 2: Medium • 3: Low Data type: Number (Integer) Default: 3</td>
</tr>
<tr>
<td>user_input</td>
<td>Additional user input. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>variable_pool</td>
<td>Grouping of variables.</td>
</tr>
<tr>
<td>variables</td>
<td>Name-value pairs of variables associated with the case. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>watch_list</td>
<td>List of sys_ids of the users who receive notifications about the case when additional comments are added or if the state of a case is changed to Resolved</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>or Closed</td>
<td>Located in the [sys_user] table. Data type: Array</td>
</tr>
<tr>
<td>wf_activity</td>
<td>Sys_id of the workflow activity record associated with the case. Located in the Workflow Activity [wf_activity] table. Data type: String</td>
</tr>
<tr>
<td>work_end</td>
<td>Date and time work ended on the case. Data type: String</td>
</tr>
<tr>
<td>work_notes</td>
<td>Information about how to resolve the case, or steps taken to resolve it. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>work_notes_list</td>
<td>List of sys_ids of the internal users who receive notifications about this case when work notes are added. Located in the [sys_user] table. Data type: Array</td>
</tr>
<tr>
<td>work_start</td>
<td>Date and time that work started on the case. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

Retrieve the third and fourth CSM cases listed for a specific account.

```bash
curl
  "https://instance.servicenow.com/api/sn_customerservice/case?sysparm_query=account=86837a386f0331003b3c498f5d3ee4ca&sysparm_limit=2&sysparm_offset=2" \
  --request GET \
  --header "Accept:application/json" \
  --user "username":"password"
```

The response shows the two cases.
"notify": "1",
"sys_class_name": "sn_customerservice_case",
"closed_by": "",
"follow_up": "",
"contact_local_time": "",
"sn_app_cs_social_social_profile": "",
"reassignment_count": "1",
"contact_time_zone": "",
"notes_to_comments": "false",
"assigned_to": "85453c616fc331003b3c498f5d3ee4fa",
"product": "5f5fbcc3c0a8010e00f3b27814f3b96b",
"variables": "",
"variable_pool": "",
"hierarchical_variables": "variable_pool",
"sla_due": "",
"change": "",
"comments_and_work_notes": "",
"partner": "",
"escalation": "0",
"upon_approval": "proceed",
"partner_contact": "",
"correlation_id": "",
"asset": "a8fb6024d7700200e5982cf65e61034d",
"made_sla": "true",
"resolved_by": "",
"sys_updated_by": "admin",
"opened_by": "60beb5e7d7600200e5982cf65e6103ad",
"user_input": "",
"sys_created_on": "2016-09-19 16:49:18",
"contact": {
  "country": "",
  "calendar_integration": "1",
  "last_position_update": "",
  "last_login_time": "2018-05-10 19:50:26",
  "last_login_device": "0:0:0:0:0:0:0:1%0",
  "source": "",
  "sys_updated_on": "2018-02-16 21:30:03",
  "building": "",
  "web_service_access_only": "false",
  "notification": "1",
  "sys_updated_by": "venki",
  "enable_multifactor_authn": "false",
  "sys_created_on": "2018-02-16 21:29:11",
  "sys_domain": "global"}
"account": {
"banner_image_light": "",
"country": "USA",
"parent": "",
"notes": "",
"stock_symbol": "",
"discount": "",
"active_escalation": "",
"sys_updated_on": "2019-01-03 19:37:35",
"apple_icon": "",
"number": "ACCT0000001",
"sys_updated_by": "admin",
"fiscal_year": "",
"sys_created_on": "2018-02-20 21:29:16",
"contact": "4d147a386f0331003b3c498f5d3ee437",
"stock_price": "",
"state": "",
"banner_image": "",
"sys_created_by": "venki",
"longitude": "-122.4005464",
"zip": "94103",
"profits": "0",
"phone": "(626) 722-6999",
"fax_phone": "",
"name": "Boxeo",
"banner_text": "",
"account_code": "~~~~1",
"primary": "false",
"city": "San Francisco",
"latitude": "37.7838391",
"sys_class_name": "customer_account",
"manufacturer": "false",
"account_parent": "",
"sys_id": "86837a386f0331003b3c498f5d3ee4ca",
"market_cap": "0",
"num_employees": "",
"rank_tier": "",
"street": "747 Howard St",
"vendor": "false",
"lat_long_error": "",
"theme": "",
"vendor_type": "",
"website": "www.boxeo.com",
"revenue_per_year": "0",

5511

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


"publicly_traded": "false",
"sys_mod_count": "9",
"sys_tags": "",
"partner": "false",
"registration_code": "BOXEO",
"vendor_manager": "",
"account_path": "~~~1",
"primary_contact": "4d147a386f0331003b3c498f5d3ee437",
"customer": "true"
},

"parent": "",
"caused_by": "",
"watch_list": "",
"active_escalation": "",
"upon_reject": "cancel",
"sys_updated_on": "2019-01-15 23:28:00",
"support_manager": "",
"approval_history": "",
"skills": "a6d64b11d772120058c92cf65e610357,9b178b11d772120058c92cf65e610382,1c76ec15d703120058c92cf65e61034a",
"number": "CS0000005",
"problem": "",
"state": "6",
"case": "Urgent issue, need help ...CS0000005",
"sys_created_by": "admin",
"knowledge": "false",
"order": "",
"cmdb_ci": "",
"delivery_plan": "",
"impact": "",
"contract": "495e16b96f0731003b3c498f5d3ee4af",
"active": "true",
"work_notes_list": "",
"auto_created_case": "false",
"priority": "1",
"sys_domain_path": "/",
"rejection_goto": "",
"first_response_time": "",
"business_duration": "",
"group_list": "",
"child_case_creation_progress": "false",
"sync_driver": "false"
Example: Python request

This request retrieves the seventh and eighth cases for a customer account, with display values instead of literal database values.

```python
# Install requests package for python
import requests

# Set the API endpoint URL for the request
url = 'https://instance.servicenow.com/api/sn_customerservice/case'

# Set the credentials
user = 'username'
pwd = 'password'

# Set the query parameters
params = {
    "sysparm_display_value": "true",
    "sysparm_query": "account=86837a386f0331003b3c498f5d3ee4ca",
    "sysparm_limit": 2,
    "sysparm_offset": 6
}

# Set the HTTP headers
```
headers = {"Accept": "application/xml"}

# Issue the HTTP request
response = requests.get(url, auth=(user, pwd), params=params, headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

Because `sysparm_display_value` is set to "true", the response includes display values, such as the state value of "Open," instead of literal database values. Removing this parameter or setting its value to "false" retrieves literal database values.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <parent/>
    <caused_by/>
    <watch_list/>
    <active_escalation/>
    <upon_reject>Cancel all future Tasks</upon_reject>
    <sys_updated_on>2020-03-18 05:47:00</sys_updated_on>
    <support_manager/>
    <approval_history/>
    <skills/>
    <number>CS0001016</number>
    <problem/>
    <initiated_as_request>false</initiated_as_request>
    <state>Open</state>
    <case>charging repeatedly\xe2\x80\x8bCS0001016</case>
    <sys_created_by>jodi.seals</sys_created_by>
    <knowledge>false</knowledge>
    <order/>
    <assigned_on/>
    <cmdb_ci/>
    <delivery_plan/>
    <impact>3 - Low</impact>
    <contract>CNTR00000603</contract>
  </result>
</response>
```
<cost_center/>
<name>George Warren</name>
<employee_number/>
<gender>null</gender>
<city/>
<hr_integration_source/>
<user_name>george.warren</user_name>
<failed_attempts>0</failed_attempts>
<edu_status>null</edu_status>
<latitude/>
<roles/>
<title>Network Administrator</title>
<sys_class_name>Contact</sys_class_name>
<sys_id>ddce70866f9331003b3c498f5d3ee417</sys_id>
<internal_integration_user>false</internal_integration_user>
<ldap_server/>
<mobile_phone>+1 858 867 7857</mobile_phone>
<street/>
<company>Boxeo</company>
<department/>
<first_name>George</first_name>
<preferred_language>null</preferred_language>
<introduction>null</introduction>
<email>geo.warren@mailinator.com</email>
<manager/>
<locked_out>false</locked_out>
/sys_mod_count>5</sys_mod_count>
<last_name>Warren</last_name>
<photo/>
<avatar>c0fbe5a0c3233100b12d9f2974d3ae89.iix</avatar>
<sys_tags/>
<middle_name/>
<time_zone>null</time_zone>
<schedule/>
<on_schedule>null</on_schedule>
<correlation_id/>
<date_format>null</date_format>
<location>2200 Powell Street, San Francisco, CA</location>
<Account>Boxeo</Account>
</contact>
<registration_code>BOXEO</registration_code>

/vendor_manager/>

/account_path>-----1</account_path>

/primary_contact>Julie Lewis</primary_contact>

/customer>true</customer>

</account>

</result>

/result>

/parent/>

/caused_by/>

/watch_list/>

/active_escalation/>

/upon_reject>CANCEL ALL FUTURE TASKS</upon_reject>

/sys_updated_on>2020-03-18 00:20:29</sys_updated_on>

/support_manager/>

/approval_history/>

/skills/>

/number>CS0001014</number>

/problem/>

/initiated_as_request>false</initiated_as_request>

/state>New</state>

/case>email server is down\xe2\x80\xe2\x80\x8bCS0001014</case>

/sys_created_by>george.warren</sys_created_by>

/knowledge>false</knowledge>

/order/>

/assigned_on/>

/cmdb_ci/>

/delivery_plan/>

/impact>3 - Low</impact>

/contract>CNTR0000603</contract>

/active>true</active>

/work_notes_list/>

/auto_created_case>false</auto_created_case>

/priority>4 - Low</priority>

/sys_domain_path>/</sys_domain_path>

/rejection_goto/>

/first_response_time/>

/business_duration/>

/group_list/>

/child_case_creation_progress>false</child_case_creation_progress>

/sync_driver>false</sync_driver>

/internal_contact/>

/approval_set/>

/wf_activity/>
<short_description>email server is down</short_description>
<sys_class_name>Order Case</sys_class_name>
<service_offering/>
<sys_created_on>2020-03-18 00:20:22</sys_created_on>
<opened_by>George Warren</opened_by>
<user_input/>
<country>null</country>
<made_sla>true</made_sla>
<sn_esign_document/>
<task_effective_number>CS0001014</task_effective_number>
<resolved_by/>
<sys_updated_by>system</sys_updated_by>
<opened_by>George Warren</opened_by>
<user_input/>
<sys_created_on>2020-03-18 00:20:22</sys_created_on>
<contact>null</contact>
<country>null</country>
<sys_updated_on>2020-03-16 12:37:35</sys_updated_on>

<apple_icon/>

<number>ACCT0000001</number>

<sys_updated_by>admin</sys_updated_by>

<fiscal_year/>

<sys_created_on>2019-05-04 14:29:16</sys_created_on>

<contact>Julie Lewis</contact>

<stock_price/>

<state/>

<sys_created_by>venki</sys_created_by>

<longitude>-122.4005464</longitude>
<zip>94103</zip>
<profits>$0.00</profits>
<phone>(626) 722-6999</phone>
<fax_phone/>
<primary>false</primary>
<city>San Francisco</city>
<primary_contact>Julie Lewis</primary_contact>
<customer>true</customer>
</account>
</result>
</response>

**Case - POST /sn_customerservice/case**

Creates a new Customer Service Management (CSM) case.

To use this endpoint, users must have the csm_ws_integration, sn_customerservice.customer, or sn_customerservice.consumer role.
You can create different types of cases depending on whether you are working in a business-to-business (B2B) or business-to-consumer (B2C) environment, or creating a case for an order or product.

In addition, you can create a social media profile for a contact, consumer, or account using this endpoint. To create a social media profile, you must specify the following parameters in the request body:

- `social_channel`
- `social_handle`
- `social_handle_type`
- `social_handle_url`
- `social_post_url`

All request body parameters are optional. Some parameters are only valid in one environment or the other (B2B or B2C), and are noted as such in the request body parameter descriptions. If you specify a contact, the account, case report, contract, and entitlement parameters are defaulted to the values defined in the contact/account records. If you specify an account, the case report, contract, and entitlement parameters are defaulted to the values defined in the account record (unless overridden by passed in parameters.) If you do not pass any parameters to the endpoint, the case record is created with only default information (defined in the table below) and instance generated values (`sys_*`). All other parameters are set with their equivalent of a null value. In addition to the list of parameters defined below, which define the parameters found in a base system, the endpoint also accepts custom case fields and any additional case fields configured in your instance. For additional information on these elements, refer to your specific table definitions [System Definition > Tables].

⚠️ **Warning:** This endpoint does not perform parameter validation as doing so can create excessive overhead. If a request parameter is misspelled, is not valid for the type of case, or is not supported by the endpoint, it is either ignored without warning or can cause unexpected results. For example, if you create a consumer or contact for the wrong type of case or mismatch a contact with an account.

⚠️ **Warning:** By default, external users who have the `sn_customerservice.customer` or `sn_customerservice.consumer` role can only set the account, asset, comments, consumer, contact, contact_type, partner, partner_contact, priority, product, short_description, and state fields using this endpoint.
**Note:** You can reference all `sysparm` query parameters using either their full name or their shortened name (without the `sysparm_` prefix). For example, for `sysparm_limit` you can also use `limit`.

**URL format**

Versioned URL: `/api/sn_customerservice/{api_version}/case`

Default URL: `/api/sn_customerservice/case`

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>api_version</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Request body parameters (XML or JSON)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element</td>
</tr>
<tr>
<td>------------</td>
</tr>
</tbody>
</table>
| account    | Sys_id of the account associated with the case; located in the Account [customer_account] table. Data type: String 
Default: If `contact` specified, defaults to account specified in contact record. |
<p>| active     | Flag that indicates whether the case is open and active. Possible values: true: Case is active false: Case is closed |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active_account_escalation</td>
<td>Sys_id of the active account escalation record associated with the case. Located in the Escalation [sn_customerservice_escalation] table. Default: String</td>
</tr>
<tr>
<td>active_escalation</td>
<td>Sys_id of the active account escalation record associated with the case. Located in the Escalation [sn_customerservice_escalation] table. Default: String</td>
</tr>
<tr>
<td>activity_due</td>
<td>Date for which the associated case is expected to be completed. Default: String</td>
</tr>
<tr>
<td>additional_assignee_list</td>
<td>List of the sys_ids of the additional persons (other than primary assignee) that have been assigned to the account. Located in the User [sys_user] table. Default: Array, Maximum: 4,000</td>
</tr>
<tr>
<td>approval</td>
<td>String that describes the type of approval required. Possible values:-approved, cancelled, duplicate, not_required, not_requested, rejected, requested</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>assigned_to</td>
<td>Sys_id of the person assigned to the case. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>assignment_group</td>
<td>Sys_id of the customer service agent group assigned to the case. Located in the Group [sys_user_group] table. Data type: String</td>
</tr>
<tr>
<td>business_duration</td>
<td>Length in calendar work hours, work days, and work weeks that it took to complete the case. Data type: String</td>
</tr>
<tr>
<td>business_impact</td>
<td>Impact of the issue on the associated customer. Data type: String Maximum characters: 4,000</td>
</tr>
<tr>
<td>business_service</td>
<td>Sys_id of the service record associated with the case. Located in the Service [cmdb_ci_service] table. Data type: String</td>
</tr>
<tr>
<td>category</td>
<td>Case category. Possible values: • 0: Question • 1: Issue • 2: Feature Data type: Number (Integer) Default: 1</td>
</tr>
<tr>
<td>cause</td>
<td>Details about the cause of the problem.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>caused_by</td>
<td>Sys_id of the change request that caused the case to be created. Located in the Change Request [change_request] table. Data type: String</td>
</tr>
<tr>
<td>change</td>
<td>Sys_id of the change request that caused the case to be created. Located in the Change Request [change_request] table. Data type: String</td>
</tr>
<tr>
<td>child_case_creation_progress</td>
<td>Flag that indicates whether the case is a child case that was created from a major case. Possible values: • true: Child case created from a major case • false: Not a child case. Data type: Boolean</td>
</tr>
<tr>
<td>close_notes</td>
<td>Notes made when the case was closed. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>closed_at</td>
<td>Date and time that the case was closed. Data type: String</td>
</tr>
<tr>
<td>closed_by</td>
<td>Sys_id of the user that closed the case. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>cmdb_ci</td>
<td>Sys_id of the configuration item associated with the case. Located in the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>comments</td>
<td>Additional comments about the case.</td>
</tr>
</tbody>
</table>
## Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>company</td>
<td>Sys_id of the company associated with the case. Located in the Company [core_company] table. Data type: String</td>
</tr>
<tr>
<td>consumer</td>
<td>Business-to-consumer cases only. Sys_id of the person to contact with regards to this case. Located in the Consumer [csm_consumer] table. Data type: String</td>
</tr>
<tr>
<td>contact</td>
<td>Business-to-business based cases only. Sys_id of the person to contact regarding this case. Located in the Contact [customer_contact] table. Data type: String</td>
</tr>
<tr>
<td>contact_time_zone</td>
<td>Time zone of the contact associated with the case. Data type: String Maximum length: 40</td>
</tr>
</tbody>
</table>
| contact_type | Method in which the case was initially reported. Possible values:  
- chat  
- email  
- phone  
- social  
- web  
Data type: String Maximum length: 40 |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| correlation_display | Correlation display.  
Data type: String  
Maximum length: 100                                                                                                                                      |
| correlation_id      | Correlation identifier.  
Data type: String  
Maximum length: 100                                                                                                                                      |
Data type: String                                                                                                                                         |
Data type: String                                                                                                                                         |
| description         | Detailed description of the problem associated with the case.  
Data type: String  
Maximum length: 4,000                                                                                                                                     |
| due_date            | Date that the case is due to be closed.  
Data type: String                                                                                                                                             |
<p>| entitlement         | Sys_id of the entitlement record associated with the case. Located in the Entitlement [service_entitlement] table. Entitlements define the type of support that a customer receives |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>as well as the supported communication channels. An entitlement can be associated with a product, an asset, an account, or a contract. Data type: String</td>
</tr>
<tr>
<td>escalation</td>
<td>Current escalation level. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Normal</td>
</tr>
<tr>
<td></td>
<td>• 1: Moderate</td>
</tr>
<tr>
<td></td>
<td>• 2: High</td>
</tr>
<tr>
<td></td>
<td>• 3: Overdue</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 0</td>
</tr>
<tr>
<td>expected_start</td>
<td>Date and time when work is scheduled to begin on the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>first_response_time</td>
<td>Date and time when the first action was taken on the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>follow_the_sun</td>
<td>Flag that indicates whether the case should be handed-off for global follow-up. If a customer enters additional comments on a Priority 1 - Critical or a Priority 2 - High case, or if the case is escalated, the flag is automatically set to true. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Case should be handed-off for global follow-up</td>
</tr>
<tr>
<td></td>
<td>• false: Case should not be handed-off for global follow-up</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Data type: Boolean  
Default: false | group_list List of sys_ids of the group records associated with the case. Located in the Group [sys_user_group] table.  
Data type: Array |
| Impact on customer.  
Possible values:  
- 1: High  
- 2: Medium  
- 3: Low  
Data type: Number (Integer)  
Default: 3 | impact |
| Flag that indicates if there is a knowledge base article available for the specified issue.  
Possible values:  
- true: Knowledge base article is available for this issue  
- false: Knowledge base article is not available for this issue  
Data type: Boolean  
Default: false | knowledge |
| Sys_id of the record describing the company location. Located in the Location [cmn_location] table.  
Data type: String | location |
<p>| Flag that indicates whether the case was resolved in alignment with the associated service level agreement. | made_sla |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Case was resolved in alignment with SLA</td>
</tr>
<tr>
<td></td>
<td>• false: Case was not resolved according to the SLA</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td>major_case_state</td>
<td>Current state of the major case.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• accepted: Initial state when a manager creates a major case or when a</td>
</tr>
<tr>
<td></td>
<td>manager promotes a candidate case.</td>
</tr>
<tr>
<td></td>
<td>• canceled: Case is canceled.</td>
</tr>
<tr>
<td></td>
<td>• proposed: Initial state when an agent or manager creates or proposes a</td>
</tr>
<tr>
<td></td>
<td>candidate case.</td>
</tr>
<tr>
<td></td>
<td>• rejected: Manager rejected candidate case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>notes_to_comments</td>
<td>Flag that indicates whether to add the resolution notes to the comments.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Resolutions notes, when added, are also added to the comments</td>
</tr>
<tr>
<td></td>
<td>• false: Resolution notes in comments are not required</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>notify</td>
<td>Method to use to notify contact/consumer.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| number           | Case number.  
Data type: String  
Maximum length: 40 |
| opened_at        | Date and time that the case was opened.  
Data type: String |
| opened_by        | Sys_id of the person that initially opened the case. Located in the User [sys_user] table.  
Data type: String |
| order            | Order of the case.  
Data type: Number (Integer) |
| parent           | Sys_id of the parent case to which this case (child) is associated. Located in the Task [task] table.  
Data type: String |
| partner          | Sys_id of the partner associated with the case. Located in the Account [customer_account] table.  
Data type: String |
| partner_contact  | Sys_id of the partner contact associated with the case. Located in the Contact [customer_contact] table.  
Data type: String |
| priority         | Priority of the case.  
Possible values: |

- 1: Do not notify
- 2: Send email
- 3: Telephone

Data type: Number (Integer)
Default: 1
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
|                    | • 1: Critical  
|                    | • 2: High  
|                    | • 3: Moderate  
|                    | • 4: Low  
|                    | Data type: Number (Integer)  
|                    | Default: 4  
| probable_cause     | Possible cause of the issue associated with the case.  
|                    | Data type: String  
|                    | Maximum length: 4,000  
| problem            | Sys_id of the issue that the customer is encountering. Located in the Problem [problem] table.  
|                    | Data type: String  
| product            | Sys_id of the product model of the asset associated to the case. Located in the Product Model [cmdb_model] table. A model is a specific version or configuration of an asset (for example, Apple Mac Book Pro).  
|                    | Data type: String  
| reassignment_count | Number of times that the case was reassigned to a person that is responsible for moving the case forward.  
|                    | Data type: Number (Integer)  
|                    | Default: 0  
| recipient_list     | Sys_id of the record that contains the list of recipients for communications about this case. Located in the Recipients List [sn_publications_recipients_list] table.  
|                    | Data type: String  

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rejection_goto</td>
<td>Sys_id of the task to execute if the case is rejected. Located in the Task [task] table. Data type: String</td>
</tr>
<tr>
<td>resolution_code</td>
<td>Resolution state for the case, such as &quot;Solved - Fixed by Support/Guidance provided&quot;. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>resolved_at</td>
<td>Date and time that the case was resolved. Data type: String</td>
</tr>
<tr>
<td>resolved_by</td>
<td>Sys_id of the person that resolved the case. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>short_description</td>
<td>Concise description of the case. Data type: String Maximum length: 160</td>
</tr>
<tr>
<td>skills</td>
<td>List of the unique identifiers (sys_id) of the skills needed to complete the case. Located in the Skill [cmn_skill] table. Data type: String</td>
</tr>
<tr>
<td>sla_due</td>
<td>Date/time at which the case must be closed based on the associated service level agreement. Data type: String</td>
</tr>
<tr>
<td>social_channel</td>
<td>Type of social media channel such as Twitter, Facebook, or Instagram. Data type: String</td>
</tr>
<tr>
<td>social_handle</td>
<td>User handle on the channel. Data type: String</td>
</tr>
<tr>
<td>social_handle_type</td>
<td>Type of case to associate the social media profile.</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• account</td>
</tr>
<tr>
<td></td>
<td>• consumer</td>
</tr>
<tr>
<td></td>
<td>• contact</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>social_handle_url</td>
<td>URL of the user's social media handle.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>social_post_url</td>
<td>URL of the social message from which the case is being generated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>Current state of the case.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: New</td>
</tr>
<tr>
<td></td>
<td>• 3: Closed</td>
</tr>
<tr>
<td></td>
<td>• 6: Resolved</td>
</tr>
<tr>
<td></td>
<td>• 10: Open</td>
</tr>
<tr>
<td></td>
<td>• 18: Awaiting Info</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 1</td>
</tr>
<tr>
<td>subcategory</td>
<td>Case subcategory.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Question</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 0</td>
</tr>
<tr>
<td>support_manager</td>
<td>Sys_id of the CSM manager assigned to the case. Located in the User [sys_user] table.</td>
</tr>
</tbody>
</table>
## Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sync_driver</td>
<td>Flag that indicates whether there is driver synchronization.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Driver is synchronized</td>
</tr>
<tr>
<td></td>
<td>• false: Driver is not synchronized</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>sys_domain_path</td>
<td>Domain path.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td></td>
<td>Default: /</td>
</tr>
<tr>
<td>time_worked</td>
<td>Total amount of time worked on the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>upon_approval</td>
<td>Action to take if the case is approved.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• do_nothing</td>
</tr>
<tr>
<td></td>
<td>• proceed</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: proceed</td>
</tr>
<tr>
<td>upon_reject</td>
<td>Action to take if the case is rejected.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• cancel</td>
</tr>
<tr>
<td></td>
<td>• goto</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Default: cancel</td>
<td>Request body parameters (XML or JSON) (continued)</td>
</tr>
<tr>
<td>urgency</td>
<td>Urgency of the case.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: High</td>
</tr>
<tr>
<td></td>
<td>• 2: Medium</td>
</tr>
<tr>
<td></td>
<td>• 3: Low</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 3</td>
</tr>
<tr>
<td>user_input</td>
<td>Additional user input.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>variables</td>
<td>Name-value pairs of variables associated with the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>watch_list</td>
<td>List of sys_ids of the users who receive notifications about this case when</td>
</tr>
<tr>
<td></td>
<td>additional comments are added or if the state of a case is changed to</td>
</tr>
<tr>
<td></td>
<td>Resolved or Closed.</td>
</tr>
<tr>
<td></td>
<td>Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>wf_activity</td>
<td>Sys_id of the workflow activity record associated with the case. Located in</td>
</tr>
<tr>
<td></td>
<td>the Workflow Activity [wf_activity] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>work_end</td>
<td>Date and time work ended on the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>work_notes_list</td>
<td>List of sys_ids of the internal users who receive notifications about this</td>
</tr>
<tr>
<td></td>
<td>case when work notes are added. Located in the User [sys_user] table.</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>work_start</td>
<td>Date and time that work started on the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>New case record was successfully created.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>
### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>number</td>
<td>Associated case number.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the new case record created in the Case [sn_customerservice_case] table.</td>
</tr>
</tbody>
</table>

**Data type:** String

#### Example: cURL request

The following example illustrates how to create a case based on a consumer and also create a new social media account that is associated with the case.

```bash
curl "https://instance.servicenow.com/api/sn_customerservice/case" \
--request POST \
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--data "{" 
  consumer: "cd97ef1ddb8bc7006b7a9646db9619ac", 
  contact_type: "social", 
  priority: "4", 
  short_description: "Consumer Test Case", 
  social_channel: "TWITTER", 
  social_handle: "consumertwo", 
  social_handle_type: "Consumer", 
  social_handle_url: "https://www.twitter.com/consumertwo", 
  social_post_url: "https://twitter.com/MuleSoft/status/829092185022750720"}" \
--user "username":"password"
```

```json
{
  "result": {
    "sys_id": "0397c743db436f0057c3fd441d96197f",
    "number": "CS0001005"
  }
}
```

#### Example: cURL request

The following example illustrates how to create a case for an account.

```bash
curl "https://instance.servicenow.com/api/sn_customerservice/case" \
--request POST \
--header "Accept:application/json" \
--header "Content-Type:application/json"
```
---header "Accept:application/json" \
---header "Content-Type:application/json" \
---data "{ \
    account: "051f62b0d7300200e5982cf65e610333", \
    contact_type: "phone", \
    priority: "1", \
    short_description: "Power Outage"}" \
---user "username":"password"

{
    "result": {
        "sys_id": "b88366ffdb1b0b00216e9ee6db96190b",
        "number": "CS0001021"
    }
}

Example: cURL request
The following example illustrates how to create a case for a specific contact. Note that if an account is specified within the contact record, that account is automatically assigned to the case.

curl "https://instance.servicenow.com/api/sn_customerservice/case" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --data "{ \
      contact: "62fe1c97db76c3006b7a9646db961999",
      contact_type: "phone", 
      priority: "3",
      short_description: "Power Outage"}" \
  --user "username":"password"

{
    "result": {
        "sys_id": "bf916affdb1b0b00216e9ee6db961937",
        "number": "CS0001013"
    }
}

Example: Python request

# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_customerservice/case'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Content-Type":"application/json", "Accept":"application/json"}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, data="{"contact":"62fe1c97db76c3006b7a9656db961999"}")

# Check for HTTP codes other than 201
if response.status_code != 201:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()

print(data)

{
    "result": {
        "sys_id":"c7940ed1db9ba30057c3fd41d961967",
        "number":"CS0001074"
    }
}

**Case - PUT /sn_customerservice/case/{id}**

Updates the specified existing Customer Service Management (CSM) case with the passed-in parameters.

⚠️ **Warning:** This endpoint does not perform parameter validation as doing so can create excessive overhead. If a request parameter is misspelled, is not valid for the type of case, or is not supported by the endpoint, it is either ignored without warning or can cause unexpected results. For example, if you create a consumer or contact for the wrong type of case or mismatch a contact with an account.
**Warning:** By default, external users who have the sn_customerservice.customer or sn_customerservice.consumer role can only set the comments and state fields using this endpoint. Admins can enable external users to set additional fields by specifying the field names in the UPDATE_ALLOWED_FIELDS array defined in the CaseRESTAPIValidatorForExternalUser script include. Exercise caution when adding new fields to this array, as there may not be existing validators for input into those fields. For more information on script includes, see Script includes.

**Note:** You can reference all sysparm query parameters using either their full name or their shortened name (without the sysparm_prefix). For example, for sysparm_limit you can also use limit.

**URL format**

Versioned URL: /api/sn_customerservice/{api_version}/case/{id}

Default URL: /api/sn_customerservice/case/{id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>Sys_id of the case to update. Located in the Case [sn_customerservice_case] table.</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_display_value</td>
<td>Data retrieval operation for reference and choice fields. Based on this value, retrieves the display value and/or the actual value from the database.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• true: Returns the display values for all fields.</td>
</tr>
<tr>
<td></td>
<td>• false: Returns the actual values from the database.</td>
</tr>
<tr>
<td></td>
<td>• all: Returns both actual and display values.</td>
</tr>
<tr>
<td>Default: false</td>
<td>Note: There is no preferred method for setting this parameter. However, specifying the display value may cause performance issues since it is not reading directly from the database and may include referencing other fields and records. For more information on display values and actual values, see Table API FAQs (KB0534905).</td>
</tr>
</tbody>
</table>

| sysparm_fields     | Comma-separated list of fields to return in the response.                |
|--------------------| Data type: String                                                        |
| sysparm_get_case_details | Flag that indicates whether to return the updated case record.        |
|                    | Valid values:                                                            |
|                    | • true: Return updated case record.                                       |
|                    | • false: Do not return the updated case record.                          |
| Data type: Boolean | Default: false                                                           |

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account</td>
<td>Sys_id of the account associated with the case. Located in the Account [customer_account] table.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Default: If contact specified, defaults to account specified in contact record.</td>
</tr>
<tr>
<td>active</td>
<td>Flag that indicates whether the case is open and active. Possible values: • true: Case is active • false: Case is closed</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td>active_account_escalation</td>
<td>Sys_id of the active account escalation record associated with the case. Located in the Escalation [sn_customerservice_escalation] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>active_escalation</td>
<td>Sys_id of the active account escalation record associated with the case. Located in the Escalation [sn_customerservice_escalation] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>activity_due</td>
<td>Date for which the associated case is expected to be completed.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>additional_assignee_list</td>
<td>List of the sys_ids of the additional persons (other than primary assignee) that have been assigned to the account. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array Maximum: 4,000</td>
</tr>
<tr>
<td>approval</td>
<td>String that describes the type of approval required. Possible values:</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>approved</td>
<td>• approved</td>
</tr>
<tr>
<td>cancelled</td>
<td>• cancelled</td>
</tr>
<tr>
<td>duplicate</td>
<td>• duplicate</td>
</tr>
<tr>
<td>not_required</td>
<td>• not_required</td>
</tr>
<tr>
<td>not_requested</td>
<td>• not_requested</td>
</tr>
<tr>
<td>rejected</td>
<td>• rejected</td>
</tr>
<tr>
<td>requested</td>
<td>• requested</td>
</tr>
<tr>
<td>assigned_to</td>
<td>Sys_id of the person assigned to the case. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td>assignment_group</td>
<td>Sys_id of the customer service agent group assigned to the case. Located in the Group [sys_user_group] table.</td>
</tr>
<tr>
<td>business_duration</td>
<td>Length in calendar work hours, work days, and work weeks that it took to complete the case.</td>
</tr>
<tr>
<td>business_impact</td>
<td>Impact of the issue on the associated customer.</td>
</tr>
<tr>
<td></td>
<td>Maximum characters: 4,000</td>
</tr>
<tr>
<td>business_service</td>
<td>Sys_id of the service record associated with the case. Located in the Service [cmdb_ci_service] table.</td>
</tr>
<tr>
<td>category</td>
<td>Case category.</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Question</td>
</tr>
<tr>
<td></td>
<td>• 1: Issue</td>
</tr>
<tr>
<td></td>
<td>• 2: Feature</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 1</td>
</tr>
<tr>
<td>cause</td>
<td>Details about the cause of the problem.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>caused_by</td>
<td>Sys_id of the change request that caused the case to be created. Located in</td>
</tr>
<tr>
<td></td>
<td>the Change Request [change_request] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>change</td>
<td>Sys_id of the change request that caused the case to be created. Located in</td>
</tr>
<tr>
<td></td>
<td>the Change Request [change_request] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>child_case_creation_progress</td>
<td>Flag that indicates whether the case is a child case that was created from</td>
</tr>
<tr>
<td></td>
<td>a major case.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Child case created from a major case</td>
</tr>
<tr>
<td></td>
<td>• false: Not a child case</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>close_notes</td>
<td>Notes made when the case was closed.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>closed_at</td>
<td>Date and time that the case was closed.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>closed_by</td>
<td>Sys_id of the user that closed the case. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>cmdm_cbi</td>
<td>Sys_id of the configuration item associated with the case. Located in the Configuration Item [cmdm_ci] table. Data type: String</td>
</tr>
<tr>
<td>comments</td>
<td>Additional comments about the case. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>company</td>
<td>Sys_id of the company associated with the case. Located in the Company [core_company] table. Data type: String</td>
</tr>
<tr>
<td>consumer</td>
<td>Business-to-consumer cases only. Sys_id of the person to contact with regards to this case. Located in the Consumer [csm_consumer] table. Data type: String</td>
</tr>
<tr>
<td>contact</td>
<td>Business-to-business based cases only. Sys_id of the person to contact regarding this case. Located in the Contact [customer_contact] table. Data type: String</td>
</tr>
<tr>
<td>contact_time_zone</td>
<td>Time zone of the contact associated with the case. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>contact_type</td>
<td>Method in which the case was initially reported. Possible values: chat, email</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contract</td>
<td>Sys_id of the contract associated with the case. Located in the Contract [ast_contract] table. This contract contains information about the type of support that is provided to the company associated to the case. A contract can include a company and contact and the specific assets that are covered. A contract can also include multiple service entitlements and SLAs. Data type: String</td>
</tr>
</tbody>
</table>
| correlation_display | Correlation display.  
Data type: String  
Maximum length: 100                                                                                                                                      |
| correlation_id  | Correlation identifier.  
Data type: String  
Maximum length: 100                                                                                                                                     |
Data type: String                                                                                                                                                |
Data type: String                                                                                                                                                |
<p>| description     | Detailed description of the problem associated with the case.                                                                                              |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>due_date</td>
<td>Date that the case is due to be closed. Data type: String</td>
</tr>
<tr>
<td>entitlement</td>
<td>Sys_id of the entitlement record associated with the case. Located in the Entitlement [service_entitlement] table. Entitlements define the type of support that a customer receives as well as the supported communication channels. An entitlement can be associated with a product, an asset, an account, or a contract. Data type: String</td>
</tr>
</tbody>
</table>
| escalation       | Current escalation level. Possible values:  
- 0: Normal  
- 1: Moderate  
- 2: High  
- 3: Overdue  
Data type: Number (Integer)  
Default: 0                                                                                                                                                                                                                                     |
| expected_start   | Date and time when work is scheduled to begin on the case. Data type: String                                                                                                                                                                                                       |
| first_response_time | Date and time when the first action was taken on the case. Data type: String                                                                                                                                                                                                         |
| follow_the_sun   | Flag that indicates whether the case should be handed-off for global follow-up.                                                                                                                                                                                                       |
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
|           | If a customer enters additional comments on a Priority 1 - Critical or a Priority 2 - High case, or if the case is escalated, the flag is automatically set to true. Possible values:  
  • true: Case should be handed-off for global follow-up  
  • false: Case should not be handed-off for global follow-up  
  Data type: Boolean  
  Default: false |
| group_list | List of sys_ids of the group records associated with the case. Located in the Group [sys_user_group] table.  
  Data type: Array |
| impact    | Impact on customer. Possible values:  
  • 1: High  
  • 2: Medium  
  • 3: Low  
  Data type: Number (Integer)  
  Default: 3 |
| knowledge | Flag that indicates if there is a knowledge base article available for the specified issue. Possible values:  
  • true: Knowledge base article is available for this issue  
  • false: Knowledge base article is not available for this issue |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
|                       | Data type: Boolean  
|                       | Default: false  |
| location              | Sys_id of the record describing the company location. Located in the Location [cmn_location] table.  
|                       | Data type: String |
| made_sla              | Flag that indicates whether the case was resolved in alignment with the associated service level agreement.  
|                       | Possible values:  
|                       | • true: Case was resolved in alignment with SLA  
|                       | • false: Case was not resolved according to the SLA  
|                       | Data type: Boolean  
|                       | Default: true |
| major_case_state      | Current state of the major case.  
|                       | Possible values:  
|                       | • accepted: Initial state when a manager creates a major case or when a manager promotes a candidate case.  
|                       | • canceled: Case is canceled.  
|                       | • proposed: Initial state when an agent or manager creates or proposes a candidate case.  
|                       | • rejected: Manager rejected candidate case.  
|                       | Data type: String  
|                       | Maximum length: 40 |
| notes_to_comments     | Flag that indicates whether to add the resolution notes to the comments.  
|                       | Possible values: |
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| **true**: Resolutions notes, when added, are also added to the comments  
**false**: Resolution notes in comments are not required |
| Data type: Boolean |
| **notify** | Method to use to notify contact/consumer.  
Possible values:  
• 1: Do not notify  
• 2: Send email  
• 3: Telephone |
| Data type: Number (Integer)  
Default: 1 |
| **number** | Case number.  
Data type: String  
Maximum length: 40 |
| **opened_at** | Date and time that the case was opened.  
Data type: String |
| **opened_by** | Sys_id of the person that initially opened the case. Located in the User [sys_user] table.  
Data type: String |
| **order** | Order of the case.  
Data type: Number (Integer) |
| **parent** | Sys_id of the parent case to which this case (child) is associated. Located in the Task [task] table.  
Data type: String |
## Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>partner</td>
<td>Sys_id of the partner associated with the case. Located in the Account [customer_account] table. Data type: String</td>
</tr>
<tr>
<td>partner_contact</td>
<td>Sys_id of the partner contact associated with the case. Located in the Contact [customer_contact] table. Data type: String</td>
</tr>
<tr>
<td>priority</td>
<td>Priority of the case. Possible values: 1: Critical 2: High 3: Moderate 4: Low Data type: Number (Integer) Default: 4</td>
</tr>
<tr>
<td>probable_cause</td>
<td>Possible cause of the issue associated with the case. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>problem</td>
<td>Sys_id of the issue that the customer is encountering. Located in the Problem [problem] table. Data type: String</td>
</tr>
<tr>
<td>product</td>
<td>Sys_id of the product model of the asset associated to the case. Located in the Product Model [cmdb_model] table. A model is a specific version or configuration of an asset (for example, Apple Mac Book Pro). Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>reassignment_count</td>
<td>Number of times that the case was reassigned to a person that is responsible for moving the case forward.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 0</td>
</tr>
<tr>
<td>recipient_list</td>
<td>Sys_id of the record that contains the list of recipients for communications about this case. Located in the Recipients List [sn_publications_recipients_list] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>rejection_goto</td>
<td>Sys_id of the task to execute if the case is rejected. Located in the Task [task] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>resolution_code</td>
<td>Resolution state for the case, such as &quot;Solved -Fixed by Support/Guidance provided&quot;.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>resolved_at</td>
<td>Date and time that the case was resolved.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>resolved_by</td>
<td>Sys_id of the person that resolved the case. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>short_description</td>
<td>Concise description of the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160</td>
</tr>
<tr>
<td>skills</td>
<td>List of the unique identifiers (sys_id) of the skills needed to complete the case. Located in the Skill [cmn_skill] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| sla_due               | Date/time at which the case must be closed based on the associated service level agreement.  
Data type: String |
| social_channel        | Type of social media channel such as Twitter, Facebook, or Instagram.  
Data type: String |
| social_handle         | User handle on the channel.  
Data type: String |
| social_handle_type    | Type of case to associate the social media profile.  
Valid values:  
• account  
• contact  
• consumer  
Data type: String |
| social_handle_url     | URL of the user's social media handle.  
Data type: String |
| social_post_url       | URL of the social message from which the case is being generated.  
Data type: String |
| state                 | Current state of the case.  
Possible values:  
• 1: New  
• 3: Closed  
• 6: Resolved  
• 10: Open  
• 18: Awaiting Info |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subcategory</td>
<td>Case subcategory. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Question</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer) Default: 0</td>
</tr>
<tr>
<td>support_manager</td>
<td>Sys_id of the CSM manager assigned to the case. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sync_driver</td>
<td>Flag that indicates whether there is driver synchronization. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Driver is synchronized</td>
</tr>
<tr>
<td></td>
<td>• false: Driver is not synchronized</td>
</tr>
<tr>
<td>sys_domain_path</td>
<td>Domain path. Maximum length: 255 Default: /</td>
</tr>
<tr>
<td>time_worked</td>
<td>Total amount of time worked on the case.</td>
</tr>
<tr>
<td>upon_approval</td>
<td>Action to take if the case is approved. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• do_nothing</td>
</tr>
<tr>
<td></td>
<td>• proceed</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> String</td>
</tr>
<tr>
<td></td>
<td><strong>Maximum length:</strong> 40</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> proceed</td>
</tr>
<tr>
<td>upon_reject</td>
<td>Action to take if the case is rejected. Possible values: • cancel • goto <strong>Data type:</strong> String</td>
</tr>
<tr>
<td></td>
<td><strong>Maximum length:</strong> 40</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> cancel</td>
</tr>
<tr>
<td>urgency</td>
<td>Urgency of the case. Possible values: • 1: High • 2: Medium • 3: Low <strong>Data type:</strong> Number (Integer)</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 3</td>
</tr>
<tr>
<td>user_input</td>
<td>Additional user input. <strong>Data type:</strong> String <strong>Maximum length:</strong> 4,000</td>
</tr>
<tr>
<td>variables</td>
<td>Name-value pairs of variables associated with the case. <strong>Data type:</strong> String <strong>Maximum length:</strong> 40</td>
</tr>
<tr>
<td>watch_list</td>
<td>List of sys_ids of the users who receive notifications about this case when additional comments are added or if the state of a case is changed. <strong>Data type:</strong> String <strong>Maximum length:</strong> 40</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>changed to Resolved or Closed. Located in the User [sys_user] table. Data type: Array</td>
</tr>
<tr>
<td>wf_activity</td>
<td>Sys_id of the workflow activity record associated with the case. Located in the Workflow Activity [wf_activity] table. Data type: String</td>
</tr>
<tr>
<td>work_end</td>
<td>Date and time work ended on the case. Data type: String</td>
</tr>
<tr>
<td>work_notes_list</td>
<td>List of sys_ids of the internal users who receive notifications about this case when work notes are added. Located in the User [sys_user] table. Data type: Array</td>
</tr>
<tr>
<td>work_start</td>
<td>Date and time that work started on the case. Data type: String</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>

#### Response headers

| Header | Description |
|--------|-------------|-------------|
| None   |             |-------------|
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Successful.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account</td>
<td>Sys_id of the account record associated with the case. Located in the Account [customer_account] table. Data type: String</td>
</tr>
</tbody>
</table>

Account details. Data type: Object

```
"account": {
    "account_code": "String",
    "account_parent": "String",
    "account_path": "String",
    "active_escalation": "String",
    "apple_icon": Image,
    "banner_image": Image,
    "banner_image_light": Image,
    "banner_text": "String",
    "city": "String",
    "contact": "String",
    "country": "String",
    "customer": "String",
    ...
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;account.account_code&quot;</td>
<td>Unique combination of values that an application uses</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account.account_parent</td>
<td>Sys_id of the parent account of this account. Located in the Account table. Data type: String</td>
</tr>
<tr>
<td>account.account_path</td>
<td>Path from the parent to child accounts in the account hierarchy.            Data type: String</td>
</tr>
<tr>
<td>account.active_escalation</td>
<td>Sys_id of the active escalation associated with the account. Located in the Escalation table. Data type: String</td>
</tr>
<tr>
<td>account.apple_icon</td>
<td>Icon for iPhone home page bookmarks.                                        Data type: Image</td>
</tr>
<tr>
<td>account.banner_image</td>
<td>Banner image that appears on the customer portal.                           Data type: Image</td>
</tr>
<tr>
<td>account.banner_image_light</td>
<td>Small banner image.                                                           Data type: Image</td>
</tr>
<tr>
<td>account.banner_text</td>
<td>Banner text that appears on the customer portal.                            Data type: String</td>
</tr>
</tbody>
</table>

Data type: String
Maximum length: 255
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account.city</td>
<td>City in which the company that is associated with this account resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>account.contact</td>
<td>Sys_id of a contact record associated with this account. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>account.country</td>
<td>Country in which the company that is associated with this account resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: USA</td>
</tr>
<tr>
<td>account.customer</td>
<td>Flag that indicates whether the account is a customer account, as opposed to a partner account. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Customer account</td>
</tr>
<tr>
<td></td>
<td>• false: Partner account</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>account.discount</td>
<td>Discount given to the account on purchases.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 15</td>
</tr>
<tr>
<td>account.fax_phone</td>
<td>Primary fax phone number for the company associated with this account.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>account.fiscal_year</td>
<td>Fiscal year for the company associated with the account. Data type: String</td>
</tr>
<tr>
<td>account.lat_long_error</td>
<td>Difference in the actual location as compared to latitude and longitude information. Data type: String Maximum length: 1,000</td>
</tr>
<tr>
<td>account.latitude</td>
<td>Latitude of the company associated with the account. Data type: Number (floating point number) Maximum length: 40</td>
</tr>
<tr>
<td>account.longitude</td>
<td>Longitude of the company associated with the account. Data type: Number (floating point number) Maximum length: 40</td>
</tr>
</tbody>
</table>
| account.manufacturer    | Flag that indicates whether the company associated with this account manufactures goods. Possible values:  
• true: Manufactures goods  
• false: Does not manufacture goods Data type: Boolean Default: false |
<p>| account.market_cap      | Market value of the associated company's publicly traded stock shares. Data type: Number (currency)                                           |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account.name</td>
<td>Name of the company associated with this account. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 20</td>
</tr>
<tr>
<td>account.notes</td>
<td>Additional information about the company. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 80</td>
</tr>
<tr>
<td>account.num_employees</td>
<td>Number of people employed by the company. Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.number</td>
<td>Number that identifies this account. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.parent</td>
<td>Sys_id of the parent account of this account. Located in the Company [core_company] table. Data type: String</td>
</tr>
</tbody>
</table>
| account.partner | Flag that indicates whether the account is a partner account or a customer account. Possible values:  
|                 | • true: Partner account  
|                 | • false: Customer account  
|                 | Data type: Boolean  
<p>|                 | Default: false                                                           |
| account.phone   | Primary phone number for the company.                                      |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account.primary</td>
<td>Flag that indicates whether a primary account.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Primary account</td>
</tr>
<tr>
<td></td>
<td>• false: Secondary account</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>account.primary_contact</td>
<td>Sys_id of the primary contact for the account.</td>
</tr>
<tr>
<td></td>
<td>Located in the Contact table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>account.profits</td>
<td>Profit information entered for the account.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Currency)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>account.publicly_traded</td>
<td>Flag that indicates whether the company associated with this account is</td>
</tr>
<tr>
<td></td>
<td>publicly traded on the stock exchange.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Publicly traded</td>
</tr>
<tr>
<td></td>
<td>• false: Private company</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>account.rank_tier</td>
<td>Type of account.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• blacklist</td>
</tr>
<tr>
<td></td>
<td>• strategic</td>
</tr>
<tr>
<td></td>
<td>• tactical</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
|                               | • valued  
|                               | • other  
|                               | Data type: String  
|                               | Maximum length: 40  
| account.registration_code    | Unique code that customers use when requesting a login on the customer portal. This provides a method for validating the customer on the company before granting access.  
|                               | Data type: String  
|                               | Maximum length: 40  
| account.revenue_per_year     | Revenue produced by the company associated with this account.  
|                               | Data type: Number (Currency)  
|                               | Maximum length: 20  
| account.state                | State in which the company resides.  
|                               | Data type: String  
|                               | Maximum length: 40  
| account.stock_price           | Price of the company stock.  
|                               | Data type: String  
|                               | Maximum length: 40  
| account.stock_symbol          | Stock symbol of the company.  
|                               | Data type: String  
|                               | Maximum length: 40  
| account.street               | Street address of the company.  
|                               | Data type: String  
|                               | Maximum length: 255  

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account.sys_class_name</td>
<td>Table that contains the associated account record. Data type: String</td>
</tr>
<tr>
<td>account.sys_created_by</td>
<td>User that originally created the account. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>account.sys_created_on</td>
<td>Date and time that the account was originally created. Data type: String</td>
</tr>
<tr>
<td>account.sys_id</td>
<td>Sys_id for the account. Data type: String</td>
</tr>
<tr>
<td>account.sys_mod_count</td>
<td>Number of times the account information has been updated. Data type: Number (Integer)</td>
</tr>
<tr>
<td>account.sys_updated_by</td>
<td>User that last modified the account information. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>account.sys_updated_on</td>
<td>Date and time the account information was last updated. Data type: String</td>
</tr>
<tr>
<td>account.theme</td>
<td>Sys_id of the customer theme used by this account. Located in the Theme [sys_ui_theme] table. Data type: String</td>
</tr>
<tr>
<td>account.vendor</td>
<td>Flag that indicates whether the company associated with the account is a vendor. Possible values:</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>account.vendor_manager</td>
<td>List of sys_ids of the vendor managers for the account. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>account.vendor_type</td>
<td>List of sys_ids of the type of vendor such as, applications, hardware, services, or software. Located in the Vendor Type [vendor_type] table. Data type: String</td>
</tr>
<tr>
<td>account.website</td>
<td>URL of the website for the company. Data type: String</td>
</tr>
<tr>
<td>account.zip</td>
<td>Zip code of the company. Data type: String</td>
</tr>
<tr>
<td>active</td>
<td>Flag that indicates whether the case is open and active. Possible values: • true: Case is active • false: Case is closed. Data type: Boolean Default: true</td>
</tr>
<tr>
<td>active_account_escalation</td>
<td>Sys_id of the active account escalation record.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>associated with the case</td>
<td>located in the Escalation [sn_customerservice_escalation] table. Data type:</td>
</tr>
<tr>
<td></td>
<td>String</td>
</tr>
<tr>
<td>active_escalation</td>
<td>sys_id of the active account escalation record associated with the case.</td>
</tr>
<tr>
<td></td>
<td>located in the Escalation [sn_customerservice_escalation] table. Data type:</td>
</tr>
<tr>
<td></td>
<td>String</td>
</tr>
<tr>
<td>activity_due</td>
<td>date for which the associated case is expected to be completed. Data type:</td>
</tr>
<tr>
<td></td>
<td>String</td>
</tr>
<tr>
<td>additional_assignee_list</td>
<td>list of the sys_ids of the persons (other than primary assignee) that have</td>
</tr>
<tr>
<td></td>
<td>been assigned to the account. located in the User [sys_user] table. Data</td>
</tr>
<tr>
<td></td>
<td>type: Array Maximum: 4,000</td>
</tr>
<tr>
<td>approval</td>
<td>string that describes the approval required. Possible values:</td>
</tr>
<tr>
<td></td>
<td>approved, cancelled, duplicate, not_required, not_requested, rejected,</td>
</tr>
<tr>
<td></td>
<td>requested</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>approval_history</td>
<td>List of all approvals associated with the case.</td>
</tr>
<tr>
<td>approval_set</td>
<td>Date and time that the associated action was approved.</td>
</tr>
<tr>
<td>asset</td>
<td>Sys_id of the asset record associated with the case. Located in the Asset [alm_asset] table.</td>
</tr>
<tr>
<td>assigned_to</td>
<td>Sys_id of the person assigned to the case. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td>assignment_group</td>
<td>Sys_id of the customer service agent group assigned to the case. Located in the Group [sys_user_group] table.</td>
</tr>
<tr>
<td>business_duration</td>
<td>Length in calendar work hours, work days, and work weeks that it took to complete the case.</td>
</tr>
<tr>
<td>business_impact</td>
<td>Impact of the issue on the associated customer.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>business_service</td>
<td>Sys_id of the service record associated with the case. Located in the Service [cmdb_ci_service] table. Data type: String</td>
</tr>
<tr>
<td>case</td>
<td>Case short description and case number. Data type: String Maximum length: 300</td>
</tr>
<tr>
<td>category</td>
<td>Case category. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Question</td>
</tr>
<tr>
<td></td>
<td>• 1: Issue</td>
</tr>
<tr>
<td></td>
<td>• 2: Feature</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer) Default: 1</td>
</tr>
<tr>
<td>cause</td>
<td>Details about the cause of the problem. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>caused_by</td>
<td>Sys_id of the change request that caused the case to be created. Located in the Change Request [change_request] table. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>change</td>
<td>Sys_id of the change request that caused the case to be created. Located in the Change Request [change_request] table. Data type: String</td>
</tr>
</tbody>
</table>
| child_case_creation_progress  | Flag that indicates whether the case is a child case that was created from a major case. Possible values:  
• true: Child case created from a major case  
• false: Not a child case  
Data type: Boolean                                                                |
<p>| closed_at                     | Date and time that the case was closed. Data type: String                                                                                                                                                   |
| closed_by                     | Sys_id of the user that closed the case. Located in the User [sys_user] table. Data type: String                                                                                                           |
| close_notes                   | Notes made when the case was closed. Data type: String                                                                                                                                                     |
| cmdb_ci                       | Sys_id of the configuration item associated with the case. Located in the Configuration Item [cmdb_ci] table. Data type: String                                                                          |
| comments                      | Additional comments on the case. Data type: String                                                                                                                                                         |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>comments_and_work_notes</td>
<td>Comments and work notes entered for the case. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>company</td>
<td>Sys_id of the company associated with the case. Located in the Company [core_company] table. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>consumer</td>
<td>Business-to-consumer cases. Sys_id of the person to contact with regards to this case. Located in the Consumer [csm_consumer] table. Data type: String</td>
</tr>
<tr>
<td>consumer.active</td>
<td>Flag that indicates whether the consumer is active. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Consumer active</td>
</tr>
<tr>
<td></td>
<td>• false: Consumer de-activated</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td>consumer.business_phone</td>
<td>Business phone number of the consumer. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.city</td>
<td>City in which the consumer resides. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consumer.country</td>
<td>Country in which the consumer resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.date_format</td>
<td>Format in which to display dates.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• dd-mm-yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd/mm/yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd.mm.yyyy</td>
</tr>
<tr>
<td></td>
<td>• mm-dd-yyyy</td>
</tr>
<tr>
<td></td>
<td>• yyyy-mm-dd</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system date format)</td>
</tr>
<tr>
<td>consumer.email</td>
<td>Email address of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>consumer.fax</td>
<td>Fax number of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.first_name</td>
<td>Consumer first name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>consumer.gender</td>
<td>Gender of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.home_phone</td>
<td>Home phone number of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consumer.household</td>
<td>Sys_id of the record that describes the household characteristics. Located in the Household [csm_household] table.</td>
</tr>
<tr>
<td>consumer.last_name</td>
<td>Consumer last name.</td>
</tr>
<tr>
<td>consumer.middle_name</td>
<td>Consumer middle name.</td>
</tr>
<tr>
<td>consumer.mobile_phone</td>
<td>Consumer mobile phone</td>
</tr>
<tr>
<td>consumer.name</td>
<td>Consumer full name; first_name+middle_name+last_name.</td>
</tr>
<tr>
<td>consumer.notes</td>
<td>Notes on consumer.</td>
</tr>
<tr>
<td>consumer.notification</td>
<td>Indicates whether the consumer should receive notifications. Valid values: 1: Disabled, 2: Enabled</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consumer.number</td>
<td>Unique number associated with the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: 2</td>
</tr>
<tr>
<td>consumer.photo</td>
<td>Photo of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: Image</td>
</tr>
<tr>
<td>consumer.preferred_language</td>
<td>Consumer primary language.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 3</td>
</tr>
<tr>
<td>consumer.prefix</td>
<td>Consumer name prefix such as, Dr., Mr., Mrs., or Ms.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.primary</td>
<td>Flag that indicates whether the primary consumer.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Primary consumer</td>
</tr>
<tr>
<td></td>
<td>• false: Not primary consumer</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>consumer.state</td>
<td>State in which the consumer resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>consumer.street</td>
<td>Consumer street address.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consumer.suffix</td>
<td>Consumer name suffix such as Jr., Sr., or II. Data type: String</td>
</tr>
<tr>
<td>consumer.sys_created_by</td>
<td>User that created the consumer record. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.sys_created_on</td>
<td>Date and time the consumer record was originally created. Data type: String</td>
</tr>
<tr>
<td>consumer.sys_domain</td>
<td>ServiceNow domain in which the consumer information resides. Data type: String</td>
</tr>
<tr>
<td>consumer.sys_id</td>
<td>Unique identifier for the consumer. Data type: String</td>
</tr>
<tr>
<td>consumer.sys_mod_count</td>
<td>Number of times that the associated consumer information has been modified. Data type: Integer</td>
</tr>
<tr>
<td>consumer.sys_updated_by</td>
<td>User that last updated consumer information. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>consumer.sys_updated_on</td>
<td>Date and time when the consumer information was last updated. Data type: String</td>
</tr>
<tr>
<td>consumer.time_format</td>
<td>Format in which to display time. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• hh.mm.ss a: hh:mm:ss (12 hour)</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss (24 hour)</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consumer.time_zone</td>
<td>Consumer time zone, such as Canada/Central or US Eastern Time. Data type: String</td>
</tr>
<tr>
<td>consumer.title</td>
<td>Consumer business title. Manager, Software Developer, Contractor. Data type: String</td>
</tr>
<tr>
<td>consumer.user</td>
<td>Sys_id of the consumer user. Located in the Consumer User [csm_consumer_user] table. Data type: String</td>
</tr>
<tr>
<td>consumer.zip</td>
<td>Consumer zip code. Data type: String</td>
</tr>
<tr>
<td>contact</td>
<td>Business-to-business based cases only. Sys_id of the person to contact regarding this case. Located in the Contact [customer_contact] table. Data type: String</td>
</tr>
<tr>
<td>contact</td>
<td>Array of contact parameters for business-to-business cases.</td>
</tr>
<tr>
<td>contact.account</td>
<td>Sys_id of the account to which the contact is associated.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>account</td>
<td>[customer_account] table.</td>
</tr>
<tr>
<td>contact.active</td>
<td>Flag that indicates whether the contact is active within the system.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Contact is active</td>
</tr>
<tr>
<td></td>
<td>• false: Contact is inactive</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td>contact.agent_status</td>
<td>Status of the agent.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Off work</td>
</tr>
<tr>
<td></td>
<td>• On break</td>
</tr>
<tr>
<td></td>
<td>• On route</td>
</tr>
<tr>
<td></td>
<td>• On site</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.building</td>
<td>Sys_id of the record that describes the building in which the contact resides; Building [cmn_building] table.</td>
</tr>
<tr>
<td>contact.calendar_integration</td>
<td>Calendar application that the contact uses.</td>
</tr>
<tr>
<td></td>
<td>• 1: Outlook</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 1</td>
</tr>
<tr>
<td>contact.city</td>
<td>City in which the contact resides.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.company</td>
<td>Sys_id of the company with which the contact is associated; Company [core_company] table. Data type: String</td>
</tr>
<tr>
<td>contact.cost_center</td>
<td>Sys_id of the cost center associated with the contact; Cost Center [cmn_cost_center] table. Data type: String</td>
</tr>
<tr>
<td>contact.country</td>
<td>Country code of the country in which the contact resides. Data type: String Maximum length: 3</td>
</tr>
</tbody>
</table>
| contact.date_format      | Format in which to display dates to contacts. Valid values:  
- dd/mm/yyyy  
- dd-mm-yyyy  
- dd.mm.yyyy  
- mm-dd-yyyy  
- yyyy-mm-dd Data type: String Maximum length: 40 Default: blank (system date format) |
<p>| contact.default_perspective | Sys_id of the default perspective for the contact. Located in the Menu List [sys_perspective] table. Data type: String |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contact.department</td>
<td>Sys_id of the department associated with the contact. Located in the Department [cmn_department] table.</td>
</tr>
<tr>
<td>contact.edu_status</td>
<td>Education status of the associated contact. Data type: String Maximum length: 40 Default: faculty</td>
</tr>
<tr>
<td>contact.email</td>
<td>Contact email address. Data type: String</td>
</tr>
<tr>
<td>contact.employee_number</td>
<td>Contact employee number. Data type: String</td>
</tr>
<tr>
<td>contact.enable_multifactor_authn</td>
<td>Flag that indicates whether multifactor authorization is required for the contact to log in to the service portal. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Multifactor authorization enabled</td>
</tr>
<tr>
<td></td>
<td>• false: Multifactor authorization disabled</td>
</tr>
<tr>
<td>contact.failed_attempts</td>
<td>Number of failed log in attempts. Data type: Number (Integer)</td>
</tr>
<tr>
<td>contact.first_name</td>
<td>Contact first name. Data type: String Maximum length: 50</td>
</tr>
<tr>
<td>contact.gender</td>
<td>Contact gender.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| contact.geolocation_tracked | Flag that indicates whether the contact location is obtained through geotracking. Possible values:  
- true: Contact location obtained through geotracking  
- false: Contact location not obtained through geotracking |
| Data type: Boolean  
Default value: false |
| contact.home_phone | Contact home phone number. |
| Data type: String  
Maximum length: 40 |
| contact.internal_integration_user | Flag that indicates whether the contact is an internal integration user. Possible values:  
- true: Internal integration user  
- false: Other type of user |
| Data type: Boolean  
Default: false |
| contact.introduction | Introduction |
| Data type: String  
Maximum length: 40 |
<p>| contact.last_login | Date on which the contact last logged into the system. |
| Data type: String (Date) |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>contact.last_login_device</code></td>
<td>Device the consumer used the last time they logged into the system. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td><code>contact.last_login_time</code></td>
<td>Date and time the contact logged in to the system. Data type: String (Date/time)</td>
</tr>
<tr>
<td><code>contact.last_name</code></td>
<td>Contact last name. Data type: String Maximum length: 50</td>
</tr>
<tr>
<td><code>contact.last_position_update</code></td>
<td>Date and time the last position was updated. Data type: String (Date/time)</td>
</tr>
<tr>
<td><code>contact.latitude</code></td>
<td>Latitude coordinate of the contact. Data type: Number (Floating point) Maximum length: 40</td>
</tr>
<tr>
<td><code>contact.ldap_server</code></td>
<td>Sys_id of the LDAP server by the contact to last log in to the system; LDAP Server [ldap_server_config] table. Data type: String</td>
</tr>
<tr>
<td><code>contact.location</code></td>
<td>Sys_id of the record that describes the location of the contact; Location [cmn_location] table. Data type: String</td>
</tr>
<tr>
<td><code>contact.locked_out</code></td>
<td>Flag that indicates if the contact is locked-out. Possible values:</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
|                        | • true: Contact locked-out  
|                        | • false: Contact not locked-out  
|                        | Data type: Boolean  
|                        | Default: false                                                             |
| contact.longitude       | Longitude coordinate of the contact.  
|                        | Data type: Number (Floating point)  
|                        | Maximum length: 40                                                          |
| contact.manager         | Sys_id of the record that describes the direct supervisor of the contact; User [sys_user]  
|                        | Data type: String                                                           |
| contact.middle_name     | Contact middle name.  
|                        | Data type: Number (Floating point)  
|                        | Maximum length: 50                                                          |
| contact.mobile_phone    | Contact mobile phone.  
|                        | Data type: String                                                           |
|                        | Maximum length: 40                                                          |
| contact.name            | Contact full name.  
|                        | Data type: String                                                           |
|                        | Maximum length: 151                                                        |
| contact.notification    | Indicates whether the contact should receive notifications.  
|                        | Valid values:  
|                        | • 1: Disabled  
|                        | • 2: Enabled  
|                        | Data type: Number (Integer)  
|                        | Default: 2                                                                 |

© 2021 ServiceNow, Inc. All rights reserved.  
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contact.on_schedule</td>
<td>Indicates the timeliness of dispatched service personnel.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• Ahead: Ahead of schedule.</td>
</tr>
<tr>
<td></td>
<td>• behind_less30: Behind schedule, but less than 30 minutes.</td>
</tr>
<tr>
<td></td>
<td>• behind_30to60: Behind schedule between 30 and 60 minutes.</td>
</tr>
<tr>
<td></td>
<td>• behind_more60: Behind schedule more than 60 minutes.</td>
</tr>
<tr>
<td></td>
<td>• on_time: On schedule.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.phone</td>
<td>Contact business phone number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.photo</td>
<td>Photo image of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: Image</td>
</tr>
<tr>
<td>contact.preferred_language</td>
<td>Country code of the contact's primary language.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 3</td>
</tr>
<tr>
<td>contact.roles</td>
<td>List of user roles associated with the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.schedule</td>
<td>Sys_id of the record that describes the work schedule.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>for the associated contact; Schedule [cmn_schedule] table. Data type: String</td>
</tr>
<tr>
<td>contact.source</td>
<td>Source of the contact. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>contact.state</td>
<td>State in which the contact resides. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>contact.street</td>
<td>Contact street address. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>contact.sys_class_name</td>
<td>Table that contains the contact record. Data type: String Maximum length: 80</td>
</tr>
<tr>
<td>contact.sys_created_by</td>
<td>User that originally created the associated contact record. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>contact.sys_created_on</td>
<td>Data and time the associated contact was originally created. Data type: String (Date)</td>
</tr>
<tr>
<td>contact.sys_domain</td>
<td>ServiceNow instance of the associated contact. Data type: String</td>
</tr>
<tr>
<td>contact.sys_domain_path</td>
<td>Contact record domain path. Data type: String Maximum length: 255 Default: / (global)</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contact.sys_id</td>
<td>Unique identifier for the associated contact record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>contact.sys_mod_count</td>
<td>Number of times that the associated contact record has been modified.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td>contact.sys_tags</td>
<td>System tags for the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>contact.sys_updated_by</td>
<td>User that last updated the associated contact information.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>contact.sys_updated_on</td>
<td>Data and time the associated contact information was last updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date/time)</td>
</tr>
<tr>
<td>contact.time_format</td>
<td>Format in which to display time.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>• HH:mm:ss: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>• HH:mm:ss: hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: Blank (system time format)</td>
</tr>
<tr>
<td>contact.time_sheet_policy</td>
<td>Sys_id of the record that contains the time sheet policy for the associated contact; Time Sheet Policy [time_sheet_policy]</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>contact.time_zone</td>
<td>Time zone in which the contact resides, such as Canada/Central or US/Eastern. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>contact.title</td>
<td>Contact business title such as Manager, Software Developer, or Contractor. Data type: String Maximum length: 60</td>
</tr>
<tr>
<td>contact.user_name</td>
<td>Contact user ID. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>contact.vip</td>
<td>Flag that indicates whether the associated contact has VIP status. Possible values: * true: VIP * false: Not VIP Data type: Boolean Default: false</td>
</tr>
<tr>
<td>contact.web_service_access_only</td>
<td>Flag that indicates whether the contact can only access services through the web. Possible values: * true: Web access only * false: Access through all available methods Data type: Boolean Default: false</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| contact.zip        | Contact zip code.  
Data type: String  
Maximum length: 40 |
| contact_local_time | Contact local time.  
Data type: String  
Maximum length: 70 |
| contact_time_zone  | Time zone of the contact associated with the case.  
Data type: String  
Maximum length: 40 |
| contact_type       | Method in which the case was initially reported.  
Possible values:  
• chat  
• email  
• phone  
• social  
• web  
Data type: String  
Maximum length: 40 |
| contract           | Sys_id of the contract associated with the case. Located in the Contract [ast_contract] table. This contract contains information about the type of support that is provided to the company associated to the case. A contract can include specific assets that are covered.  
A contract can also include multiple service entitlements and SLAs. |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| correlation_display | Correlation display.  
Data type: String  
Maximum length: 100 |
| correlation_id   | Correlation identifier.  
Data type: String  
Maximum length: 100 |
Data type: String |
Data type: String |
| description      | Detailed description of the problem associated with the case.  
Data type: String  
Maximum length: 4,000 |
| due_date         | Date that the case is due to be closed.  
Data type: String |
Entitlements define the type of support that a customer receives as well as the communication channels the entitlement can be assigned to.  
Data type: String |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| escalation       | Current escalation level. Possible values:  
|                  | • 0: Normal  
|                  | • 1: Moderate  
|                  | • 2: High  
|                  | • 3: Overdue  
|                  | Data type: Number (Integer)  
|                  | Default: 0 |
| expected_start   | Date and time when work is scheduled to begin on the case.  
|                  | Data type: String |
| first_response_time | Date and time when the first action was taken on the case.  
|                  | Data type: String |
| follow_the_sun   | Flag that indicates whether the case should be handed-off for global follow-up.  
|                  | If a customer enters additional comments on a Priority 1 - Critical or a Priority 2 - High case, or if the case is escalated, the flag is automatically set to true.  
|                  | Possible values:  
|                  | • true: Case should be handed-off for global follow-up  
|                  | • false: Case should not be handed-off for global follow-up  
<p>|                  | Data type: Boolean |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>follow_up</td>
<td>Date and time of the next follow-up action. Data type: String</td>
</tr>
<tr>
<td>group_list</td>
<td>List of sys_ids of the group records associated with the case. Located in the Group [sys_user_group] table. Data type: Array</td>
</tr>
<tr>
<td>impact</td>
<td>Impact on customer. Possible values: • 1: High • 2: Medium • 3: Low Data type: Number (Integer) Default: 3</td>
</tr>
<tr>
<td>knowledge</td>
<td>Flag that indicates if there is a knowledge base article available for the specified issue. Possible values: • true: Knowledge base article is available for this issue • false: Knowledge base article is not available for this issue Data type: Boolean Default: false</td>
</tr>
<tr>
<td>location</td>
<td>Sys_id of the record describing the company location. Located in the Location [cmn_location] table. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| made_sla         | Flag that indicates whether the case was resolved in alignment with the associated service level agreement. Possible values:  
|                  | • true: Case was resolved in alignment with SLA  
|                  | • false: Case was not resolved according to the SLA                          |
|                  | Data type: Boolean                                                          |
|                  | Default: true                                                               |
| major_case_state | Current state of the major case. Possible values:  
|                  | • accepted: Initial state when a manager creates a major case or when a manager promotes a candidate case.  
|                  | • canceled: Case is canceled.                                                |
|                  | • proposed: Initial state when an agent or manager creates or proposes a candidate case.  
|                  | • rejected: Manager rejects a candidate case.                                 |
|                  | Data type: String                                                            |
|                  | Maximum length: 40                                                           |
| needs_attention  | Flag that indicates whether the case needs attention. Possible values:  
|                  | • true: Case needs additional attention  
<p>|                  | • false: Case does not need additional attention                             |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
|               | Data type: Boolean  
|               | Default: false |
| notes_to_comments | Flag that indicates whether to add the resolution notes to comments.  
|               | Possible values:  
|               | • true: Resolutions notes, when added, are also added to comments  
|               | • false: Resolution notes in comments are not required |
|               | Data type: Boolean |
| notify        | Method to use to notify contact/consumer.  
|               | Possible values:  
|               | • 1: Do not notify  
|               | • 2: Send email  
|               | • 3: Telephone  
|               | Data type: Number (Integer)  
|               | Default: 1 |
| number        | Case number.  
|               | Data type: String  
|               | Maximum length: 40 |
| opened_at     | Date and time that the case was opened.  
|               | Data type: String |
| opened_by     | Sys_id of the person that opened the case. Located in the User [sys_user] table.  
<p>|               | Data type: String |
| order         | Order of the case. |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent</td>
<td>Sys_id of the parent case to which this case (child) is associated. Located in the [task] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>partner</td>
<td>Sys_id of the partner associated with the case. Located in the Account [customer_account] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>partner_contact</td>
<td>Sys_id of the partner contact associated with the case. Located in the Contact [customer_contact] table.</td>
</tr>
<tr>
<td>priority</td>
<td>Priority of the case. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Critical</td>
</tr>
<tr>
<td></td>
<td>• 2: High</td>
</tr>
<tr>
<td></td>
<td>• 3: Moderate</td>
</tr>
<tr>
<td></td>
<td>• 4: Low</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 4</td>
</tr>
<tr>
<td>probable_cause</td>
<td>Possible cause of the issue associated with the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>problem</td>
<td>Sys_id of the issue that the customer is encountering. Located in the Problem [problem] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>product</strong></td>
<td>Sys_id of the product model of the asset associated to the case. Located in the Product Model [cmdb_model] table. A model is a specific version or configuration of an asset (for example, Apple Mac Book Pro). Data type: String</td>
</tr>
<tr>
<td><strong>reassignment_count</strong></td>
<td>Number of times that the case was reassigned to a person who is responsible for moving the case forward. Data type: Number (Integer) Default: 0</td>
</tr>
<tr>
<td><strong>recipient_list</strong></td>
<td>Sys_id of the record that contains the list of recipients for communications about this case. Located in the Recipient List [sn_publications_recipients_list] table. Data type: String</td>
</tr>
<tr>
<td><strong>rejection_goto</strong></td>
<td>Sys_id of the task to execute if the case is rejected. Located in the Task [task] table. Data type: String</td>
</tr>
<tr>
<td><strong>resolution_code</strong></td>
<td>Resolution state for the case, such as &quot;Solved - Fixed by Support/Guidance provided&quot;. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td><strong>resolved_at</strong></td>
<td>Date and time that the case was resolved. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>resolved_by</td>
<td>Sys_id of the person that resolved the case. Located in the [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result</td>
<td>Array of objects in which each object describes a single value for the requested field.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>&quot;result&quot;: [</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;label&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;value&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>result.label</td>
<td>Display value for the field.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.value</td>
<td>Field value.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>short_description</td>
<td>Concise description of the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160</td>
</tr>
<tr>
<td>skills</td>
<td>List of the unique identifiers (sys_id) of the skills needed to complete the case. Located in the Skill [cmn_skill] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sla_due</td>
<td>Date/time at which the case must be closed based on the associated service level agreement.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile</td>
<td>Array of objects that contain details for a specific social media profile.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.account</td>
<td>Unique identifier of the record containing account information for the social media profile. Located in the Account [customer_account] table. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel</td>
<td>Details on the associated social media profile channel. Data type: Object</td>
</tr>
</tbody>
</table>

```json
*sn_app_cs_social_social_profile:
 {
  "account": "String",
  "channel": [Array],
  "contact": {Object},
  "consumer": {Object},
  "created_on": "String",
  "profile": "String",
  "profile_url": "String",
  "social_id": "String",
  "source": "String",
  "sys_created_by": "String",
  "sys_created_on": "String",
  "sys_id": "String",
  "sys_mod_count": Number,
  "sys_tags": "String",
  "sys_updated_by": "String",
  "sys_updated_on": "String"
 }
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.channel.name</td>
<td>Name of the social media channel. Data type: String Maximum length: 100</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_class_name</td>
<td>Table that contains the social media channel record. Data type: String Maximum length: 80</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_created_by</td>
<td>Person that created the social media channel. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_created_on</td>
<td>Date and time the social profile was created. Data type: Date/time Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_id</td>
<td>Unique identifier of the associated social media channel. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_mod_count</td>
<td>Number of times that information was modified for the associated social media profile channel. Data type: Integer</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.channel.sys_name</td>
<td>System name of channel. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>sn_app_cs_social_social_profile.channel.sys_package</code></td>
<td>Unique identifier of the record that contains information about the package associated with the profile; Package [sys_package] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Reference</td>
</tr>
<tr>
<td><code>sn_app_cs_social_social_profile.channel.sys_policy</code></td>
<td>System protection policy.</td>
</tr>
<tr>
<td></td>
<td>Possible values: protected, read</td>
</tr>
<tr>
<td></td>
<td>Data type: String Maximum length: 40</td>
</tr>
<tr>
<td><code>sn_app_cs_social_social_profile.channel.sys_scope</code></td>
<td>Unique identifier of the record that contains information about the scope of the social profile; Application [sys_scope] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Reference</td>
</tr>
<tr>
<td><code>sn_app_cs_social_social_profile.channel.sys_tags</code></td>
<td>System tags associated with the channel.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>sn_app_cs_social_social_profile.channel.sys_update_name</code></td>
<td>Name of the person that last updated the social media profile channel.</td>
</tr>
<tr>
<td></td>
<td>Data type: String Maximum length: 250</td>
</tr>
<tr>
<td><code>sn_app_cs_social_social_profile.channel.sys_updated_by</code></td>
<td>User that last updated the social media profile channel.</td>
</tr>
<tr>
<td></td>
<td>Data type: String Maximum length: 40</td>
</tr>
<tr>
<td><code>sn_app_cs_social_social_profile.channel.sys_updated_on</code></td>
<td>Date and time the social profile channel was last updated.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| sn_app_cs_social_social_profile.consumer | Data type: Date/time  
Maximum length: 40 |

For business-to-consumer cases.  
Details about the consumer associated with the case.  
Data type: Object

```
"consumer": {  
  "active": Boolean,  
  "business_phone": "String",  
  "city": "String",  
  "country": "String",  
  "date_format": "String",  
  "email": "String",  
  "fax": "String",  
  "first_name": "String",  
  "gender": "String",  
  "home_phone": "String",  
  "household": "String",  
  "last_name": "String",  
  "middle_name": "String",  
  "mobile_phone": "String",  
  "name": "String",  
  "notes": "String",  
  "notification": Number,  
  "number": "String",  
  "photo": Image,  
  "preferred_language": "String",  
  "prefix": "String",  
  "primary": Boolean,  
  "state": "String",  
  "street": "String",  
  "suffix": "String",  
  "sys_created_by": "String",  
  "sys_created_on": "String",  
  "sys_domain": "String",  
  "sys_id": "String",  
  "sys_mod_count": Number,  
  "sys_tags": "String",  
  "sys_updated_by": "String",  
  "sys_updated_on": "String",  
  "time_format": "String"  
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| sn_app_cs_social_social_profile.consumer.active | Flag that indicates whether the consumer is active. Possible values:  
  • true: Consumer active  
  • false: Consumer de-activated  
  Data type: Boolean  
  Default: true |
| sn_app_cs_social_social_profile.consumer.business_phone | Business phone number of the consumer.  
  Data type: String  
  Maximum length: 40 |
| sn_app_cs_social_social_profile.consumer.city | City in which the consumer resides.  
  Data type: String  
  Maximum length: 100 |
| sn_app_cs_social_social_profile.consumer.country | Country in which the consumer resides.  
  Data type: String  
  Maximum length: 40 |
| sn_app_cs_social_social_profile.consumer.date_format | Format in which to display dates.  
  Valid values:  
  • dd-mm-yyyy  
  • dd/mm/yyyy  
  • dd.mm.yyyy  
  • mm-dd-yyyy  
  • yyyy-mm-dd |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.consumer.email</td>
<td>Email address of the consumer.</td>
</tr>
<tr>
<td>Data type: String&lt;br&gt;Maximum length: 100</td>
<td></td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.fax</td>
<td>Fax number of the consumer.</td>
</tr>
<tr>
<td>Data type: String&lt;br&gt;Maximum length: 40</td>
<td></td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.first_name</td>
<td>Consumer first name.</td>
</tr>
<tr>
<td>Data type: String&lt;br&gt;Maximum length: 50</td>
<td></td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.gender</td>
<td>Gender of the consumer.</td>
</tr>
<tr>
<td>Data type: String&lt;br&gt;Maximum length: 40</td>
<td></td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.home_phone</td>
<td>Home phone number of the consumer.</td>
</tr>
<tr>
<td>Data type: String&lt;br&gt;Maximum length: 40</td>
<td></td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.household</td>
<td>Sys_id of the record that describes the household characteristics. Located in the Household [csm_household] table.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.last_name</td>
<td>Consumer last name.</td>
</tr>
<tr>
<td>Data type: String&lt;br&gt;Maximum length: 50</td>
<td></td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.middle_name</td>
<td>Consumer middle name.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.mobile_phone</td>
<td>Consumer mobile phone number. Data type: String, Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.name</td>
<td>Consumer full name; first_name +middle_name+last_name. Data type: String, Maximum length: 152</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.notes</td>
<td>Notes on consumer. Data type: String, Maximum length: 4,000</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.notification</td>
<td>Indicates whether the consumer should receive notifications. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Disabled</td>
</tr>
<tr>
<td></td>
<td>• 2: Enabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Integer, Maximum length: 40, Default: 2</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.number</td>
<td>Unique number associated with the consumer. Data type: String, Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.photo</td>
<td>Photo of the consumer. Data type: Image</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.preferred_language</td>
<td>Consumer primary language. Data type: String, Maximum length: 3</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.prefix</td>
<td>Consumer name prefix; Dr., Mr., Mrs., or Ms.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.primary</td>
<td>Flag that indicates whether this is the primary consumer. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Primary consumer</td>
</tr>
<tr>
<td></td>
<td>• false: Not primary consumer</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.state</td>
<td>State in which the consumer resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.street</td>
<td>Consumer street address</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.suffix</td>
<td>Consumer name suffix such as Jr., Sr., or II.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_created_by</td>
<td>User that created the consumer record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_created_on</td>
<td>Date and time the consumer record was originally created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_domain</td>
<td>ServiceNow domain in which the consumer information resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_id</td>
<td>Unique identifier for the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_mod_count</td>
<td>Number of times that the associated consumer information has been modified. Data type: Integer</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_tags</td>
<td>Consumer system tags.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_updated_by</td>
<td>User that last updated the consumer information. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.sys_updated_on</td>
<td>Date and time when the consumer information was last updated. Data type: String</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.consumer.time_format                    | Format in which to display time. Valid values:  
  • hh.mm.ss a: hh.mm.ss  
  • hh:mm:ss a: hh:mm:ss  
  • HH.mm.ss: hh.mm.ss  
  • HH:mm:ss: hh:mm:ss  
  Data type: String Maximum length: 40 Default: blank (system time format) |
<p>| sn_app_cs_social_social_profile.consumer.time_zone                      | Consumer time zone, such as Canada/Central or US/Eastern. Data type: String Maximum length: 40                                               |
| sn_app_cs_social_social_profile.consumer.title                          | Consumer business title such as Manager, Software Developer, or Contractor.                                                                |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.consumer.user</td>
<td>Sys_id of the consumer user. Located in the Consumer User [csm_consumer_user] table.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.consumer.zip</td>
<td>Consumer zip code.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact</td>
<td>Unique identifier of the record containing details about the social media profile that belongs to the contact associated with the case. Located in the Contact [customer_contact] table.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.account</td>
<td>Sys_id of the account with which the contact is associated. Located in the Account [customer_account] table.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.active</td>
<td>Flag that indicates whether the contact is active within the system. Possible values: true: Contact is active false: Contact is inactive</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.agent_status</td>
<td>Status of the agent. Possible values:</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>• Off work</td>
<td>Data type: String</td>
</tr>
<tr>
<td>• On break</td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>• On route</td>
<td></td>
</tr>
<tr>
<td>• On site</td>
<td></td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.building</td>
<td>Sys_id of the record that describes the building in which the contact resides; Building [cmn_building] table.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.calendar_integration</td>
<td>Calendar application that the contact uses.</td>
</tr>
<tr>
<td></td>
<td>• 1: Outlook</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.city</td>
<td>City in which the contact resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.company</td>
<td>Sys_id of the company to which the contact is associated; Company [core_company] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.cost_center</td>
<td>Sys_id of the cost center associated with the contact; Cost Center [cmn_cost_center] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.country</td>
<td>Country code of the country in which the contact resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.contact.date_format | Format in which to display dates to contacts. Valid values:  
- dd/mm/yyyy  
- dd-mm-yyyy  
- dd.mm.yyyy  
- mm-dd-yyyy  
- yyyy-mm-dd  
Data type: String  
Maximum length: 40  
Default: blank (system date format) |
| sn_app_cs_social_social_profile.contact.default_perspective | Sys_id of the default perspective for the contact. Located in the Menu List [sys_perspective] table.  
Data type: String |
| sn_app_cs_social_social_profile.contact.department | Sys_id of the department associated with the contact. Located in the Department [cmn_department] table.  
Data type: String |
| sn_app_cs_social_social_profile.contact.edu_status | Education status of the associated contact.  
Data type: String  
Maximum length: 40  
Default: faculty |
| sn_app_cs_social_social_profile.contact.email | Contact email address.  
Data type: String |
| sn_app_cs_social_social_profile.contact.employee_number | Contact employee number.  
Data type: String |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.contact.enable_multifactor_authn</td>
<td>Flag that indicates whether multifactor authorization is required for the contact to log in to the service portal. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Multifactor authorization enabled</td>
</tr>
<tr>
<td></td>
<td>• false: Multifactor authorization disabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.failed_attempts</td>
<td>Number of failed log in attempts.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.first_name</td>
<td>Contact first name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.gender</td>
<td>Contact gender.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.geolocation_tracked</td>
<td>Flag that indicates whether the contact location is obtained through geotracking. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Contact location obtained through geotracking</td>
</tr>
<tr>
<td></td>
<td>• false: Contact location not obtained through geotracking</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default value: false</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.home_phone</td>
<td>Contact home phone number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
| © 2021 ServiceNow, Inc. All rights reserved.                           | ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| sn_app_cs_social_social_profile.contact.internal_integration_user | Flag that indicates whether the contact is an internal integration user. Possible values:  
- true: Internal integration user  
- false: Other type of user  
Data type: Boolean  
Default: false |
| sn_app_cs_social_social_profile.contact.introduction | Introduction  
Data type: String  
Maximum length: 40 |
| sn_app_cs_social_social_profile.contact.last_login | Date on which the contact logged into the system.  
Data type: String (Date) |
| sn_app_cs_social_social_profile.contact.last_login_device | Device the consumer used the last time they logged in to the system.  
Data type: String  
Maximum length: 40 |
| sn_app_cs_social_social_profile.contact.last_login_time | Date and time the contact logged into the system.  
Data type: String (Date/time) |
| sn_app_cs_social_social_profile.contact.last_name | Contact last name.  
Data type: String  
Maximum length: 50 |
| sn_app_cs_social_social_profile.contact.last_position_update | Date and time the last position was updated.  
Data type: String (Date/time) |
<p>| sn_app_cs_social_social_profile.contact.latitude | Latitude coordinate of the contact. |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.contact.ldap_server</td>
<td>Sys_id of the LDAP server used by the contact to last log in to the system; LDAP Server [ldap_server_config] table. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.location</td>
<td>Sys_id of the record that describes the location of the contact; Location [cmn_location] table. Data type: String</td>
</tr>
</tbody>
</table>
| sn_app_cs_social_social_profile.contact.locked_out | Flag that indicates if the contact is locked-out. Possible values:  
  • true: Contact locked-out  
  • false: Contact not locked-out  
Data type: Boolean  
Default: false |
| sn_app_cs_social_social_profile.contact.longitude | Longitude coordinate of the contact. Data type: Number (Floating point)  
Maximum length: 40 |
<p>| sn_app_cs_social_social_profile.contact.manager | Sys_id of the record that describes the direct supervisor of the contact; User [sys_user] table. Data type: String |
| sn_app_cs_social_social_profile.contact.middle_name | Contact middle name. Data type: Number (Floating point) |</p>
<table>
<thead>
<tr>
<th><strong>Element</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.contact.mobile_phone</td>
<td>Contact mobile phone number. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.name</td>
<td>Contact full name. Data type: String Maximum length: 151</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.notification</td>
<td>Indicates whether the contact should receive notifications. Valid values: • 1: Disabled • 2: Enabled Data type: Number (Integer) Default: 2</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.on_schedule</td>
<td>Indicates the timeliness of dispatched service personnel. Valid values: • Ahead: Ahead of schedule • behind_less30: Behind schedule, but less than 30 minutes. • behind_30to60: Behind schedule between 30 and 60 minutes. • behind_more60: Behind schedule more than 60 minutes. • on_time: On schedule Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.phone</td>
<td>Contact business phone number. Data type: String. Maximum length: 40.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.photo</td>
<td>Photo image of the contact. Data type: Image.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.preferred_language</td>
<td>Country code of the contact's primary language. Data type: String. Maximum length: 3.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.roles</td>
<td>List of user roles associated with the contact. Data type: String. Maximum length: 40.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.schedule</td>
<td>Sys_id of the record that describes the work schedule for the associated contact. Schedule [cmn_schedule] table. Data type: String.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.source</td>
<td>Source of the contact. Data type: String. Maximum length: 255.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.state</td>
<td>State in which the contact resides. Data type: String. Maximum length: 40.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.street</td>
<td>Contact street address. Data type: String. Maximum length: 255.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_class_name</td>
<td>Table that contains the contact record. Data type: String.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_created_by</td>
<td>User that originally created the associated contact record. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_created_on</td>
<td>Data and time the associated contact was originally created. Data type: String (Date/Time)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_domain</td>
<td>ServiceNow instance domain of the associated contact record. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_domain_path</td>
<td>Contact record domain path. Data type: String Maximum length: 255 Default: / (global)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_id</td>
<td>Unique identifier for the associated contact record. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_mod_count</td>
<td>Number of times that the associated contact record has been modified. Data type: Number (Integer)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_tags</td>
<td>Contact system tags.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_updated_by</td>
<td>User that last updated the associated contact information. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.sys_updated_on</td>
<td>Data and time the associated contact information was updated. Data type: String (Date/Time)</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.time_format</td>
<td>Format in which to display time.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.contact.time_sheet_policy</td>
<td>Sys_id of the record that contains the time sheet policy for the associated contact; Time Sheet Policy [time_sheet_policy] table. Data type: String</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.time_zone</td>
<td>Time zone in which the contact resides, such as Canada/Central or US/Eastern. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.title</td>
<td>Contact business title such as Manager, Software Developer, or Contractor. Data type: String Maximum length: 60</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.user_name</td>
<td>Contact user ID. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.vip</td>
<td>Flag that indicates whether the associated contact has VIP status. Possible values:</td>
</tr>
</tbody>
</table>

Valid values:
- hh:mm:ss a: hh:mm:ss (12 hour)
- hh:mm:ss a: hh:mm:ss (12 hour)
- HH:mm:ss: hh:mm:ss (24 hour)
- HH:mm:ss: hh:mm:ss (24 hour)

Data type: String
Maximum length: 40
Default: Blank (system format)
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn_app_cs_social_social_profile.contact.web_service_access_only</td>
<td>Flag that indicates whether the contact can only access services through the web. Possible values: • true: Web access only • false: Access through all available methods</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.contact.zip</td>
<td>Contact zip code. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.created_on</td>
<td>Date and time the associated social media profile was created.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.profile</td>
<td>Social profile. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.profile_url</td>
<td>URL to use to access the social media profile. Data type: String Maximum length: 1,024</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.social_id</td>
<td>Unique social media account provider identifier for the associated social media account. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.source</td>
<td>Source of the social profile. Data type: String. Maximum length: 255.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_created_by</td>
<td>User that initially created the social media profile. Data type: String. Maximum length: 40.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_created_on</td>
<td>Date and time the social media profile was initially created. Data type: String (Date/Time).</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_id</td>
<td>Unique identifier for the social media profile. Data type: String.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_mod_count</td>
<td>Number of times that information was modified for the associated social media profile. Data type: Number (Integer).</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_tags</td>
<td>Profile system tags.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_updated_by</td>
<td>User that initially created the social media profile. Data type: String. Maximum length: 40.</td>
</tr>
<tr>
<td>sn_app_cs_social_social_profile.sys_updated_on</td>
<td>User that initially created the social media profile. Data type: String. Maximum length: 40.</td>
</tr>
<tr>
<td>state</td>
<td>Current state of the case. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: New</td>
</tr>
<tr>
<td></td>
<td>• 3: Closed</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>• 6: Resolved</td>
</tr>
<tr>
<td></td>
<td>• 10: Open</td>
</tr>
<tr>
<td></td>
<td>• 18: Awaiting Info</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 1</td>
</tr>
<tr>
<td>subcategory</td>
<td>Case subcategory.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Question</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 0</td>
</tr>
<tr>
<td>support_manager</td>
<td>Sys_id of the CSM manager assigned to the case.</td>
</tr>
<tr>
<td></td>
<td>Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sync_driver</td>
<td>Flag that indicates whether driver synchronization is on.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Driver is synchronized</td>
</tr>
<tr>
<td></td>
<td>• false: Driver is not synchronized</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>sys_class_name</td>
<td>Table that contains the case record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 80</td>
</tr>
<tr>
<td>sys_created_by</td>
<td>Person that initially opened the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sys_created_on</td>
<td>Date and time when the case was initially created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_domain</td>
<td>Domain associated with the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 32</td>
</tr>
<tr>
<td></td>
<td>Default: global</td>
</tr>
<tr>
<td>sys_domain_path</td>
<td>Domain path.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td></td>
<td>Default: /</td>
</tr>
<tr>
<td>sys_id</td>
<td>Unique identifier for the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 32</td>
</tr>
<tr>
<td>sys_mod_count</td>
<td>Number of updates to the case since it was initially created.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td>sys_tags</td>
<td>System tags.</td>
</tr>
<tr>
<td>sys_updated_by</td>
<td>Person that last updated the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sys_updated_on</td>
<td>Date and time when the case was last updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>time_worked</td>
<td>Total amount of time worked on the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>upon_approval</td>
<td>Action to take if the case is approved.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• do_nothing</td>
</tr>
<tr>
<td></td>
<td>• proceed</td>
</tr>
<tr>
<td>upon_reject</td>
<td>Action to take if the case is rejected.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• cancel</td>
</tr>
<tr>
<td></td>
<td>• goto</td>
</tr>
<tr>
<td>urgency</td>
<td>Urgency of the case.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: High</td>
</tr>
<tr>
<td></td>
<td>• 2: Medium</td>
</tr>
<tr>
<td></td>
<td>• 3: Low</td>
</tr>
<tr>
<td>user_input</td>
<td>Additional user input.</td>
</tr>
<tr>
<td>variable_pool</td>
<td>Grouping of variables.</td>
</tr>
<tr>
<td>variables</td>
<td>Name-value pairs of variables associated with the case.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>watch_list</td>
<td>List of sys_ids of the users who receive notifications about this case when additional comments are added or if the state of a case is changed to Resolved or Closed. Located in the [sys_user] table. Data type: Array</td>
</tr>
<tr>
<td>wf_activity</td>
<td>Sys_id of the workflow activity record associated with the case. Located in the Workflow Activity [wf_activity] table. Data type: String</td>
</tr>
<tr>
<td>work_end</td>
<td>Date and time work ended on the case. Data type: String</td>
</tr>
<tr>
<td>work_notes</td>
<td>Information about how to resolve the case, or steps taken to resolve it. Data type: String</td>
</tr>
<tr>
<td>work_notes_list</td>
<td>List of sys_ids of the internal users who receive notifications about this case when work notes are added. Located in the [sys_user] table. Data type: Array</td>
</tr>
<tr>
<td>work_start</td>
<td>Date and time that work started on the case. Data type: String</td>
</tr>
</tbody>
</table>

**Notes on request body parameters**

All request body parameters are optional. Not all parameters apply to all case types (B2B vs B2C and order vs product.) Ensure that the parameters that you
set do not conflict, such as a mismatch between the account and contact or consumer. In addition to the list of elements defined below (which define the elements found in a base system), the endpoint also accepts custom case fields and any additional case fields configured in your instance. For additional information on these elements, refer to your specific table definitions [System Definition > Tables].

Example: cURL request
The following example illustrates how to update a consumer-based case.

```
curl  
"https://instance.servicenow.com/api/sn_customerservice/case/0397c743db436f0057c3fd441d96197f"  
--request PUT  
--header "Accept:application/json"  
--header "Content-Type:application/json"  
--data "{"  
  consumer: "cd97ef1dddb8bc7006b7a9646db9619ac",  
  contact_type: "phone",  
  priority: "4",  
  short_description: "Consumer Test Case"}"  
--user "username":"password"
```

```
{
  "result": {
    "sys_id": "0397c743db436f0057c3fd441d96197f",
    "number": "CS0001005"
  }
}
```

Example: Python request
```
# Install requests package for python
import requests

# Set the request parameters
url =  
'https://instance.servicenow.com/api/sn_customerservice/case/c7940ed1db9ba30057c3fd441d961967'

# Set the user credentials
user = 'username'
pwd = 'password'
```
# Set the proper headers
headers = {"Content-Type":"application/json","Accept":"application/json"}

# Make the HTTP request
response = requests.put(url, auth=(user, pwd), headers=headers, data="{"contact":"62fe1c97db76c3006b7a9646db961999"}"

# Check for HTTP codes other than 201
if response.status_code != 201:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "sys_id": "c7940ed1db9ba30057c3fd441d961967",
        "number": "CS0001074"
    }
}

**Change Management API**

The Change Management API provides REST APIs that enable third-party application integration with the ServiceNow Change Management process.

By integrating your application with the ServiceNow Change Management process, all change requests, regardless of where they are initiated, have a single source of truth, providing a single audit source.

Use this REST API to integrate your change management process with external applications and when developing ServiceNow client-side applications.

This REST API enables integrators to:

- Initiate a standard change request from a published standard change request template.
- Create a change request of type emergency or normal.
- Update any field that exists in the change request table for any change request.
- Update any field that exists in the change task table and work tasks from creation through closure/cancellation.
• Retrieve a specific change request, standard template, change request task, or change model.
• Retrieve multiple change requests, standard templates, change request tasks using pagination.
• Perform risk evaluation.
• Refresh impacted services.
• Generate and process any related approval activity associated with a change request.
• Identify potential scheduling conflicts and identify periods where conflicts do not exist.
• Delete change requests, change request tasks, and conflict checking processes.
• Create a change request record based on a change model record.

The Change REST API supports ITIL types and Change models using Flow Designer and Workflow. Change models deliver fit-for-purpose change. Types and models define transition criteria that must be met before the change request can progress to the next state. You can define this criteria using states, workflows, tools, and business rules.

• Traditional ITIL types: Standard, Emergency, and Normal.
• Change model states: New, Scheduled, Implement, Review, and Closed.

You can configure additional change models within your ServiceNow instance and then create change requests based on those change models using endpoints in this API. Use the Change Management - GET /sn_chg_rest/change/model/{sys_id} and Change Management - GET /sn_chg_rest/change/model endpoints to obtain the available change models in an instance. Then use the Change Management - POST /sn_chg_rest/change to create a change request based on a specified change model.

For information on configuring change models, see Configure Change Management.

The following roles are required to access the Change Management endpoints:
• DELETE: change_manager or admin
• GET: change_manager, itil, sn_change_read, or admin
• PATCH/POST: change_manager, itil, or admin
In addition, administrators can set change properties to configure Change Management behavior. For a list of these properties and a description of their available functionality, see Change Management properties.

Change Management - DELETE /sn_chg_rest/change/{change_sys_id}/task/{task_sys_id}

Deletes the change request task identified by the specified sys_ids.

**URL format**

**Versioned URL:** /api/sn_chg_rest/{api_version}/change/{change_sys_id}/task/{task_sys_id}

**Default URL:** /api/sn_chg_rest/change/{change_sys_id}/task/{task_sys_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>change_sys_id</td>
<td>Sys_id of the change request to which the task is associated. Located in the Change Request [change_request] table. Verifies that the task is actually associated to the specified change request. Data type: String</td>
</tr>
<tr>
<td>task_sys_id</td>
<td>Sys_id of the change request task to delete. Located in the Change Tasks [change_task] table. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>All fields (key) with their associated values for the identified change request task prior to the delete. Data type: Object</td>
</tr>
<tr>
<td>parent</td>
<td>Information for the change request to which this task was associated. Data type: Object</td>
</tr>
<tr>
<td>parent.display_value</td>
<td>Sys_id of the parent task to display in a UI. Data type: String</td>
</tr>
<tr>
<td>parent.value</td>
<td>Sys_id of the parent task. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id information for the deleted change request. Data type: Object</td>
</tr>
<tr>
<td>sys_id.display_value</td>
<td>Sys_id of the change request to display in a UI. Data type: String</td>
</tr>
<tr>
<td>sys_id.value</td>
<td>Sys_id of the change request. Data type: String</td>
</tr>
</tbody>
</table>

### Example: cURL request

```bash
curl "https://instance.servicenow.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6ee1ia9fe/task/12629ec4b750230096c3e4f6ee1ia9d5" \
--request DELETE 
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6ee11a9fe/task/12629ec4b750230096c3e4f6ee11a9d5'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept':'application/json'}

# Make the HTTP request
response = requests.delete(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
```
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
    response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <sys_id>
      <value>12629ec4b750230096c3e4f6ee11a9d5</value>
      <display_value>12629ec4b750230096c3e4f6ee11a9d5</display_value>
    </sys_id>
    <parent>
      <value>0f4ac6c4b750230096c3e4f6ee11a9fe</value>
      <display_value>CHG0033046</display_value>
    </parent>

    // all valid fields in record, single parameter example below
    <short_description>
      <value>Retire both nodes</value>
      <display_value>Retire both nodes</display_value>
    </short_description>
  </result>
</response>

Change Management - DELETE /sn_chg_rest/change/{sys_id}
Deletes the change request associated with the specified sys_id.

URL format
Versioned URL: /api/sn_chg_rest/{api_version}/change/{sys_id}
Default URL: /api/sn_chg_rest/change/{sys_id}
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request record to delete. Located in the Change Request [change_request] table.</td>
</tr>
</tbody>
</table>

Data type: String

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Change request record that was deleted. Each element in this object corresponds to a field in the record of the Change Request [change_request] table.</td>
</tr>
</tbody>
</table>

Data type: Object

Example: cURL request

```sh
curl "https://instance.servicenow.com/api/sn_chg_rest/v1/change/b0dbda5347c12200e0ef563dbb9a718f" \
  --request DELETE \
  --header "Accept:application/json" \
  --user "username":"password"
```
{ "result": { "reason": { "display_value": "", "value": "" }, "parent": { "display_value": "", "value": "" }, "watch_list": { "display_value": "", "value": "" }, "proposed_change": { "display_value": "", "value": "" }, "upon_reject": { "display_value": "Cancel all future Tasks", "value": "cancel" }, "sys_updated_on": { "display_value": "2015-07-06 11:59:27", "value": "2015-07-06 18:59:27", "display_value_internal": "2015-07-06 11:59:27" }, "type": { "display_value": "Standard", "value": "standard" }, "approval_history": { "display_value": "", "value": "" }, "skills": { "display_value": "", "value": "" }, "test_plan": { "display_value": "--Confirm that there are no monitoring alerts for the router", "value": "--Confirm that there are no monitoring alerts for the router" } }
-- Place router into maintenance mode in the monitoring platform
-- Logon to router through SSH
-- Run the following command

```
#router bgp 12345
#neighbor {neighbor ip} soft-reconfig [inbound]
#clear ip bgp {neighbor ip} soft in
```

-- Confirm the sessions have been cleared
-- Place router back into operational mode in the monitoring platform


"display_value": "Due to the limited number of commands in the implementation plan it is not possible to backout the change. If required you are authorized to reboot the router if BGP fails to work",
"value": "Due to the limited number of commands in the implementation plan it is not possible to backout the change. If required you are authorized to reboot the router if BGP fails to work"
]}
"conflict_status": {
"display_value": "Not Run",
"value": "Not Run"
},
"task_effective_number": {
"display_value": "CHG0000024",
"value": "CHG0000024"
},
"sys_updated_by": {
"display_value": "admin",
"value": "admin"
},
"opened_by": {
"display_value": "System Administrator",
"value": "6816f79cc0a8016401c5a33be04be441"
},
"user_input": {
"display_value": "",
"value": ""
},
"sys_created_on": {
"display_value": "2015-07-06 11:55:46",
"value": "2015-07-06 18:55:46",
"display_value_internal": "2015-07-06 11:55:46"}
"on_hold_task": { 
  "display_value": "",
  "value": ""
},
"sys_domain": { 
  "display_value": "global",
  "value": "global"
},
"route_reason": { 
  "display_value": "",
  "value": ""
},
"closed_at": { 
  "display_value": "2015-07-06 11:56:23",
  "value": "2015-07-06 18:56:23",
  "display_value_internal": "2015-07-06 11:56:23"
},
"review_comments": { 
  "display_value": "",
  "value": ""
},
"business_service": { 
  "display_value": "",
  "value": ""
},
"time_worked": { 
  "display_value": "",
  "value": ""
},
"chg_model": { 
  "display_value": "",
  "value": ""
},
"expected_start": { 
  "display_value": "",
  "value": "",
  "display_value_internal": ""
},
"opened_at": { 
  "display_value": "2015-06-09 11:55:46",
  "value": "2015-06-09 18:55:46",
  "display_value_internal": "2015-06-09 11:55:46"
},
"work_end": { 
  "display_value": "",
  "value": ""
}
"display_value": "2015-07-06 11:56:10",
"value": "2015-07-06 18:56:10",
"display_value_internal": "2015-07-06 11:56:10"
},
"phase_state": {
"display_value": "Open",
"value": "open"
},
"cab_date": {
"display_value": "",
"value": "",
"display_value_internal": ""
},
"work_notes": {
"display_value": "",
"value": ""
},
"close_code": {
"display_value": "Successful",
"value": "successful"
},
"assignment_group": {
"display_value": "Network",
"value": "287ebd7da9fe198100f92cc8d1d2154e"
},
"description": {
"display_value": "Resend the complete BGP table to neighboring routers.\n\n--Both neighbors need to support soft reset route refresh capability.\n\n--Stores complete BGP table of you neighbor in router memory.\n\n--Not a good idea on a peering router with full feed, due to the memory requirements.\n\n",
"value": "Resend the complete BGP table to neighboring routers.\n\n--Both neighbors need to support soft reset route refresh capability.\n\n--Stores complete BGP table of you neighbor in router memory.\n\n--Not a good idea on a peering router with full feed, due to the memory requirements.\n\n"
},
"on_hold_reason": {
"display_value": "",
"value": ""
},
"calendar_duration": {
"display_value": "",
"value": ""
},
"close_notes": {
"display_value": "Completed without issues",
"value": "Completed without issues"
},
"sys_id": {
"display_value": "b0dbda5347c12200e0ef563dbb9a718f",
"value": "b0dbda5347c12200e0ef563dbb9a718f"
},
"contact_type": {
"display_value": "Phone",
"value": "phone"
},
"cab_required": {
"display_value": "false",
"value": false
},
"urgency": {
"display_value": "3 - Low",
"value": 3.0
},
"scope": {
"display_value": "Medium",
"value": 3.0
},
"company": {
"display_value": "",
"value": ""
},
"justification": {
"display_value": "",
"value": ""
},
"activity_due": {
"display_value": "UNKNOWN",
"value": "",
"display_value_internal": ""
},
"comments": {
"display_value": "",
"value": ""
},
"approval": {
"display_value": "Approved",
"value": "approved"
},
"due_date": {
    "display_value": "",
    "value": "",
    "display_value_internal": ""
},
"sys_mod_count": {
    "display_value": "10",
    "value": 10.0
},
"on_hold": {
    "display_value": "false",
    "value": false
},
"sys_tags": {
    "display_value": "",
    "value": ""
},
"conflict_last_run": {
    "display_value": "",
    "value": "",
    "display_value_internal": ""
},
"risk_value": {
    "display_value": "",
    "value": ""
},
"unauthorized": {
    "display_value": "false",
    "value": false
},
"risk": {
    "display_value": "Moderate",
    "value": 3.0
},
"location": {
    "display_value": "",
    "value": ""
},
"category": {
    "display_value": "Other",
    "value": "Other"
},
"risk_impact_analysis": {
    "display_value": ""
}
**Change Management - DELETE /sn_chg_rest/change/{sys_id}/conflict**

Cancels the running conflict checking process for the specified change request (sys_id).

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/{sys_id}/conflict

Default URL: /api/sn_chg_rest/change/{sys_id}/conflict

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request record for which to cancel the running conflict checking process. Located in the Change Request [change_request] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes
The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad request. Cancel request failed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
**Example: cURL request**

```bash
curl
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6ee11a9fe/conflict" \
  --request DELETE \
  --user "username":"password"
```

**Example: Python request**

```python
# Install requests package for python
import requests

# Set the request parameters
url =
  'https://instance.servicenow.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6ee11a9fe/conflict'

# Set the user credentials
user = 'username'
pwd = 'password'

# Make the HTTP request
response = requests.delete(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()
```

**Change Management - DELETE /sn_chg_rest/change/emergency/{sys_id}**

Deletes the emergency change request identified by the specified sys_id.

**URL format**

Versioned URL: `/api/sn_chg_rest/{api_version}/change/emergency/{sys_id}`

Default URL: `/api/sn_chg_rest/change/emergency/{sys_id}`
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the emergency change request to delete. Located in the Change Request [change_request] table.</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
## Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](rest-api-http-response-codes).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Values for all fields in the associated change request. Data type: Object</td>
</tr>
<tr>
<td>state</td>
<td>State of the change request prior to the delete. Data type: Object</td>
</tr>
<tr>
<td>state.display_value</td>
<td>State to display in the UI. Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Internal state value. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id information for the change request. Data type: Object</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sys_id.display_value</td>
<td>Sys_id of the change request to display in a UI.</td>
</tr>
<tr>
<td>sys_id.value</td>
<td>Sys_id of the change request.</td>
</tr>
<tr>
<td>type</td>
<td>Type of change request.</td>
</tr>
<tr>
<td>type.display_value</td>
<td>Change type to display in a UI.</td>
</tr>
<tr>
<td>type.value</td>
<td>Internal type value.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl "https://instance.servicenow.com/api/sn_chg_rest/v1/change/emergency/b0dbda5347c12200e0ef563dbb9a718f" \
--request DELETE \
--header "Accept:application/json" \
--user "username":"password"
```

```
{
  result: [
    {
      sys_id: {
        value: "b0dbda5347c12200e0ef563dbb9a718f",
        display_value: "b0dbda5347c12200e0ef563dbb9a718f"
      }
    }
  ]
}
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/emergency/b0dbda5347c12200e0ef563dbb9a718f'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.delete(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
# Decode the XML response into a dictionary and use the data
print(response.content)

```xml
<response>
  <result>
    <sys_id>
      <value>b0dbda5347c12200e0ef563dbb9a718f</value>
      <display_value>b0dbda5347c12200e0ef563dbb9a718f</display_value>
    </sys_id>
    <state>
      <value>-5</value>
      <display_value>New</display_value>
    </state>
    <type>
      <value>emergency</value>
      <display_value>Emergency</display_value>
    </type>
    // all valid fields in record, single parameter example below
    <short_description>
      <value>Remove server</value>
      <display_value>Remove server</display_value>
    </short_description>
  </result>
</response>
```

### Change Management - DELETE /sn_chg_rest/change/normal/{sys_id}

Deletes the normal change request identified by the specified sys_id.

### URL format

**Versioned URL:** /api/sn_chg_rest/{api_version}/change/normal/{sys_id}

**Default URL:** /api/sn_chg_rest/change/normal/{sys_id}

### Supported request parameters

#### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
</tbody>
</table>
### Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the normal change request to delete Located in the Change Request [change_request] table. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json Of application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Values for all fields in the associated change request. Data type: Object</td>
</tr>
<tr>
<td>state</td>
<td>State of the change request prior to the delete. Data type: Object</td>
</tr>
<tr>
<td>state.display_value</td>
<td>State to display in the UI. Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Internal state value. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request. Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Type of change request.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>type: `{</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>type.display_value</td>
<td>Change type to display in the UI.</td>
</tr>
<tr>
<td></td>
<td>Value is always &quot;Normal&quot;.</td>
</tr>
<tr>
<td>type.value</td>
<td>Internal type value.</td>
</tr>
<tr>
<td></td>
<td>Value is always &quot;normal&quot;.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/normal/b0dbda5347c12200e0ef563dbb9a718f" \
  --request DELETE \
  --header "Accept:application/json" \
  --user "username":"password"

{
  result: [
    {
      sys_id: "b0dbda5347c12200e0ef563dbb9a718f",
      state: {
        value: "-5",
        display_value: "New"
      },
      type: {
        value: "normal",
        display_value: "Normal"
      },
      ... // all valid fields in record, example below
      short_description: {
        value: "Remove server",
        display_value: "Remove server"
      },
    },
  ]
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url =
    'https://instance.servicenow.com/api/sn_chg_rest/v1/change/normal/b0dbda5347c12200e0ef563dbb9a718f'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Accept": "application/xml"}

# Make the HTTP request
response = requests.delete(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
    <result>
        <sys_id>b0dbda5347c12200e0ef563dbb9a718f</sys_id>
        <state>
            <value>-5</value>
            <display_value>New</display_value>
        </state>
        <type>
            <value>normal</value>
            <display_value>Normal</display_value>
        </type>
    </result>
</response>
```
// all valid fields in record, single parameter example below
<short_description>
  <value>Remove server</value>
  <display_value>Remove server</display_value>
</short_description>
</result>
</response>

Change Management - DELETE /sn_chg_rest/change/standard/{sys_id}
Deletes the standard change request identified by the specified sys_id.

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/standard/{sys_id}
Default URL: /api/sn_chg_rest/change/standard/{sys_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the standard change request to delete. Located in the Change Request [change_request] table. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>All fields (key) with their associated values for the identified change request.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>state</td>
<td>State of the change request prior to the delete.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>state: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td>state.display_value</td>
<td>State to display in the UI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Internal state value.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Unique identifier of the change request.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
   "https://instance.servicenow.com/api/sn_chg_rest/v1/change/standard/1c87925347c12200e0ef563dbb9a7177"
   --request DELETE 
   --header "Accept:application/json" 
   --user "username":"password"
{
   result: [
   {
   sys_id: "1c87925347c12200e0ef563dbb9a7177",
   state: {
   value: "-5",
   display_value: "New"
   },
   ..., // all valid fields in record, example below
   short_description: {
   value: "Add network switch to cabinet",
   display_value: "Add network switch to cabinet"
   },
   },
   ]
}
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url =
'https://instance.servicenow.com/api/sn_chg_rest/v1/change/standard/1c87925347c12200e0ef563dbb9a7177'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.delete(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <sys_id>1c87925347c12200e0ef563dbb9a7177</sys_id>
    <state>
      <value>-5</value>
      <display_value>New</display_value>
    </state>
    // all valid fields in record, single parameter example below
    <short_description>
      <value>Add network switch to cabinet</value>
      <display_value>Add network switch to cabinet</display_value>
    </short_description>
  </result>
</response>
```
Change Management - GET /sn_chg_rest/change/ci/{cmdb_ci_sys_id}/schedule

Enables retrieving available time slots by configuration item ID and duration, with an option to include planned start time.

Role required: sn_change_writer

⚠️ Note: Running this endpoint does not list the available start and end times. Use the link provided in the response body `worker.link` property to get the schedule data.

See also:
- GET /sn_chg_rest/change/{change_sys_id}/schedule
- PATCH /sn_chg_rest/change/{change_sys_id}/schedule/first_available

**URL format**

**Versioned URL:** /api/sn_chg_rest/{api_version}/change/ci/{cmdb_ci_sys_id}/schedule

**Default URL:** /api/sn_chg_rest/change/ci/{cmdb_ci_sys_id}/schedule

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>cmdb_ci_sys_id</td>
<td>Sys_id of a record in the Configuration Items [cmdb_ci] table. This endpoint does not require a change request.</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>duration_in_seconds</td>
<td>Duration of change in seconds, that is, how much time is required to complete the change request task.</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>planned_start_time</td>
<td>Optional. Date and time that the change request is planned to start implementation in UTC. Retrieve the available time slot start at or later than this time. If not provided, the system uses the current time as the start time. Time format: yyyy-mm-dd hh:mm:ss</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>202</td>
<td>System accepted the request.</td>
</tr>
</tbody>
</table>
| 400         | Bad Request. A bad request type or malformed request was detected. Possible issues:  
  - Cannot find the cmdb_ci with the provided sys_id. The record either does not exist or the user does not have read access to it.  
  - The duration_in_seconds query parameter value was not provided.  
  - Invalid duration_in_seconds or planned_start_time query parameter value provided. |

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| error        | Information on any errors encountered while processing the endpoint request.  
  Data type: Object  
  
  ```json
  "error": {
    "detail": "String",
    "message": "String",
    "status": "String"
  }
  ``` |
| error.detail | Additional information about the error.  
  Data type: String |
| error.message| Message that identifies the error.  
  Data type: String |
| messages     | Message information.  
  Data type: Object  
  
  ```json
  "messages": {
    "errorMessages": [Array],
    "infoMessages": [Array],
    "warningMessages": [Array]
  }
  ``` |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>messages.errorMessages</td>
<td>Error messages encountered while processing the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>messages.infoMessages</td>
<td>Information messages encountered while processing the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>messages.warningMessages</td>
<td>Warning messages encountered while processing the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>request</td>
<td>Original endpoint request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>Information on the current state of the worker.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>state: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: Number</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>state.display_value</td>
<td>Display value of the state of the worker. These values directly correlate to</td>
</tr>
<tr>
<td></td>
<td>the state.value parameter.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Complete</td>
</tr>
<tr>
<td></td>
<td>• Error</td>
</tr>
<tr>
<td></td>
<td>• In-Progress</td>
</tr>
<tr>
<td></td>
<td>• Waiting</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Numeric value of the state of the worker.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1</td>
</tr>
<tr>
<td></td>
<td>• 2</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>type</td>
<td>Indicates the type of request. Valid value: schedule</td>
</tr>
<tr>
<td>worker</td>
<td>Information about the associated worker.</td>
</tr>
<tr>
<td>worker.link</td>
<td>Link for retrieving time slot data. Use the sys_id in GET /sn_chg_rest/change/worker/{sys_id} to view results.</td>
</tr>
<tr>
<td>worker.sysId</td>
<td>Sys_id of the worker associated with the change request.</td>
</tr>
<tr>
<td>status</td>
<td>Only appears if an error is encountered. Status of the endpoint processing. Possible values: failure</td>
</tr>
</tbody>
</table>

### Get available time slots

Use the value provided in the `worker.link` to get schedule window details. The value is in the following format:

```
https://instance.service-now.com/api/sn_chg_rest/change/worker/<worker_sys_id>
```

Use the `worker_sys_id` in GET /sn_chg_rest/change/worker/{sys_id} to view results.

The response body contains the status and provides results when processing is complete.
Worker response body parameter results vary depending on time slot availability.

- If the provided time slot is available for the change request within the schedule time slot, the worker API lists the available time slots in the `payload.spans` property. The `payload.spans` property is not listed in the results otherwise.

- If there are no time slots available for change request duration provided within the defined scheduling time slot, the `messages.infoMessages` states the following:

  No slots found for <number> days from <date and time in format yyyy-mm-dd hh:mm:ss>

  **Note:** The change request scheduling time slot default value is 90 days. To change this value, modify the `change.conflict.next_available.schedule_window` property. For more information, see Configure conflict analysis properties.

The following GET `/sn_chg_rest/change/worker/{sys_id}` example shows output provided using the ID provided in the worker.link detail. The results list open time spans available for the task duration.

```json
{
  "result": {
    "worker": {
      "sysId": "d7d1f2b4a444b010f87712198fe9caae",
      "link": "https://instance.service-now.com/api/sn_chg_rest/change/worker/d7d1f2b4a444b010f87712198fe9caae"
    },
    "request": {
      "cmdb_ci_sys_id": "82967cdd0ad3370236092104ce988d76",
      "planned_start_time": "",
      "duration_in_seconds": 10800,
      "timezone": "America/Los_Angeles"
    },
    "state": {
      "value": 3,
      "display_value": "Complete"
    },
    "type": "schedule",
    "messages": {
      "errorMessages": [],
      "warningMessages": [],
      "infoMessages": []
    },
    "payload": {
```
"spans": [
    {
        "start": {
            "value": "2021-05-15 08:00:00",
            "display_value": "2021-05-15 01:00:00"
        },
        "end": {
            "value": "2021-05-15 11:00:00",
            "display_value": "2021-05-15 04:00:00"
        }
    },
    {
        "start": {
            "value": "2021-05-22 08:00:00",
            "display_value": "2021-05-22 01:00:00"
        },
        "end": {
            "value": "2021-05-22 11:00:00",
            "display_value": "2021-05-22 04:00:00"
        }
    },
    ...
]

Example: cURL request

curl
"https://instance.service-now.com/api/sn_chg_rest/change/ci/<cmdb_ci_sys_id>/schedule?duration_in_seconds=10800" \
--request GET \
--header "Accept:application/json" \
--user "username":"password"

Results include worker.link details you can use to run the provided sys_id in the GET /sn_chg_rest/change/worker/ endpoint.

{
    "result": {
        "worker": {
            "sysId": "1049419c1b4c3010f58a6572604bcb7a",
        }
    }
}
Change Management - GET /sn_chg_rest/change

Retrieves one or more change requests based on the specified criteria.

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change

Default URL: /api/sn_chg_rest/change

**Supported request parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>
### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Name-value pairs to use to filter the result set. The name is the field on which the specified value is filtered. This parameter is mutually exclusive with <strong>sysparm_query</strong>. For example, instead of using &amp;sysparm_query=active=true, you can simplify the calling statement by using &amp;active=true. You can also use the display value when the field is a choice or reference type field, such as &amp;state=closed instead of &amp;state=7. To specify multiple key-value pairs, separate each with an ampersand, such as &amp;active=true&amp;assigned_to=john.smith. Data type: String</td>
</tr>
<tr>
<td>order</td>
<td>Field by which to sort the returned change requests.</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, <strong>sysparm_offset</strong> is set to &quot;0&quot;. Simply page through all available records, use <strong>sysparm_offset=sysparm_offset+sysparm_limit</strong>, until you reach the end of all records. Do not pass a negative number in the <strong>sysparm_offset</strong> parameter. Data type: Number Default: 0</td>
</tr>
</tbody>
</table>
| sysparm_query         | Encoded query used to filter the result set. **Syntax:** sysparm_query=<col_name><operator><value>.  
  - **<col_name>:** Name of the table column to filter against.  
  - **<operator>:** Supports the following values:  
    - =: Exactly matches <value>.  
    - !=: Does not match <value>.  
    - ^: Logically AND multiple query statements.  
    - ^OR: Logically OR multiple query statements.  
    - LIKE: <col_name> contains the specified string. Only works for <col_name> fields whose data type is string. |
**Query parameters (continued)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>STARTSWITH</strong>:</td>
<td>&lt;col_name&gt; starts with the specified string. Only works for &lt;col_name&gt; fields whose data type is string.</td>
</tr>
<tr>
<td><strong>ENDSWITH</strong>:</td>
<td>&lt;col_name&gt; ends with the specified string. Only works for &lt;col_name&gt; fields whose data type is string.</td>
</tr>
<tr>
<td><strong>&lt;value&gt;</strong>:</td>
<td>Value to match against.</td>
</tr>
</tbody>
</table>

All parameters are case-sensitive. Queries can contain more than one entry, such as
```
sysparm_query=<col_name><operator><value>[<operator><col_name><operator><value>]
```
For example:
```
(sysparm_query=caller_id=javascript:gs.getUserID()^active=true)
```
Encoded queries also supports order by functionality. To sort responses based on certain fields, use the ORDERBY and ORDERBYDESC clauses in `sysparm_query`.

**Syntax:**
- ORDERBY<col_name>
- ORDERBYDESC<col_name>

**For example:** `sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory`

This query filters all active records and orders the results in ascending order by number, and then in descending order by category.

If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property `glide.invalid_query.returns_no_rows`. Set this property to true to return no rows for invalid queries.

**Note:** The `glide.invalid_query.returns_no_rows` property controls the behavior of all queries across the instance, such as in lists, scripts (`GlideRecord.query()`), and web service APIs.

**Data type:** String

<table>
<thead>
<tr>
<th>textSearch</th>
<th>String to use to search all normal change request record fields. This search uses ServiceNow full text search platform functionality. For more information on ServiceNow search capabilities, see <a href="#">Search administration</a>.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default:</strong></td>
<td>IR_AND_OR_QUERY</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>
Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>List containing one or more change request record objects. Each object describes a change request. Each element in the change request object corresponds to a field in its associated record in the Change Request [change_request] table. All elements contain value and display_value name-value pairs. Date fields also contain display_value_internal name-value pairs. Data type: Array</td>
</tr>
</tbody>
</table>
| action_status | Current action status of the associated change request. Possible values:  
  • 1: Blocked internally  
  • 2: Blocked by customer  
  • 3: Blocked internally and by customer  
  • 4: Needs attention  
Data type: Number                                                                                                         |
| active        | Flag that indicates whether the change request is active. Possible values:  
  • true: Change request is active  
  • false: Change request is not active  
Data type: Boolean  
Default: true                                                                                                              |
| activity_due  | Date and time for which the associated case is expected to be completed.  
Data type: String                                                                                                          |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>additional_assignee_list</td>
<td>List of sys_ids of additional persons assigned to work on the change request. Data type: Array</td>
</tr>
<tr>
<td>approval</td>
<td>Type of approval process required.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: not requested</td>
</tr>
<tr>
<td>approval_history</td>
<td>Most recent approval history journal entry.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>approval_set</td>
<td>Date and time that the associated action was approved.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>assigned_to</td>
<td>Sys_id of the user assigned to the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>assignment_group</td>
<td>Sys_id of the group assigned to the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>backout_plan</td>
<td>Description of the plan to execute if the change must be reversed.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>business_duration</td>
<td>Length in scheduled work hours, work days, and work weeks that it took to complete the change.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>business_service</td>
<td>Sys_id of the business service associated with the change request. Located in the Service [cmdb_ci_service] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>cab_date</td>
<td>Date on which the Change Advisory Board (CAB) meets.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>cab_delegate</td>
<td>Sys_id of the user that can substitute for the CAB manager during a CAB meeting. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>cab_recommendation</td>
<td>Description of the CAB recommendations for the change request. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>cab_required</td>
<td>Flag that indicates whether the CAB is required. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Change Advisory Board is required.</td>
</tr>
<tr>
<td></td>
<td>• false: Change Advisory Board is not required.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>calendar_duration</td>
<td>Not currently used by Change Management. Data type: String</td>
</tr>
<tr>
<td>category</td>
<td>Category of the change, for example hardware, network, or software. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: Other</td>
</tr>
<tr>
<td>change_plan</td>
<td>Activities and roles for managing and controlling the change request. Data type: String</td>
</tr>
<tr>
<td>chg_model</td>
<td>Sys_id of the change model that the associated change request was based on. Located in the Change Model [chg_model] table. The Change Model defines the state flow, transitions, and process activities that must be completed for the change request.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>closed_at</td>
<td>Date and time that the associated change request was closed.</td>
</tr>
<tr>
<td>closed_by</td>
<td>Sys_id of the person that closed the change request. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td>close_code</td>
<td>Code assigned to the change request when it was closed. For example, Successful, Successful with issues, and Unsuccessful.</td>
</tr>
<tr>
<td>close_notes</td>
<td>Notes that the person entered when closing the change request.</td>
</tr>
<tr>
<td>cmdb_ci</td>
<td>Sys_id of the configuration item associated with the change request. Located in the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td>comments</td>
<td>List of customer-facing work notes entered in the associated change request.</td>
</tr>
<tr>
<td>comments_and_work_notes</td>
<td>List of both internal and customer-facing work notes entered for the associated change request.</td>
</tr>
<tr>
<td>company</td>
<td>Sys_id of the company associated with the change request. Located in the Company [core_company] table.</td>
</tr>
<tr>
<td>conflict_last_run</td>
<td>Date and time that the conflict detection script was last run on the change request.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>conflict_status</td>
<td>Current conflict status as detected by the conflict detection script, such as Conflict and Not Run. Data type: String, Maximum length: 40, Default: Not Run</td>
</tr>
<tr>
<td>contact_type</td>
<td>Method in which the change request was initially requested. Possible values: chat, email, phone, social, web. Data type: String</td>
</tr>
<tr>
<td>contract</td>
<td>Sys_id of the contract associated with the change request. Located in the Contract [ast_contract] table. Data type: String</td>
</tr>
<tr>
<td>correlation_display</td>
<td>User-friendly name for the correlation_id. Data type: String, Maximum length: 100</td>
</tr>
<tr>
<td>correlation_id</td>
<td>Globally unique ID (GUID) of a matching change request record in a third-party system. Data type: String, Maximum length: 100</td>
</tr>
<tr>
<td>delivery_plan</td>
<td>No longer in use. Sys_id of the delivery plan associated with the change request. Located in the Execution Plan [sc_cat_item_delivery_plan] table.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>delivery_task</td>
<td>No longer in use. Sys_id of the delivery task associated with the change request. Located in the Execution Plan Task [sc_cat_item_delivery_task] table. Data type: String</td>
</tr>
<tr>
<td>description</td>
<td>Detailed description of the change request. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>due_date</td>
<td>Task due date. Not used by change request process. Data type: String</td>
</tr>
<tr>
<td>end_date</td>
<td>Date and time when the change request is to be completed. Data type: String</td>
</tr>
</tbody>
</table>
| escalation    | Current escalation level. Possible values:  
- 0: Normal  
- 1: Moderate  
- 2: High  
- 3: Overdue  
Data type: Number (Integer) Default: 0                                                                                                                                                                                                                                                                                                                                 |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group_list</td>
<td>List of sys_ids and names of the groups associated with the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>impact</td>
<td>Impact on the change request will have on the customer.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: High</td>
</tr>
<tr>
<td></td>
<td>• 2: Medium</td>
</tr>
<tr>
<td></td>
<td>• 3: Low</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 3</td>
</tr>
<tr>
<td>implementation_plan</td>
<td>Sequential steps to execute to implement this change. It also contains any</td>
</tr>
<tr>
<td></td>
<td>dependencies between steps and assignee details for each step.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>justification</td>
<td>Benefits of implementing this change and the impact if this change is not</td>
</tr>
<tr>
<td></td>
<td>implemented.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>knowledge</td>
<td>Flag that indicates whether there are any knowledge base (KB) articles</td>
</tr>
<tr>
<td></td>
<td>associated with the change request.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Associated KB articles</td>
</tr>
<tr>
<td></td>
<td>• false: No associated KB articles</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>location</td>
<td>Sys_id and name of the location of the equipment referenced in the change request. Located in the Location Located in the Location [cmn_location] table. Data type: String</td>
</tr>
<tr>
<td>made_sla</td>
<td>No longer used. Flag that indicates whether the change request was implemented in alignment with the associated service level agreement. Data type: Boolean</td>
</tr>
</tbody>
</table>
| needs_attention | Flag that indicates whether the change request needs attention. Possible values:  
  • true: Change request needs additional attention.  
  • false: Change request does not need additional attention.  
  Data type: Boolean  
  Default: false |
| number        | Change number assigned to the change request by the system, such as CHG0040007. Data type: String                                                                                                           |
| on_hold       | Flag that indicates whether the change request is currently on hold. Possible values:  
  • true: On hold  
  • false: Not on hold  
  Data type: Boolean  
  Default: false |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>on_hold_reason</td>
<td>If the <code>on_hold</code> parameter is &quot;true&quot;, description of the reason why the change request is being held up. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>on_hold_task</td>
<td>If the <code>on_hold</code> parameter is &quot;true&quot;, list of the sys_ids of the tasks that must be completed before the hold is released. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>opened_at</td>
<td>Date and time that the change release was created. Data type: String</td>
</tr>
<tr>
<td>opened_by</td>
<td>Sys_id and name of the user that created the change release. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>order</td>
<td>Not used by Change Management. Optional numeric field by which to order records, such as when retrieving them from a database. Data type: Number (Integer)</td>
</tr>
</tbody>
</table>
| outside_maintenance_schedule       | Flag that indicates whether maintenance by an outside company has been schedule for the change request. Possible values:  
  • true: Outside maintenance scheduled  
  • false: No outside maintenance scheduled  
 Data type: Boolean Default: false |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent</td>
<td>Sys.id and name of the parent task to this change request, if any. Located in the Task [task] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>phase</td>
<td>Current phase of the change request. This defines what the change is doing in greater detail. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• accept</td>
</tr>
<tr>
<td></td>
<td>• build</td>
</tr>
<tr>
<td></td>
<td>• plan</td>
</tr>
<tr>
<td></td>
<td>• requested</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>phase_state</td>
<td>Change_phase records that should be created for a change. They are dependent on the category, such that each type of change may have different change_phase records. The change_phase records provide an opportunity to control the approval process as each change_phase can have a schedule and a set of approvers. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• complete</td>
</tr>
<tr>
<td></td>
<td>• on hold</td>
</tr>
<tr>
<td></td>
<td>• open</td>
</tr>
<tr>
<td></td>
<td>• rejected</td>
</tr>
<tr>
<td></td>
<td>• requested</td>
</tr>
<tr>
<td></td>
<td>• work in progress</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>priority</td>
<td>Priority of the change request. Possible values:</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
|                      | • 1: Critical  
|                      | • 2: High     
|                      | • 3: Moderate  
|                      | • 4: Low      
|                      | Data type: Number (Integer)  
|                      | Default: 4    |
| production_system    | Flag that indicates whether the change request is for a ServiceNow instance that is in a production environment.  
|                      | Possible values:  
|                      | • true: Production environment  
|                      | • false: Non-production environment  
|                      | Data type: Boolean |
| reason               | Description of why the change request was initiated.  
|                      | Possible values:  
|                      | • Business requirements  
|                      | • Hardware upgrade  
|                      | • Legislation  
|                      | • Location change  
|                      | • Network requirements  
|                      | • New or removed CI  
|                      | • Other  
|                      | • Problem resolved  
|                      | • Product or service changed  
|                      | • Software upgrade  
|                      | • User requested  
|                      | Data type: String  
<p>|                      | Maximum length: 40 |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reassignment_count</td>
<td>Number of times that the change request has been reassigned to a new owner. Data type: Number (Integer) Default: 0</td>
</tr>
<tr>
<td>rejection_goto</td>
<td>Sys_id of the task to perform if the change request is rejected. Located in the Task [table]. Data type: String</td>
</tr>
<tr>
<td>requested_by</td>
<td>Sys_id of the user that requested the change. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>requested_by_date</td>
<td>Date and time when the change is requested to be implemented by. Data type: String</td>
</tr>
<tr>
<td>review_comments</td>
<td>Comments entered when the change request was reviewed. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>review_date</td>
<td>Date that the change request was reviewed. Data type: String</td>
</tr>
<tr>
<td>review_status</td>
<td>Current status of the requested change request review. Data type: String</td>
</tr>
</tbody>
</table>
| risk             | Level of risk associated with the change request. Valid values:  
|                  | • 1: High  
|                  | • 2: Moderate  
|                  | • 3: Low  
<p>|                  | Data type: Number                                                                 |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default: 3</td>
<td></td>
</tr>
<tr>
<td>risk_impact_analysis</td>
<td>Description of the risk and analysis of implementing the change request. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>route_reason</td>
<td>Not currently used by Change Management. Reason that the change request was transferred.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Transfer with Resolution</td>
</tr>
<tr>
<td></td>
<td>• 9: Transfer without Resolutions</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>scope</td>
<td>Size of the change request. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Massive</td>
</tr>
<tr>
<td></td>
<td>• 2: Large</td>
</tr>
<tr>
<td></td>
<td>• 3: Medium</td>
</tr>
<tr>
<td></td>
<td>• 4: Small</td>
</tr>
<tr>
<td></td>
<td>• 5: Tiny</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Default: 3</td>
</tr>
<tr>
<td>service_offering</td>
<td>Sys_id of the service offering associated with the change request. Service offerings</td>
</tr>
<tr>
<td></td>
<td>uniquely define the level of service in terms of availability, scope, pricing, and packaging</td>
</tr>
<tr>
<td></td>
<td>options. Located in the Offering [service_offering] table. Data type: String</td>
</tr>
<tr>
<td>short_description</td>
<td>Description of the change request. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>skills</td>
<td>List of the sys_ids of all of the skills required to implement the change request. Located in the Skill [cmn_skill] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>sla_due</td>
<td>No longer in use. Date and time that the change request must be completed based on the associated service level agreement.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>start_date</td>
<td>Date and time that the change request is planned to start implementation.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>Current state of the change request. Possible values are defined in the change model.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 1</td>
</tr>
<tr>
<td>std_change_producer_version</td>
<td>Sys_id of the record producer and change proposal associated with the change request. It also includes the number and percentage of successful and unsuccessful change requests created from the proposal. Located in the Standard Change Template Version [std_change_producer_version] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sys_class_name</td>
<td>Name of the table in which the change request is located.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_created_by</td>
<td>Name of the user that initially created the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sys_created_on</td>
<td>Date and time that the associated change request record was originally created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_domain</td>
<td>If using domains in the instance, the name of the domain to which the change module record is associated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_domain_path</td>
<td>If using domains in the instance, the domain path in which the associated change module record resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Unique identifier of the associated change request record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_mod_count</td>
<td>Number of updates to the case since it was initially created.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td>sys_updated_by</td>
<td>Person that last updated the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sys_updated_on</td>
<td>Date and time when the case was last updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>task_effective_number</td>
<td>Universal request number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>task_for</td>
<td>Not used by Change Management. Sys_id of the user that the task was created for. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>test_plan</td>
<td>Description of the associated test plan for the change. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>time_worked</td>
<td>Total amount of time worked on the change request. Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Change request type. Possible values: • emergency • normal • standard Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>unauthorized</td>
<td>Flag that indicates whether the change request is unauthorized Possible values: • true: Unauthorized • false: Authorized Data type: Boolean</td>
</tr>
<tr>
<td>universal_request</td>
<td>Sys_id of the Parent Universal request to which this change request is a part of. Located in the Task [task] table. Data type: String</td>
</tr>
<tr>
<td>upon_approval</td>
<td>Action to take if the change request is approved.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• do_nothing</td>
</tr>
<tr>
<td></td>
<td>• proceed</td>
</tr>
<tr>
<td>upon_reject</td>
<td>Action to take if the change request is rejected.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• cancel</td>
</tr>
<tr>
<td></td>
<td>• goto</td>
</tr>
<tr>
<td>urgency</td>
<td>Urgency of the change request.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: High</td>
</tr>
<tr>
<td></td>
<td>• 2: Medium</td>
</tr>
<tr>
<td></td>
<td>• 3: Low</td>
</tr>
<tr>
<td>user_input</td>
<td>Additional user input.</td>
</tr>
<tr>
<td>variables</td>
<td>Name-value pairs of variables associated with the change request.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>watch_list</td>
<td>List of sys_ids of the users who receive notifications about this change request when additional comments are added or if the state of a change request is changed to Resolved or Closed. Located in the User [sys_user] table. Data type: Array</td>
</tr>
<tr>
<td>wf_activity</td>
<td>Sys_id of the workflow activity record associated with the change request. Located in the Workflow Activity [wf_activity] table. Data type: String</td>
</tr>
<tr>
<td>work_end</td>
<td>Date and time work ended on the change request. Data type: String</td>
</tr>
<tr>
<td>work_notes</td>
<td>Information about how to resolve the change request, or steps taken to resolve it. Data type: String</td>
</tr>
<tr>
<td>work_notes_list</td>
<td>List of sys_ids of the internal users who receive notifications about this change request when work notes are added. Located in the User [sys_user] table. Data type: Array</td>
</tr>
<tr>
<td>work_start</td>
<td>Date and time that work started on the change request. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl  
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change?sysparm_query=active=true"ORDER 
BYnumber" \ 
  --request GET \ 
  --header "Accept:application/json" \ 
  --user "username":"password"
```
For brevity, the results only contain a single change request record.

```json
{
  "result": [
    {
      "reason": {
        "display_value": "",
        "value": ""
      },
      "parent": {
        "display_value": "",
        "value": ""
      },
      "watch_list": {
        "display_value": "",
        "value": ""
      },
      "proposed_change": {
        "display_value": "",
        "value": ""
      },
      "upon_reject": {
        "display_value": "Cancel all future Tasks",
        "value": "cancel"
      },
      "sys_updated_on": {
        "display_value": "2015-07-06 11:59:27",
        "value": "2015-07-06 18:59:27",
        "display_value_internal": "2015-07-06 11:59:27"
      },
      "type": {
        "display_value": "Standard",
        "value": "standard"
      },
      "approval_history": {
        "display_value": "",
        "value": ""
      },
      "skills": {
        "display_value": "",
        "value": ""
      },
      "test_plan": {
        "display_value": "$confirm that there are no monitoring alerts for the router",
        "value": ""
      }
    }
  ]
}```
"value": "---Confirm that there are no monitoring alerts for the router"
},
"number": {
  "display_value": "CHG0000024",
  "value": "CHG0000024"
},
"is_bulk": {
  "display_value": "false",
  "value": false
},
"cab_delegate": {
  "display_value": "",
  "value": ""
},
"requested_by_date": {
  "display_value": "",
  "value": "",
  "display_value_internal": ""
},
"ci_class": {
  "display_value": "cmdb_ci",
  "value": "cmdb_ci"
},
"state": {
  "display_value": "Closed",
  "value": 3.0
},
"sys_created_by": {
  "display_value": "admin",
  "value": "admin"
},
"knowledge": {
  "display_value": "false",
  "value": false
},
"order": {
  "display_value": "",
  "value": ""
},
"phase": {
  "display_value": "Requested",
  "value": "requested"
},
"cmdb_ci": {
"display_value": "", "value": ""
],
"delivery_plan": {
  "display_value": "", "value": ""
},
"impact": {
  "display_value": "3 - Low", "value": 3.0
},
"contract": {
  "display_value": "", "value": ""
},
"active": {
  "display_value": "false", "value": false
},
"work_notes_list": {
  "display_value": "", "value": ""
},
"priority": {
  "display_value": "4 - Low", "value": 4.0
},
"sys_domain_path": {
  "display_value": "/", "value": "/
"},
"cab_recommendation": {
  "display_value": "", "value": ""
},
"production_system": {
  "display_value": "false", "value": false
},
"rejection_goto": {
  "display_value": "", "value": ""
},
"review_date": {
implementation_plan": {
  "display_value": "-- Place router into maintenance mode in the monitoring platform\n-- Logon to router through SSH\n-- Run the following command\nrouter(config-router)#router bgp 12345\nneighbor {neighbor ip} soft-reconfig [inbound]\nrouter#clear ip bgp {neighbor ip} soft in\n-- Confirm the sessions have been cleared\n-- Place router back into operational mode in the monitoring platform",
  "value": "-- Place router into maintenance mode in the monitoring platform\n-- Logon to router through SSH\n-- Run the following command\nrouter(config-router)#router bgp 12345\nneighbor {neighbor ip} soft-reconfig [inbound]\nrouter#clear ip bgp {neighbor ip} soft in\n-- Confirm the sessions have been cleared\n-- Place router back into operational mode in the monitoring platform",
  "universal_request": {
"display_value": "",
"value": ""
},
"end_date": {
"display_value": "",
"value": "",
"display_value_internal": ""
},
"short_description": {
"display_value": "Clear BGP sessions on a Cisco router",
"value": "Clear BGP sessions on a Cisco router"
},
"correlation_display": {
"display_value": "",
"value": ""
},
"work_start": {
"display_value": "2015-07-06 11:56:04",
"value": "2015-07-06 18:56:04",
"display_value_internal": "2015-07-06 11:56:04"
},
"delivery_task": {
"display_value": "",
"value": ""
},
"outside_maintenance_schedule": {
"display_value": "false",
"value": false
},
"additional_assignee_list": {
"display_value": "",
"value": ""
},
"std_change_producer_version": {
"display_value": "Clear BGP sessions on a Cisco router - 1",
"value": "16c2273c47010200e90d87e8dee49006"
},
"sys_class_name": {
"display_value": "Change Request",
"value": "change_request"
},
"service_offering": {
"display_value": "",
"value": ""
}
"upon_approval": {
    "display_value": "Proceed to Next Task",
    "value": "proceed"
},
"correlation_id": {
    "display_value": "",
    "value": ""
},
"made_sla": {
    "display_value": "true",
    "value": true
},
"backout_plan": {
    "display_value": "Due to the limited number of commands in the implementation plan it is not possible to backout the change.\r\n\r\nIf required you are authorized to reboot the router if BGP fails to work",
    "value": "Due to the limited number of commands in the implementation plan it is not possible to backout the change.\r\n\r\nIf required you are authorized to reboot the router if BGP fails to work"
},
"conflict_status": {
    "display_value": "Not Run",
    "value": "Not Run"
},
"task_effective_number": {
    "display_value": "CHG0000024",
    "value": "CHG0000024"
},
"sys_updated_by": {
    "display_value": "admin",
    "value": "admin"
},
"opened_by": {
    "display_value": "System Administrator",
    "value": "6816f79cc0a8016401c5a33be04be441"
},
"user_input": {
    "display_value": "",
    "value": ""
},
"sys_created_on": {
    "display_value": "2015-07-06 11:55:46",
    "value": "2015-07-06 18:55:46",
    "display_value_internal": "2015-07-06 11:55:46"
"work_end": {
  "display_value": "2015-07-06 11:56:10",
  "value": "2015-07-06 18:56:10",
  "display_value_internal": "2015-07-06 11:56:10"
},
"phase_state": {
  "display_value": "Open",
  "value": "open"
},
"cab_date": {
  "display_value": "",
  "value": "",
  "display_value_internal": ""
},
"work_notes": {
  "display_value": "",
  "value": ""
},
"close_code": {
  "display_value": "Successful",
  "value": "successful"
},
"assignment_group": {
  "display_value": "Network",
  "value": "287ebd7da9fe198100f92cc8d1d2154e"
},
"description": {
  "display_value": "Resend the complete BGP table to neighboring routers
--Both neighbors need to support soft reset route refresh capability.
--Stores complete BGP table of you neighbor in router memory.
--Not a good idea on a peering router with full feed, due to the memory requirements.
  "value": "Resend the complete BGP table to neighboring routers
--Both neighbors need to support soft reset route refresh capability.
--Stores complete BGP table of you neighbor in router memory.
--Not a good idea on a peering router with full feed, due to the memory requirements."}
"on_hold_reason": {
  "display_value": "",
  "value": ""
},
"calendar_duration": {
  "display_value": "",
  "value": ""
}
"close_notes": {
    "display_value": "Completed without issues",
    "value": "Completed without issues"
},
"sys_id": {
    "display_value": "1766f1de47410200e90d87e8dee490f6",
    "value": "1766f1de47410200e90d87e8dee490f6"
},
"contact_type": {
    "display_value": "Phone",
    "value": "phone"
},
"cab_required": {
    "display_value": "false",
    "value": false
},
"urgency": {
    "display_value": "3 - Low",
    "value": 3.0
},
"scope": {
    "display_value": "Medium",
    "value": 3.0
},
"company": {
    "display_value": "",
    "value": ""
},
"justification": {
    "display_value": "",
    "value": ""
},
"activity_due": {
    "display_value": "UNKNOWN",
    "value": "",
    "display_value_internal": ""
},
"comments": {
    "display_value": "",
    "value": ""
},
"approval": {
    "display_value": "Approved",
    "value": "approved"
Change Management - GET /sn_chg_rest/change/{change_sys_id}/schedule

Enables retrieving the available time slots for a change request.

Role required: sn_change_writer

Note: Running this endpoint does not list the available start and end times. Use the link provided in the response body `worker.link` property to get the schedule data.

See also:
- GET /sn_chg_rest/change/ci/{cmdb_ci_sys_id}/schedule
- PATCH /sn_chg_rest/change/{change_sys_id}/schedule/first_available

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/{change_sys_id}/schedule

Default URL: /api/sn_chg_rest/change/{change_sys_id}/schedule

**Supported request parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>change_sys_id</td>
<td>Sys_id of the change request on which to find the next available time slot. Located in the [change_request] table. The selected change request must have a configuration item (cmdb_ci) with planned start and planned end times.</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>202</td>
<td>System accepted the request.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected. Possible issues:</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- The specified change request is missing the planned start and end times.</td>
</tr>
<tr>
<td></td>
<td>- The specified change request does not have an associated configuration item (cmdb_ci).</td>
</tr>
<tr>
<td></td>
<td>- User does not have read access to the fields of the change request.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td></td>
<td>Possible issues:</td>
</tr>
<tr>
<td></td>
<td>- System cannot find the change request based on information provided.</td>
</tr>
<tr>
<td></td>
<td>- User does not have read access to the record.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Information on any errors encountered while processing the endpoint request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;detail&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>error.detail</td>
<td>Additional information about the error.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>error.message</td>
<td>Message that identifies the error.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>messages</td>
<td>Message information.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;messages&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;errorMessages&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;infoMessages&quot;: [Array],</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>messages.errorMessages</td>
<td>Error messages encountered while processing the request. Data type: Array</td>
</tr>
<tr>
<td>messages.infoMessages</td>
<td>Information messages encountered while processing the request. Data type: Array</td>
</tr>
<tr>
<td>messages.warningMessages</td>
<td>Warning messages encountered while processing the request. Data type: Array</td>
</tr>
<tr>
<td>request</td>
<td>Original endpoint request. Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>Information on the current state of the worker. Data type: Object</td>
</tr>
</tbody>
</table>
| state.display_value         | Display value of the state of the worker. These values directly correlate to the state.value parameter. Possible values:  
• Complete  
• Error  
• In-Progress  
• Waiting  
Data type: String |
| state.value                 | Numeric value of the state of the worker. Possible values:  
• Complete  
• Error  
• In-Progress  
• Waiting  |

© 2021 ServiceNow, Inc. All rights reserved.

Servicenow, the servicenow logo, now, and other servicenow marks are trademarks and/or registered trademarks of servicenow, inc. in the united states and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>Indicates the type of request. Valid value: schedule</td>
</tr>
<tr>
<td>worker</td>
<td>Information about the associated worker. Data type: Object</td>
</tr>
<tr>
<td>worker.link</td>
<td>Link for retrieving time slot data. Use the sys_id in GET /sn_chg_rest/change/worker/{sys_id} to view results.</td>
</tr>
<tr>
<td>worker.sysId</td>
<td>Sys_id of the worker associated with the change request.</td>
</tr>
<tr>
<td>status</td>
<td>Only appears if an error is encountered. Status of the endpoint processing. Possible values: failure</td>
</tr>
</tbody>
</table>

**Get available time slots**

Use the value provided in the `worker.link` to get schedule window details. The value is in the following format:

https://instance.service-now.com/api/sn_chg_rest/change/worker/<worker_sys_id>
Use the worker_sys_id in GET /sn_chg_rest/change/worker/{worker_sys_id} to view results.

The response body contains the status and provides results when processing is complete.

Worker response body parameter results vary depending on time slot availability.

- If the provided time slot is available for the change request within the schedule time slot, the worker API lists the available time slots in the payload.spans property. The payload.spans property is not listed in the results otherwise.

- If there are no time slots available for change request duration provided within the defined scheduling time slot, the messages.infoMessages states the following:

  No slots found for <number> days from <date and time in format yyyy-mm-dd hh:mm:ss>

  🔄 Note: The change request scheduling time slot default value is 90 days. To change this value, modify the change.conflict.next_available.schedule_window property. For more information, see Configure conflict analysis properties.

The following GET /sn_chg_rest/change/worker/{sys_id} example shows output provided using the ID provided in the worker.link detail. The results list open time spans available for the task duration.

```json
{
  "result": {
    "worker": {
      "sysId": "9b3f62e0a4c87010f87712198fe9cad1",
      "link": "https://instance.service-now.com/api/snchg_rest/change/worker/9b3f62e0a4c87010f87712198fe9cad1"
    },
    "request": {
      "change_sys_id": "87ae5e900a0a2c3e263e8304e727c646",
      "timezone": "America/Los_Angeles"
    },
    "state": {
      "value": 3,
      "display_value": "Complete"
    },
    "type": "schedule",
    "messages": {
```
"errorMessages": [],
"warningMessages": [],
"infoMessages": []
},
"payload": {
"spans": [
{
"start": {
"value": "2021-05-08 08:00:00",
"display_value": "2021-05-08 01:00:00"
},
"end": {
"value": "2021-05-08 11:00:00",
"display_value": "2021-05-08 04:00:00"
}
},
{
"start": {
"value": "2021-05-15 08:00:00",
"display_value": "2021-05-15 01:00:00"
},
"end": {
"value": "2021-05-15 11:00:00",
"display_value": "2021-05-15 04:00:00"
}
},
...]
}
}

Example: cURL request

```bash
curl "https://instance.servicenow.com/api/sn_chg_rest/change/{change_sys_id}/schedule" \
--request GET \
--header "Accept:application/json" \
--user "username":"password"
```

Results include worker.link details you can use to run the provided sys_id in the GET /sn_chg_rest/change/worker/ endpoint.

```
{
"result": {
"worker": {
```
Change Management - GET /sn_chg_rest/change/{change_sys_id}/task

Retrieves one or more tasks associated with a specified change request based on the specified criteria.

**URL format**

**Versioned URL:** /api/sn_chg_rest/{api_version}/change/{change_sys_id}/task

**Default URL:** /api/sn_chg_rest/change/{change_sys_id}/task

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>change_sys_id</td>
<td>Sys_id of the change request whose tasks are to be retrieved. Located in the Change Request [change_request] table.</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key-value pairs</td>
<td>Fields to modify when creating the request. The key is the field name within the template and the value is the information to populate in the field. Fields that cannot be modified and are ignored if passed in:</td>
</tr>
<tr>
<td></td>
<td>• Business rules</td>
</tr>
<tr>
<td></td>
<td>• Read-only fields as defined in ACLs</td>
</tr>
<tr>
<td></td>
<td>• Fields that do not exist</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>order</td>
<td>Field by which to sort the returned change requests. Default: number</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. For requests that exceed this number of records, use the <strong>sysparm_offset</strong> parameter to paginate record retrieval.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Default: 500</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks.</td>
</tr>
<tr>
<td></td>
<td>For example, the first time you call this endpoint, <strong>sysparm_offset</strong> is set to &quot;0&quot;. To simply page through all available records, use <strong>sysparm_offset=sysparm_offset+sysparm_limit</strong>, until you reach the end of all records.</td>
</tr>
<tr>
<td></td>
<td>Do not pass a negative number in the <strong>sysparm_offset</strong> parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Default: 0</td>
</tr>
<tr>
<td>sysparm_query</td>
<td>Encoded query used to filter the result set.</td>
</tr>
<tr>
<td></td>
<td><strong>Syntax:</strong> <strong>sysparm_query=&lt;col_name&gt;&lt;operator&gt;&lt;value&gt;</strong>.</td>
</tr>
<tr>
<td></td>
<td>• <strong>&lt;col_name&gt;</strong>: Name of the table column to filter against.</td>
</tr>
<tr>
<td></td>
<td>• <strong>&lt;operator&gt;</strong>: Supports the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>=:</td>
<td>Exactly matches &lt;value&gt;.</td>
</tr>
<tr>
<td>!=:</td>
<td>Does not match &lt;value&gt;.</td>
</tr>
<tr>
<td>^:</td>
<td>Logically AND multiple query statements.</td>
</tr>
<tr>
<td>^OR:</td>
<td>Logically OR multiple query statements.</td>
</tr>
<tr>
<td>LIKE:</td>
<td>&lt;col_name&gt; contains the specified string. Only works for &lt;col_name&gt; fields whose data type is string.</td>
</tr>
<tr>
<td>STARTSWITH:</td>
<td>&lt;col_name&gt; starts with the specified string. Only works for &lt;col_name&gt; fields whose data type is string.</td>
</tr>
<tr>
<td>ENDSWITH:</td>
<td>&lt;col_name&gt; ends with the specified string. Only works for &lt;col_name&gt; fields whose data type is string.</td>
</tr>
<tr>
<td>&lt;value&gt;:</td>
<td>Value to match against.</td>
</tr>
</tbody>
</table>

All parameters are case-sensitive. Queries can contain more than one entry, such as
sysparm_query=<col_name><operator><value>[<operator><col_name><operator><value>]

For example:
(sysparm_query=caller_id=javascript:gs.getUserID()^active=true)

Encoded queries also supports order by functionality. To sort responses based on certain fields, use the ORDERBY and ORDERBYDESC clauses in sysparm_query.

Syntax:
- ORDERBY<col_name>
- ORDERBYDESC<col_name>

For example: sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory

This query filters all active records and orders the results in ascending order by number, and then in descending order by category.

If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property glide.invalid_query.returns_no_rows. Set this property to true to return no results for an invalid query.
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>textSearch</td>
<td>String to use to search all change task record fields. This search uses ServiceNow full text search platform functionality and defaults to IR_AND_OR_QUERY. Data type: String</td>
</tr>
</tbody>
</table>

**Note:** The glide.invalid_query.returns_no_rows property controls the behavior of all queries across the instance, such as in lists, scripts (GlideRecord.query()), and web service APIs.

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>All fields (key) with their associated values for the identified change request task prior to the delete.</td>
</tr>
<tr>
<td>parent</td>
<td>Information for the change request associated to the task.</td>
</tr>
<tr>
<td>parent.display_value</td>
<td>Change request information to display in a UI.</td>
</tr>
<tr>
<td>parent.value</td>
<td>Sys_id of the change request associated to the task.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id information for the change request task.</td>
</tr>
</tbody>
</table>

```json
parent: {
    display_value: "String",
    value: "String"
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id.display_value</td>
<td>Sys_id of the change request task to display in a UI. Data type: String</td>
</tr>
<tr>
<td>sys_id.value</td>
<td>Sys_id of the change request task. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```c
curl
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6ee11a9fe/task?sysparm_query=active=true"ORDERBYnumber"
--request GET
--header "Accept:application/json"
--user 'username':'password'
```

```json
{
  result: [
    {
      sys_id: {
        value: "12629ec4b750230096c3e4f6ee11a9d5",
        display_value: "12629ec4b750230096c3e4f6ee11a9d5"
      },
      parent: {
        value: "0f4ac6c4b750230096c3e4f6ee11a9fe",
        display_value: "CHG0033046"
      },
      ... // all valid fields in record, example below
      short_description: {
        value: "Retire node",
        display_value: "Retire node"
      }
    },
    ... // next record found }
  ]
}
```

**Python request**

```python
# Install requests package for python
import requests
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6ee11a9fe/task?sysparm_query=active=true&ORDERBYnumber'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <sys_id>
      <value>12629ec4b750230096c3e4f6ee11a9d5</value>
      <display_value>12629ec4b750230096c3e4f6ee11a9d5</display_value>
    </sys_id>
    <parent>
      <value>0f4ac6c4b750230096c3e4f6ee11a9fe</value>
      <display_value>CHG0033046</display_value>
    </parent>

    // all valid fields in record, single parameter example below
    <short_description>
      <value>Retire node</value>
      <display_value>Retire node</display_value>
    </short_description>
  </result>
</response>
Change Management - GET /sn_chg_rest/change/{sys_id}
Retrieves the change request identified by the specified sys_id.

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/{sys_id}
Default URL: /api/sn_chg_rest/change/{sys_id}

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request record to retrieve from Change Request [change_request] table. Data type: String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Request body parameters (XML or JSON)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>
### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>List containing one or more change request record objects. Each object describes a change request. Each element in the change request object corresponds to a field in its associated record in the Change Request [change_request] table.</td>
</tr>
<tr>
<td></td>
<td>All elements contain <code>value</code> and <code>display_value</code> name-value pairs. Date fields also contain <code>display_value_internal</code> name-value pairs.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>action_status</td>
<td>Current action status of the associated change request.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Blocked internally</td>
</tr>
<tr>
<td></td>
<td>• 2: Blocked by customer</td>
</tr>
<tr>
<td></td>
<td>• 3: Blocked internally and by customer</td>
</tr>
<tr>
<td></td>
<td>• 4: Needs attention</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>active</td>
<td>Flag that indicates whether the change request is active.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Change request is active</td>
</tr>
<tr>
<td></td>
<td>• false: Change request is not active</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td>activity_due</td>
<td>Date and time for which the associated case is expected to be completed.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>additional_assignee_list</td>
<td>List of sys_ids of additional persons assigned to work on the change request. Data type: Array</td>
</tr>
<tr>
<td>approval</td>
<td>Type of approval process required. Data type: String Default: not requested</td>
</tr>
<tr>
<td>approval_history</td>
<td>Most recent approval history journal entry. Data type: String</td>
</tr>
<tr>
<td>approval_set</td>
<td>Date and time that the associated action was approved. Data type: String</td>
</tr>
<tr>
<td>assigned_to</td>
<td>Sys_id of the user assigned to the change request. Data type: String</td>
</tr>
<tr>
<td>assignment_group</td>
<td>Sys_id of the group assigned to the change request. Data type: String</td>
</tr>
<tr>
<td>backout_plan</td>
<td>Description of the plan to execute if the change must be reversed. Data type: String</td>
</tr>
<tr>
<td>business_duration</td>
<td>Length in scheduled work hours, work days, and work weeks that it took to complete the change. Data type: String</td>
</tr>
<tr>
<td>business_service</td>
<td>Sys_id of the business service associated with the change request. Located in the Service [cmdb_ci_service] table. Data type: String</td>
</tr>
<tr>
<td>cab_date</td>
<td>Date on which the Change Advisory Board (CAB) meets. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>cab_delegate</td>
<td>Sys_id of the user that can substitute for the CAB manager during a CAB meeting. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>cab_recommendation</td>
<td>Description of the CAB recommendations for the change request. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>cab_required</td>
<td>Flag that indicates whether the CAB is required. Possible values: • true: Change Advisory Board is required. • false: Change Advisory Board is not required. Data type: Boolean Default: false</td>
</tr>
<tr>
<td>calendar_duration</td>
<td>Not currently used by Change Management. Data type: String</td>
</tr>
<tr>
<td>category</td>
<td>Category of the change, for example hardware, network, or software. Data type: String Default: Other</td>
</tr>
<tr>
<td>change_plan</td>
<td>Activities and roles for managing and controlling the change request. Data type: String</td>
</tr>
<tr>
<td>chg_model</td>
<td>Sys_id of the change model that the associated change request was based on. Located in the Change Model [chg_model] table. The Change Model defines the state flow, transitions, and process activities that must be completed for the change request.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>closed_at</strong></td>
<td>Date and time that the associated change request was closed.</td>
</tr>
<tr>
<td><strong>closed_by</strong></td>
<td>Sys_id of the person that closed the change request. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td><strong>close_code</strong></td>
<td>Code assigned to the change request when it was closed. For example, Successful, Successful with issues, and Unsuccessful.</td>
</tr>
<tr>
<td><strong>close_notes</strong></td>
<td>Notes that the person entered when closing the change request.</td>
</tr>
<tr>
<td><strong>cmdb_ci</strong></td>
<td>Sys_id of the configuration item associated with the change request. Located in the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td><strong>comments</strong></td>
<td>List of customer-facing work notes entered in the associated change request.</td>
</tr>
<tr>
<td><strong>comments_and_work_notes</strong></td>
<td>List of both internal and customer-facing work notes entered for the associated change request.</td>
</tr>
<tr>
<td><strong>company</strong></td>
<td>Sys_id of the company associated with the change request. Located in the Company [core_company] table.</td>
</tr>
<tr>
<td><strong>conflict_last_run</strong></td>
<td>Date and time that the conflict detection script was last run on the change request.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>conflict_status</td>
<td>Current conflict status as detected by the conflict detection script, such as Conflict and Not Run.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: Not Run</td>
</tr>
<tr>
<td>contact_type</td>
<td>Method in which the change request was initially requested. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• chat</td>
</tr>
<tr>
<td></td>
<td>• email</td>
</tr>
<tr>
<td></td>
<td>• phone</td>
</tr>
<tr>
<td></td>
<td>• social</td>
</tr>
<tr>
<td></td>
<td>• web</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>contract</td>
<td>Sys_id of the contract associated with the change request. Located in the Contract [ast_contract] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>correlation_display</td>
<td>User-friendly name for the correlation_id.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>correlation_id</td>
<td>Globally unique ID (GUID) of a matching change request record in a third-party system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>delivery_plan</td>
<td>No longer in use. Sys_id of the delivery plan associated with the change request. Located in the Execution Plan [sc_cat_item_delivery_plan] table.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>delivery_task</td>
<td>No longer in use. Sys_id of the delivery task associated with the change request. Located in the Execution Plan Task [sc_cat_item_delivery_task] table.</td>
</tr>
<tr>
<td>description</td>
<td>Detailed description of the change request.</td>
</tr>
<tr>
<td>due_date</td>
<td>Task due date. Not used by change request process.</td>
</tr>
<tr>
<td>end_date</td>
<td>Date and time when the change request is to be completed.</td>
</tr>
<tr>
<td>escalation</td>
<td>Current escalation level. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 0: Normal</td>
</tr>
<tr>
<td></td>
<td>• 1: Moderate</td>
</tr>
<tr>
<td></td>
<td>• 2: High</td>
</tr>
<tr>
<td></td>
<td>• 3: Overdue</td>
</tr>
<tr>
<td>expected_start</td>
<td>Date and time when the task is to start. Not used by the change request process.</td>
</tr>
<tr>
<td>follow_up</td>
<td>Date and time when a user followed-up with the person requesting the change request.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>group_list</td>
<td>List of sys_ids and names of the groups associated with the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>impact</td>
<td>Impact on the change request will have on the customer.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: High</td>
</tr>
<tr>
<td></td>
<td>• 2: Medium</td>
</tr>
<tr>
<td></td>
<td>• 3: Low</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 3</td>
</tr>
<tr>
<td>implementation_plan</td>
<td>Sequential steps to execute to implement this change. It also contains any dependencies between steps and assignee details for each step.</td>
</tr>
<tr>
<td>justification</td>
<td>Benefits of implementing this change and the impact if this change is not implemented.</td>
</tr>
<tr>
<td>knowledge</td>
<td>Flag that indicates whether there are any knowledge base (KB) articles associated with the change request.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Associated KB articles</td>
</tr>
<tr>
<td></td>
<td>• false: No associated KB articles</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>location</td>
<td>Sys_id and name of the location of the equipment referenced in the change request. Located in the Location Located in the Location [cmn_location] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>made_sla</td>
<td>No longer used. Flag that indicates whether the change request was implemented in alignment with the associated service level agreement.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>needs_attention</td>
<td>Flag that indicates whether the change request needs attention.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Change request needs additional attention.</td>
</tr>
<tr>
<td></td>
<td>• false: Change request does not need additional attention.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>number</td>
<td>Change number assigned to the change request by the system, such as CHG0040007.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>on_hold</td>
<td>Flag that indicates whether the change request is currently on hold.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: On hold</td>
</tr>
<tr>
<td></td>
<td>• false: Not on hold</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>on_hold_reason</td>
<td>If the on_hold parameter is &quot;true&quot;, description of the reason why the change request is being held up.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>on_hold_task</td>
<td>If the on_hold parameter is &quot;true&quot;, list of the sys_ids of the tasks that must be completed before the hold is released.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>opened_at</td>
<td>Date and time that the change release was created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>opened_by</td>
<td>Sys_id and name of the user that created the change release. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>order</td>
<td>Not used by Change Management. Optional numeric field by which to order records, such as when retrieving them from a database.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td>outside_maintenance_schedule</td>
<td>Flag that indicates whether maintenance by an outside company has been schedule for the change request.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Outside maintenance scheduled</td>
</tr>
<tr>
<td></td>
<td>• false: No outside maintenance scheduled</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent</td>
<td>Sys_id and name of the parent task to this change request, if any. Located in the Task [task] table. Data type: String</td>
</tr>
<tr>
<td>phase</td>
<td>Current phase of the change request. This defines what the change is doing in greater detail. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• accept</td>
</tr>
<tr>
<td></td>
<td>• build</td>
</tr>
<tr>
<td></td>
<td>• plan</td>
</tr>
<tr>
<td></td>
<td>• requested</td>
</tr>
<tr>
<td>phase_state</td>
<td>Change_phase records that should be created for a change. They are dependent on the category, such that each type of change may have different change_phase records. The change_phase records provide an opportunity to control the approval process as each change_phase can have a schedule and a set of approvers. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• complete</td>
</tr>
<tr>
<td></td>
<td>• on hold</td>
</tr>
<tr>
<td></td>
<td>• open</td>
</tr>
<tr>
<td></td>
<td>• rejected</td>
</tr>
<tr>
<td></td>
<td>• requested</td>
</tr>
<tr>
<td></td>
<td>• work in progress</td>
</tr>
<tr>
<td>priority</td>
<td>Priority of the change request. Possible values:</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
|                  | • 1: Critical  
|                  | • 2: High  
|                  | • 3: Moderate  
|                  | • 4: Low  
| Data type: Number (Integer) | Default: 4 |
| production_system  | Flag that indicates whether the change request is for a ServiceNow instance that is in a production environment.  
|                  | Possible values:  
|                  | • true: Production environment  
|                  | • false: Non-production environment  
| Data type: Boolean | |
| reason  | Description of why the change request was initiated.  
|                  | Possible values:  
|                  | • Business requirements  
|                  | • Hardware upgrade  
|                  | • Legislation  
|                  | • Location change  
|                  | • Network requirements  
|                  | • New or removed CI  
|                  | • Other  
|                  | • Problem resolved  
|                  | • Product or service changed  
|                  | • Software upgrade  
|                  | • User requested  
<p>| Data type: String | Maximum length: 40 |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reassignment_count</td>
<td>Number of times that the change request has been reassigned to a new owner. Data type: Number (Integer) Default: 0</td>
</tr>
<tr>
<td>rejection_goto</td>
<td>Sys_id of the task to perform if the change request is rejected. Located in the Task [table]. Data type: String</td>
</tr>
<tr>
<td>requested_by</td>
<td>Sys_id of the user that requested the change. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>requested_by_date</td>
<td>Date and time when the change is requested to be implemented by. Data type: String</td>
</tr>
<tr>
<td>review_comments</td>
<td>Comments entered when the change request was reviewed. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>review_date</td>
<td>Date that the change request was reviewed. Data type: String</td>
</tr>
<tr>
<td>review_status</td>
<td>Current status of the requested change request review. Data type: String</td>
</tr>
<tr>
<td>risk</td>
<td>Level of risk associated with the change request. Valid values: 1: High 2: Moderate 3: Low Data type: Number</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Default: 3</td>
<td>Description of the risk and analysis of implementing the change request. Data type: String Maximum length: 4,000</td>
</tr>
</tbody>
</table>
| risk_impact_analysis | Not currently used by Change Management. Reason that the change request was transferred. Possible values:  
  • 1: Transfer with Resolution  
  • 9: Transfer without Resolutions Data type: Number |
| scope              | Size of the change request. Possible values:  
  • 1: Massive  
  • 2: Large  
  • 3: Medium  
  • 4: Small  
  • 5: Tiny Data type: Number Default: 3 |
<p>| service_offering   | Sys_id of the service offering associated with the change request. Service offerings uniquely define the level of service in terms of availability, scope, pricing, and packaging options. Located in the Offering [service_offering] table. Data type: String |
| short_description  | Description of the change request. Data type: String Maximum length: 40 |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>skills</td>
<td>List of the sys_ids of all of the skills required to implement the change request. Located in the Skill [cmn_skill] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>sla_due</td>
<td>No longer in use. Date and time that the change request must be completed based on the associated service level agreement.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>start_date</td>
<td>Date and time that the change request is planned to start implementation.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>Current state of the change request. Possible values are defined in the change model.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 1</td>
</tr>
<tr>
<td>std_change_producer_version</td>
<td>Sys_id of the record producer and change proposal associated with the change request. It also includes the number and percentage of successful and unsuccessful change requests created from the proposal. Located in the Standard Change Template Version [std_change_producer_version] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sys_class_name</td>
<td>Name of the table in which the change request is located.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_created_by</td>
<td>Name of the user that initially created the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sys_created_on</td>
<td>Date and time that the associated change request record was originally created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_domain</td>
<td>If using domains in the instance, the name of the domain to which the change module record is associated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_domain_path</td>
<td>If using domains in the instance, the domain path in which the associated change module record resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Unique identifier of the associated change request record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_mod_count</td>
<td>Number of updates to the case since it was initially created.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td>sys_updated_by</td>
<td>Person that last updated the case.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sys_updated_on</td>
<td>Date and time when the case was last updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>task_effective_number</td>
<td>Universal request number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>task_for</td>
<td>Not used by Change Management. Sys_id of the user that the task was created for. Located in the User [sys_user] table. Data type: String</td>
</tr>
<tr>
<td>test_plan</td>
<td>Description of the associated test plan for the change. Data type: String Maximum length: 4,000</td>
</tr>
<tr>
<td>time_worked</td>
<td>Total amount of time worked on the change request. Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Change request type. Possible values: emergency normal standard Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>unauthorized</td>
<td>Flag that indicates whether the change request is unauthorized Possible values: true: Unauthorized false: Authorized Data type: Boolean</td>
</tr>
<tr>
<td>universal_request</td>
<td>Sys_id of the Parent Universal request to which this change request is a part of. Located in the Task [task] table. Data type: String</td>
</tr>
<tr>
<td>uponApproval</td>
<td>Action to take if the change request is approved.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• do_nothing</td>
</tr>
<tr>
<td></td>
<td>• proceed</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Maximum length: 40</td>
<td></td>
</tr>
<tr>
<td>Default: proceed</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>upon_reject</th>
<th>Action to take if the change request is rejected. Possible values:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• cancel</td>
</tr>
<tr>
<td></td>
<td>• goto</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Maximum length: 40</td>
<td></td>
</tr>
<tr>
<td>Default: cancel</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>urgency</th>
<th>Urgency of the change request. Possible values:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• 1: High</td>
</tr>
<tr>
<td></td>
<td>• 2: Medium</td>
</tr>
<tr>
<td></td>
<td>• 3: Low</td>
</tr>
<tr>
<td>Data type: Number (Integer)</td>
<td></td>
</tr>
<tr>
<td>Default: 3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>user_input</th>
<th>Additional user input.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Maximum length: 4,000</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>variables</th>
<th>Name-value pairs of variables associated with the change request.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Maximum length: 40</td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>watch_list</td>
<td>List of sys_ids of the users who receive notifications about this change request when additional comments are added or if the state of a change request is changed to Resolved or Closed. Located in the User [sys_user] table. Data type: Array</td>
</tr>
<tr>
<td>wf_activity</td>
<td>Sys_id of the workflow activity record associated with the change request. Located in the Workflow Activity [wf_activity] table. Data type: String</td>
</tr>
<tr>
<td>work_end</td>
<td>Date and time work ended on the change request. Data type: String</td>
</tr>
<tr>
<td>work_notes</td>
<td>Information about how to resolve the change request, or steps taken to resolve it. Data type: String</td>
</tr>
<tr>
<td>work_notes_list</td>
<td>List of sys_ids of the internal users who receive notifications about this change request when work notes are added. Located in the User [sys_user] table. Data type: Array</td>
</tr>
<tr>
<td>work_start</td>
<td>Date and time that work started on the change request. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/b0dbda5347c12200e0ef563dbb9a718f"
  --request GET
  --header "Accept:application/json"
  --user "username":"password"
```
"value": "",
"display_value_internal": ""
},
"requested_by": {
"display_value": "System Administrator",
"value": "6816f79cc0a8016401c5a33be04be441"
},
"business_duration": {
"display_value": "",
"value": ""
},
"group_list": {
"display_value": "",
"value": ""
},
"change_plan": {
"display_value": "",
"value": ""
},
"approval_set": {
"display_value": "",
"value": ""
},
"wf_activity": {
"display_value": "",
"value": ""
},
"implementation_plan": {
"display_value": "-- Place router into maintenance mode in the monitoring platform
-- Logon to router through SSH
-- Run the following command
router(config-router)#router bgp 12345
router(config-router)#neighbor [neighbor ip] soft-reconfig [inbound]
r#clear ip bgp [neighbor ip] soft
-- Confirm the sessions have been cleared
-- Place router back into operational mode in the monitoring platform",
"value": "-- Place router into maintenance mode in the monitoring platform
-- Logon to router through SSH
-- Run the following command
router(config-router)#neighbor [neighbor ip] soft-reconfig [inbound]
r#clear ip bgp [neighbor ip] soft
-- Confirm the sessions have been cleared
-- Place router back into operational mode in the monitoring platform",
"universal_request": {
"display_value": "",
"value": ""
}
"display_value": "Proceed to Next Task",
"value": "proceed"
},
"correlation_id": {
"display_value": "",
"value": ""
},
"made_sla": {
"display_value": "true",
"value": true
},
"backout_plan": {
"display_value": "Due to the limited number of commands in the implementation plan it is not possible to backout the change. If required you are authorized to reboot the router if BGP fails to work",
"value": "Due to the limited number of commands in the implementation plan it is not possible to backout the change. If required you are authorized to reboot the router if BGP fails to work"
},
"conflict_status": {
"display_value": "Not Run",
"value": "Not Run"
},
"task_effective_number": {
"display_value": "CHG0000024",
"value": "CHG0000024"
},
"sys_updated_by": {
"display_value": "admin",
"value": "admin"
},
"opened_by": {
"display_value": "System Administrator",
"value": "6816f79cc0a8016401c5a33be04be441"
},
"user_input": {
"display_value": "",
"value": ""
},
"sys_created_on": {
"display_value": "2015-07-06 11:55:46",
"value": "2015-07-06 18:55:46",
"display_value_internal": "2015-07-06 11:55:46"}
"on_hold_task": {
  "display_value": "",
  "value": ""
},
"sys_domain": {
  "display_value": "global",
  "value": "global"
},
"route_reason": {
  "display_value": "",
  "value": ""
},
"closed_at": {
  "display_value": "2015-07-06 11:56:23",
  "value": "2015-07-06 18:56:23",
  "display_value_internal": "2015-07-06 11:56:23"
},
"review_comments": {
  "display_value": "",
  "value": ""
},
"business_service": {
  "display_value": "",
  "value": ""
},
"time_worked": {
  "display_value": "",
  "value": ""
},
"chg_model": {
  "display_value": "",
  "value": ""
},
"expected_start": {
  "display_value": "",
  "value": "",
  "display_value_internal": ""
},
"opened_at": {
  "display_value": "2015-06-09 11:55:46",
  "value": "2015-06-09 18:55:46",
  "display_value_internal": "2015-06-09 11:55:46"
},
"work_end": {

Resend the complete BGP table to neighboring routers.

--Both neighbors need to support soft reset route refresh capability.
--Stores complete BGP table of your neighbor in router memory.
--Not a good idea on a peering router with full feed, due to memory requirements.
"display_value": "Completed without issues",
"value": "Completed without issues"
},
"sys_id": {
"display_value": "1766f1de47410200e90d87e8dee490f6",
"value": "1766f1de47410200e90d87e8dee490f6"
},
"contact_type": {
"display_value": "Phone",
"value": "phone"
},
"cab_required": {
"display_value": "false",
"value": false
},
"urgency": {
"display_value": "3 - Low",
"value": 3.0
},
"scope": {
"display_value": "Medium",
"value": 3.0
},
"company": {
"display_value": "",
"value": ""
},
"justification": {
"display_value": "",
"value": ""
},
"activity_due": {
"display_value": "UNKNOWN",
"value": "",
"display_value_internal": ""
},
"comments": {
"display_value": "",
"value": ""
},
"approval": {
"display_value": "Approved",
"value": "approved"
}
"due_date": {
    "display_value": "",
    "value": "",
    "display_value_internal": ""
},
"sys_mod_count": {
    "display_value": "10",
    "value": 10.0
},
"on_hold": {
    "display_value": "false",
    "value": false
},
"sys_tags": {
    "display_value": "",
    "value": ""
},
"conflict_last_run": {
    "display_value": "",
    "value": "",
    "display_value_internal": ""
},
"risk_value": {
    "display_value": "",
    "value": ""
},
"unauthorized": {
    "display_value": "false",
    "value": false
},
"risk": {
    "display_value": "Moderate",
    "value": 3.0
},
"location": {
    "display_value": "",
    "value": ""
},
"category": {
    "display_value": "Other",
    "value": "Other"
},
"risk_impact_analysis": {
    "display_value": ""}
Change Management - GET /sn_chg_rest/change/{sys_id}/ci
Retrieves multiple configuration items (CIs) associated to a specified change request based on the specified association type.

URL format
Versioned URL: /api/sn_chg_rest/{api_version}/change/{sys_id}/ci
Default URL: /api/sn_chg_rest/change/{sys_id}/ci

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request for which to return the associated CMDB CIs. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>association_type</td>
<td>Required. Type of association between the CMDB CI and the change request. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• affected: CIs that are affected by the change request</td>
</tr>
<tr>
<td></td>
<td>• impacted: Services impacted by the change request</td>
</tr>
<tr>
<td></td>
<td>• offering: Impacted service offerings</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>name-value pairs</td>
<td>Name-value pairs to use to filter the result set. The name is the field on which the specified value is filtered. This parameter is mutually exclusive with <code>sysparm_query</code>. For example, instead of using <code>&amp;sysparm_query=active=true</code>, you can simplify the calling statement by using <code>&amp;active=true</code>. You can also use the display value for the field if it is a choice or reference type field, such as <code>&amp;state=closed</code> instead of <code>&amp;state=7</code>. To specify multiple key-value pairs, separate each with an ampersand, such as <code>&amp;active=true&amp;assigned_to=john.smith</code>. Data type: String</td>
</tr>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. For requests that exceed this number of records, use the <code>sysparm_offset</code> parameter to paginate record retrieval. Data type: Number Default: 500</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, <code>sysparm_offset</code> is set to &quot;0&quot;. Simply page through all available records, use <code>sysparm_offset=sysparm_offset+sysparm_limit</code>, until you reach the end of all records. Do not pass a negative number in the <code>sysparm_offset</code> parameter. Data type: Number Default: 0</td>
</tr>
<tr>
<td>sysparm_query</td>
<td>Encoded query used to filter the result set. Syntax: <code>sysparm_query=&lt;col_name&gt;&lt;operator&gt;&lt;value&gt;</code>. ----------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;col_name&gt;</code>: Name of the table column to filter against.</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;operator&gt;</code>: Supports the following values:</td>
</tr>
<tr>
<td></td>
<td>  ◦ <code>=</code>: Exactly matches <code>&lt;value&gt;</code>.</td>
</tr>
<tr>
<td></td>
<td>  ◦ <code>!</code>: Does not match <code>&lt;value&gt;</code>.</td>
</tr>
<tr>
<td></td>
<td>  ◦ <code>^</code>: Logically AND multiple query statements.</td>
</tr>
<tr>
<td></td>
<td>  ◦ <code>^OR</code>: Logically OR multiple query statements.</td>
</tr>
<tr>
<td></td>
<td>  ◦ <code>LIKE</code>: <code>&lt;col_name&gt;</code> contains the specified string. Only works for <code>&lt;col_name&gt;</code> fields whose data type is string.</td>
</tr>
</tbody>
</table>
## Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STARTSWITH: &lt;col_name&gt;</td>
<td>starts with the specified string. Only works for &lt;col_name&gt; fields whose data type is string.</td>
</tr>
<tr>
<td>ENDSWITH: &lt;col_name&gt;</td>
<td>ends with the specified string. Only works for &lt;col_name&gt; fields whose data type is string.</td>
</tr>
<tr>
<td>&lt;value&gt;</td>
<td>Value to match against.</td>
</tr>
</tbody>
</table>

All parameters are case-sensitive. Queries can contain more than one entry, such as:
```
sysparm_query=<col_name><operator><value>[<operator><col_name><operator><value>]
```
For example:
```
(sysparm_query=caller_id=javascript:gs.getUserID()^active=true)
```
Encoded queries also supports order by functionality. To sort responses by certain fields, use the `ORDERBY` and `ORDERBYDESC` clauses in `sysparm_query`.

**Syntax:**
- `ORDERBY<col_name>`
- `ORDERBYDESC<col_name>`

For example: `sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory`

This query filters all active records and orders the results in ascending order by number, and then in descending order by category.

If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property `glide.invalid_query.returns_no_rows`. Set this property to true to return no rows on an invalid query.

ℹ️ **Note:** The `glide.invalid_query.returns_no_rows` property controls the behavior of all queries across the instance, such as in lists, scripts (`GlideRecord.query()`), and web service APIs.

Data type: String
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

#### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected. The error response contains pertinent messages to help troubleshoot the problem.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>`ci_item</td>
<td>cmdb_ci_service`</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>`ci_item</td>
</tr>
<tr>
<td>`ci_item</td>
<td>cmdb_ci_service.value`</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>`ci_item</td>
<td>cmdb_ci_service.display_value`</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>sys_id</code></td>
<td>Change request sys_id information.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td><code>sys_id: { display_value: &quot;String&quot;, value: &quot;String&quot; }</code></td>
</tr>
<tr>
<td><code>sys_id.value</code></td>
<td>Sys_id of the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>sys_id.display_value</code></td>
<td>Sys_id of the change request to display in a UI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&lt;record_fields&gt;</td>
<td>All valid fields in the ci_item or cmdb_ci_service record; table based on association type. Data type: Object</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
   "https://instance.servicenow.com/api/sn_chg_rest/v1/change/c286d61347c12200e0ef563dbb9a71df/ci?association_type=affected"
   --request GET
   --header "Accept:application/json"
   --user 'username':'password'

{
   result: [
     {
       sys_id: {
           value: "92b8544047810200e90d87e8dee490b0",
           display_value: "92b8544047810200e90d87e8dee490b0"
       },
       ci_item|cmdb_ci_service : {
           value: "3a27d4370a0a0bb4006316812bf45439",
           display_value: "PS Apache01"
       },
       ..., // all valid fields in record, table based on association type
     },
     ..., // next record found }, ... // and so on
   ]
}
```

**Example: Python request**

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/c286d61347c12200e0ef563dbb9a71df/ci?association_type=affected'

# Set the user credentials
user = 'username'
```
pwd = 'password'

# Set the proper headers
headers = {"Accept":"application/json"}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    result: [
        {
            sys_id: {
                value: "92b8544047810200e90d87e8dee490b0",
                display_value: "92b8544047810200e90d87e8dee490b0"
            },
            ci_item|cmdb_ci_service : {
                value: "3a27d4370a0a0bb4006316812bf45439",
                display_value: "PS Apache01"
            },
            ... // all valid fields in record, table based on association type
        },
        { // next record found }, ... // and so on
    ]
}

**Change Management - GET /sn_chg_rest/change/{sys_id}/conflict**

Retrieves the status of the currently running change request conflict checking process or the results of the last completed conflict checking process for the specified change request.

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/{sys_id}/conflict
Default URL: /api/sn_chg_rest/change/{sys_id}/conflict

### Supported request parameters

#### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request for which to return the status of the running/completed conflict checking process. Located in the</td>
</tr>
<tr>
<td></td>
<td>Change Request [change_request] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

#### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

#### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Current status of the conflict checking process; including conflicts if any are detected.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>conflicts</td>
<td>List of conflicts found for the change request. An empty object indicates that no conflicts were detected. Data type: Array</td>
</tr>
<tr>
<td>job_status</td>
<td>Status of the actual conflict checking job. Data type: String</td>
</tr>
<tr>
<td>last_run</td>
<td>Date and time the last conflict checking process started. Data type: String</td>
</tr>
<tr>
<td>record_count</td>
<td>Number of records checked. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>status</td>
<td>Outcome of the conflict checking process, such as &quot;Conflict&quot; or &quot;Not run&quot;.</td>
</tr>
</tbody>
</table>

**Note:** Even if the change request does not have any conflicts, this field is set to "Conflict." However, the `conflicts` object is empty.

Data type: String

### Example: cURL request

**curl**

```bash
curl "https://instance.servicenow.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6e11a9fe/conflict" \   --request GET \   --header "Accept:application/json" \   --user "username":"password"
```

**Successful response - no conflicts**

```json
{
    result: {
        status: "Conflict",
        last_run": "2018-08-30 12:58:05",
        record_count: "1",
        job_status: "2",
        conflicts: []
    }
}
```

### Example: cURL request

**curl**

```bash
curl "https://instance.servicenow.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6e11a9fe/conflict" \   --request GET \   --header "Accept:application/json" \   --user "username":"password"
```

**Successful response - with conflicts**

```json
{
    result: {
        status: "Conflict",
        last_run": "2018-08-30 12:58:05",
```
record_count: "1",
job_status: "2",
conflicts: [
  {
    change: {
      display_value: "CHG0030001",
      value: "afbffb24b758230096c3e4f6ee11a972"
    },
    type: {
      display_value: "Not In Maintenance Window",
      value: "not_in_maintenance_window"
    }
  }]

Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.service-now.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6ee11a9fe/conflict'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Accept":"application/xml"}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()
```
# Decode the XML response into a dictionary and use the data
print(response.content)

Successful response - no conflicts

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
    <result>
        <status>Conflict</status>
        <last_run>2018-08-30 12:58:05</last_run>
        <record_count>1</record_count>
        <job_status>2</job_status>
        <conflicts/>
    </result>
</response>
```

Change Management - GET /sn_chg_rest/change/emergency

Retrieves one or more emergency change requests based on the specified criteria.

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/emergency

Default URL: /api/sn_chg_rest/change/emergency

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>order</td>
<td>Field by which to sort the returned change requests. Default: number Data type: String Default: number</td>
</tr>
</tbody>
</table>
**Query parameters (continued)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. For requests that exceed this number of records, use the <code>sysparm_offset</code> parameter to paginate record retrieval. Data type: Number Default: 500</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, <code>sysparm_offset</code> is set to &quot;0&quot;. To simply page through all available records, use <code>sysparm_offset=sysparm_offset+sysparm_limit</code>, until you reach the end of all records. Do not pass a negative number in the <code>sysparm_offset</code> parameter. Data type: Number Default: 0</td>
</tr>
</tbody>
</table>
| sysparm_query      | Encoded query used to filter the result set. **Syntax:** `sysparm_query=<col_name><operator><value>`.  
  - `<col_name>`: Name of the table column to filter against.  
  - `<operator>`: Supports the following values:  
    - `=`: Exactly matches `<value>`.  
    - `!`: Does not match `<value>`.  
    - `^`: Logically AND multiple query statements.  
    - `^OR`: Logically OR multiple query statements.  
    - `LIKE`: `<col_name>` contains the specified string. Only works for `<col_name>` fields whose data type is string.  
    - `STARTSWITH`: `<col_name>` starts with the specified string. Only works for `<col_name>` fields whose data type is string.  
    - `ENDSWITH`: `<col_name>` ends with the specified string. Only works for `<col_name>` fields whose data type is string.  
  - `<value>`: Value to match against. All parameters are case-sensitive. Queries can contain more than one entry, such as  
    ```
    sysparm_query=<col_name><operator><value>[<operator><col_name><operator><value>]
    ```
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>For example: (\text{sysparm_query} = \text{caller_id} = \text{javascript:gs.getUserID()} \land \text{active} = \text{true})</td>
</tr>
<tr>
<td></td>
<td>Encoded queries also supports order by functionality. To sort responses based on certain fields, use the ORDERBY and ORDERBYDESC clauses in \text{sysparm_query}.</td>
</tr>
<tr>
<td>Syntax</td>
<td>• ORDERBY&lt;col_name&gt;</td>
</tr>
<tr>
<td></td>
<td>• ORDERBYDESC&lt;col_name&gt;</td>
</tr>
<tr>
<td>For example</td>
<td>(\text{sysparm_query} = \text{active} = \text{true} \land \text{ORDERBYnumber} \land \text{ORDERBYDESC} \text{category})</td>
</tr>
<tr>
<td></td>
<td>This query filters all active records and orders the results in ascending order by number, and then in descending order by category.</td>
</tr>
<tr>
<td></td>
<td>If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property glide.invalid_query.returns_no_rows. Set this property to true to return no rows on an invalid query.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>The glide.invalid_query.returns_no_rows property controls the behavior of all queries across the instance, such as in lists, scripts (\text{GlideRecord_query()}), and web service APIs.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>key-value pairs</td>
<td>Name-value pairs to use to filter the result set. The name is the field on which the specified value is filtered. This parameter is mutually exclusive with \text{sysparm_query}. For example, instead of using &amp;sysparm=query=active=true, you can simplify the calling statement by using &amp;active=true. You can also use the display value of the field is a choice or reference type field, such as &amp;state=closed instead of &amp;state=7. To specify multiple key-value pairs, separate each with an ampersand, such as &amp;active=true&amp;assigned_to=john.smith.</td>
</tr>
<tr>
<td>textSearch</td>
<td>String to use to search all emergency change request record fields. This search uses ServiceNow full text search platform functionality and defaults to IR_AND_OR_QUERY.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>
# Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Values for all fields in the associated change request. Data type: Object</td>
</tr>
<tr>
<td>state</td>
<td>State of the change request. Data type: Object</td>
</tr>
<tr>
<td>state.display_value</td>
<td>State to display in a UI. Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Internal state value. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id information for the change request. Data type: Object</td>
</tr>
<tr>
<td>sys_id.display_value</td>
<td>Sys_id of the change request to display in a UI. Data type: String</td>
</tr>
<tr>
<td>sys_id.value</td>
<td>Sys_id of the change request. Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Type of change request. Data type: Object</td>
</tr>
<tr>
<td>type.display_value</td>
<td>Change type to display in a UI.</td>
</tr>
</tbody>
</table>

```json
state: {
  display_value: "String",
  value: "String"
}
```

```json
sys_id: {
  display_value: "String",
  value: "String"
}
```

```json
type: {
  display_value: "String",
  value: "String"
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Value is always &quot;Emergency&quot;.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>type.value</td>
<td>Internal type value.</td>
</tr>
<tr>
<td></td>
<td>Value is always &quot;emergency&quot;.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl 
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/emergency?sysparm_query=active=true^ORDERBYnumber" \ 
  --request GET \ 
  --header "Accept: application/json" \ 
  --user "username":"password"
```

```json
{
  result: [
    {
      sys_id: {
        value: "1c87925347c12200e0ef563dbb9a7177",
        display_value: "1c87925347c12200e0ef563dbb9a7177"
      },
      state: {
        value: "-5",
        display_value: "New"
      },
      type: {
        value: "emergency",
        display_value: "Emergency"
      },
      ..., // all valid fields in record, example below
      short_description: {
        value: "Reboot server",
        display_value: "Reboot server"
      },
    },
    { // next record found }, ...
  ]
}
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url =
'https://instance.servicenow.com/api/sn_chg_rest/v1/change/emergency?sysparm_query=active=true^ORDERBYnumber''

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <sys_id>
      <value>1c87925347c12200e0ef563dbb9a7177</value>
      <display_value>1c87925347c12200e0ef563dbb9a7177</display_value>
    </sys_id>
    <state>
      <value>-5</value>
      <display_value>New</display_value>
    </state>
    <type>
      <value>emergency</value>
      <display_value>Emergency</display_value>
    </type>
  </result>
</response>
```
Change Management - GET /sn_chg_rest/change/emergency/{sys_id}
Retrieves the emergency change request identified by the specified sys_id.

URL format
Versioned URL: /api/sn_chg_rest/{api_version}/change/emergency/{sys_id}
Default URL: /api/sn_chg_rest/change/emergency/{sys_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the emergency change request to retrieve from the Change Request [change_request] table. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Values for all fields in the associated change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>state</td>
<td>State of the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>state: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>state.display_value</td>
<td>State to display in a UI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Internal state value.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id information for the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>sys_id: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>sys_id.display_value</td>
<td>Sys_id of the change request to display in a UI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id.value</td>
<td>Sys_id of the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Type of change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>type: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>type.display_value</td>
<td>Change type to display in a UI.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Value is always &quot;Emergency&quot;.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>type.value</td>
<td>Internal type value.</td>
</tr>
<tr>
<td></td>
<td>Value is always &quot;emergency&quot;.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
"https://instance.servicenow.com/api/snchg_rest/v1/change/emergency/b0dbda5347c12200e0ef563dbb9a718f"
--request GET
--header "Accept:application/json"
--user "username":"password"
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/emergency/b0dbda5347c12200e0ef563dbb9a718f'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<?xml version="1.0" encoding="UTF-8"?>
<response>
    <result>
        <sys_id>
            <value>b0dbda5347c12200e0ef563dbb9a718f</value>
            <display_value>b0dbda5347c12200e0ef563dbb9a718f</display_value>
        </sys_id>
        <state>
            <value>-5</value>
            <display_value>New</display_value>
        </state>
        <type>
            <value>emergency</value>
            <display_value>Emergency</display_value>
        </type>
    </result>
</response>
```
Change Management - GET /sn_chg_rest/change/model

Retrieves one or more change models based on the specified criteria.

Use this endpoint to find a change model that best fits the change request that you are trying to create. For additional information on change models, see Change Models.

URL format

Versioned URL: /api/sn_chg_rest/{api_version}/change/model

Default URL: /api/sn_chg_rest/change/model

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Name-value pairs to use to filter the result set. The name is the field on which the specified value is filtered. This parameter is mutually exclusive with sysparm_query. For example, instead of using &amp;sysparm_query=active=true, you can simplify the calling statement by using &amp;active=true. You can also use the display value when the field is a choice or reference type field, such as &amp;state=closed instead of &amp;state=7. To specify multiple key-value pairs, separate each with an ampersand, such as &amp;active=true&amp;assigned_to=john.smith.</td>
</tr>
</tbody>
</table>
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td><strong>order</strong></td>
<td>Field by which to sort the returned change models.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td><strong>sysparm_offset</strong></td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate</td>
</tr>
<tr>
<td></td>
<td>record retrieval. This functionality enables the retrieval of all records regardless of the</td>
</tr>
<tr>
<td></td>
<td>number of records, in small manageable chunks.</td>
</tr>
<tr>
<td></td>
<td>For example, the first time you call this endpoint, <code>sysparm_offset</code> is set to &quot;0&quot;.</td>
</tr>
<tr>
<td></td>
<td>Simply page through all available records, use <code>sysparm_offset=sysparm_offset+sysparm_limit</code></td>
</tr>
<tr>
<td></td>
<td>until you reach the end of all records.</td>
</tr>
<tr>
<td></td>
<td>Do not pass a negative number in the <code>sysparm_offset</code> parameter.</td>
</tr>
<tr>
<td>Data type: Number</td>
<td></td>
</tr>
<tr>
<td><strong>sysparm_query</strong></td>
<td>Encoded query used to filter the result set.</td>
</tr>
<tr>
<td>Syntax: <code>sysparm_query=&lt;col_name&gt;&lt;operator&gt;&lt;value&gt;</code></td>
<td></td>
</tr>
<tr>
<td>• <code>&lt;col_name&gt;</code>: Name of the table column to filter against.</td>
<td></td>
</tr>
<tr>
<td>• <code>&lt;operator&gt;</code>: Supports the following values:</td>
<td></td>
</tr>
<tr>
<td>○ <code>=</code>: Exactly matches <code>&lt;value&gt;</code>.</td>
<td></td>
</tr>
<tr>
<td>○ <code>!=</code>: Does not match <code>&lt;value&gt;</code>.</td>
<td></td>
</tr>
<tr>
<td>○ <code>^</code>: Logically AND multiple query statements.</td>
<td></td>
</tr>
<tr>
<td>○ <code>^OR</code>: Logically OR multiple query statements.</td>
<td></td>
</tr>
<tr>
<td>○ <code>LIKE</code>: <code>&lt;col_name&gt;</code> contains the specified string. Only works for <code>&lt;col_name&gt;</code> fields</td>
<td></td>
</tr>
<tr>
<td>○ <code>STARTSWITH</code>: <code>&lt;col_name&gt;</code> starts with the specified string. Only works for <code>&lt;col_name&gt;</code> fields</td>
<td></td>
</tr>
<tr>
<td>○ <code>ENDSWITH</code>: <code>&lt;col_name&gt;</code> ends with the specified string. Only works for <code>&lt;col_name&gt;</code> fields</td>
<td></td>
</tr>
<tr>
<td><code>&lt;value&gt;</code>: Value to match against.</td>
<td></td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td>(sysparm_query=caller_id=javascript:gs.getUserID() ^ active=true)</td>
</tr>
<tr>
<td></td>
<td>Encoded queries also supports order by functionality. To sort responses based on certain fields, use the ORDERBY and ORDERBYDESC clauses in sysparm_query.</td>
</tr>
<tr>
<td></td>
<td>Syntax:</td>
</tr>
<tr>
<td></td>
<td>• ORDERBY&lt;col_name&gt;</td>
</tr>
<tr>
<td></td>
<td>• ORDERBYDESC&lt;col_name&gt;</td>
</tr>
<tr>
<td></td>
<td>For example: sysparm_query=active=true ^ ORDERBY number ^ ORDERBYDESC category</td>
</tr>
<tr>
<td></td>
<td>This query filters all active records and orders the results in ascending order by number, and then in descending order by category.</td>
</tr>
<tr>
<td></td>
<td>If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property glide.invalid_query.returns_no_rows. Set this property to true to return no rows for an invalid query.</td>
</tr>
<tr>
<td></td>
<td>Note: The glide.invalid_query.returns_no_rows property controls the behavior of all queries across the instance, such as in lists, scripts (GlideRecord.query()), and web service APIs.</td>
</tr>
<tr>
<td>textSearch</td>
<td>String to use to search all change model record fields. This search uses ServiceNow full text search platform functionality. For more information on ServiceNow search capabilities, see Search administration.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: IR_AND_OR_QUERY</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>List containing one or more change model record objects. Each element in the change model object corresponds to a record in the Change Model [chg_model] table.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>All elements contain <code>value</code> and <code>display_value</code> name-value pairs. <code>display_value_internal</code> name-value pairs.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td><code>result.active</code></td>
<td>Flag that indicates whether the associated change model record is active and available within the instance.</td>
</tr>
<tr>
<td></td>
<td>Valid values: <code>true</code>: Change model is active. <code>false</code>: Change model is not active.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td><code>result.available_in_ui</code></td>
<td>Flag that indicates whether the associated change model record is available within the user interface.</td>
</tr>
<tr>
<td></td>
<td>Valid values: <code>true</code>: Change model is available in the user interface. <code>false</code>: Change model is not available in the user interface.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td><code>result.color</code></td>
<td>Color of the associated change model on the change request landing page.</td>
</tr>
<tr>
<td></td>
<td>Default: #cbcbcb</td>
</tr>
<tr>
<td><code>result.default_change_model</code></td>
<td>Flag that indicates whether the associated change model record is the default change model.</td>
</tr>
<tr>
<td></td>
<td>Valid values: <code>true</code>: Default <code>false</code>: Not the default</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td><code>result.description</code></td>
<td>Short description of the purpose of the change model.</td>
</tr>
<tr>
<td></td>
<td>Data type: String                                                                  Maximum length: 4,000</td>
</tr>
<tr>
<td><code>result.name</code></td>
<td>Name of the change model.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.record_preset</td>
<td>Name-value pairs of the fields that should automatically be populated with specific values, when a new change request record is created. Values are separated by caret symbols. For example: &quot;type=normal^assignment_group=a715cd759f2002002920bde8132e7018^short_description=Automated Change^EQ&quot;</td>
</tr>
<tr>
<td>result.state_field</td>
<td>Choice list field from which to collect choices, based on the context. For change models, this is always set to &quot;state&quot;.</td>
</tr>
<tr>
<td>result.sys_class_name</td>
<td>Change module table name. Always Change Model/chg_model.</td>
</tr>
<tr>
<td>result.sys_created_by</td>
<td>Name of the user that initially created the associated change module record.</td>
</tr>
<tr>
<td>result.sys_created_on</td>
<td>Date and time that the change module record was originally created.</td>
</tr>
<tr>
<td>result.sys_domain</td>
<td>If using domains in the instance, the name of the domain to which the change module record is associated.</td>
</tr>
<tr>
<td>result.sys_domain_path</td>
<td>If using domains in the instance, the domain path in which the change module record resides.</td>
</tr>
<tr>
<td>result.sys_id</td>
<td>Unique identifier of the associated change model record.</td>
</tr>
<tr>
<td>result.sys_mod_count</td>
<td>Number of times that the associated change model record has been modified.</td>
</tr>
<tr>
<td>result.sys_name</td>
<td>Name of the change model. Always the same as the <code>name</code> parameter.</td>
</tr>
<tr>
<td>result.sys_tags</td>
<td>System tags associated with the change model record.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.sys_updated_by</td>
<td>Name of the user that last updated the associated change model record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>result.sys_updated_on</td>
<td>Date and time the associated change model record was last updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.table_name</td>
<td>Table that defines the Choice list field from which to collect choices.</td>
</tr>
<tr>
<td></td>
<td>For change models, it is always set to &quot;change_request&quot;.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 80</td>
</tr>
</tbody>
</table>

**Example: cURL Request**

This example shows a request for obtaining all change model records.

```
curl "https://instance.servicenow.com/api/sn_chg_rest/v1/change/model \
--request GET \
--header "Accept:application/json" \
--user "username":"password"
```

For brevity, this response only shows a single change model object.

```
{
  "result": {
    "record_preset": {
      "display_value": "type=normal^assignment_group=a715cd759f2002002920bde8132e7018^short_description=Automated Change^EQ",
      "value": "type=normal^assignment_group=a715cd759f2002002920bde8132e7018^short_description=Automated Change^EQ"
    },
    "color": {
      "display_value": "#488df4",
      "value": "#488df4"
    },
    "default_change_model": {
      "display_value": "false",
      "value": false
    }
  }
```
This model is intended to capture a record of an automated change. There are no approvals associated with this change model.

There are no approvals associated with this change model.

This model is intended to capture a record of an automated change. There are no approvals associated with this change model.
Change Management - GET /sn_chg_rest/change/model/{sys_id}
Retrieves the change model identified by the specified sys_id.

You can then use this change model to create the desired change request. For additional information on change models, see Change Models.

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/model/{sys_id}
Default URL: /api/sn_chg_rest/change/model/{sys_id}
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change model record to retrieve from the Change Model [chg_model] table.</td>
</tr>
</tbody>
</table>

Data type: String

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Requested change model record object. Each element in this object corresponds to a field in the record in the Change Model [chg_model] table. All elements contain <code>value</code> and <code>display_value</code> name-value pairs. Some contain <code>display_value_internal</code> name-value pairs. Data type: Object</td>
</tr>
<tr>
<td>result.active</td>
<td>Flag that indicates whether the associated change model record is active and available within the instance. Valid values:</td>
</tr>
<tr>
<td></td>
<td>- true: Change model is active.</td>
</tr>
<tr>
<td></td>
<td>- false: Change model is not active.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.available_in_ui</td>
<td>Flag that indicates whether the associated change model record is available within the user interface. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Change model is available in the user interface.</td>
</tr>
<tr>
<td></td>
<td>• false: Change model is not available in the user interface.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td>result.color</td>
<td>Color of the associated change model on the change request landing page.</td>
</tr>
<tr>
<td></td>
<td>Default: #cbcbcb</td>
</tr>
<tr>
<td>result.default_change_model</td>
<td>Flag that indicates whether the associated change model record is the default change model.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Default</td>
</tr>
<tr>
<td></td>
<td>• false: Not the default</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>result.description</td>
<td>Short description of the purpose of the change model.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>result.name</td>
<td>Name of the change model.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 200</td>
</tr>
<tr>
<td>result.record_preset</td>
<td>Name-value pairs of the fields that should automatically be populated, with their associated values, when a new change request record is created. Values are separated by caret symbols. For example:</td>
</tr>
<tr>
<td></td>
<td>&quot;type=normal^assignment_group=a715cd759f2002002920bde8132e7018^short_description=Automated Change^EQ&quot;</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.state_field</td>
<td>Choice list field from which to collect choices, based on the provided table_name. For change models, this is always set to &quot;state&quot;. Data type: String</td>
</tr>
<tr>
<td>result.sys_class_name</td>
<td>Change module table name. Always Change Model/chg_model. Data type: String</td>
</tr>
<tr>
<td>result.sys_created_by</td>
<td>Name of the user that initially created the associated change module record. Data type: String</td>
</tr>
<tr>
<td>result.sys_created_on</td>
<td>Date and time that the change module record was originally created. Data type: String</td>
</tr>
<tr>
<td>result.sys_domain</td>
<td>If using domains in the instance, the name of the domain to which the change module record is associated. Data type: String</td>
</tr>
<tr>
<td>result.sys_domain_path</td>
<td>If using domains in the instance, the domain path in which the associated change module record resides. Data type: String</td>
</tr>
<tr>
<td>result.sys_id</td>
<td>Unique identifier of the associated change model record. Data type: String</td>
</tr>
<tr>
<td>result.sys_mod_count</td>
<td>Number of times that the associated change model record has been modified. Data type: Number</td>
</tr>
<tr>
<td>result.sys_name</td>
<td>Name of the change model. Always the same as the name parameter. Data type: String</td>
</tr>
<tr>
<td>result.sys_tags</td>
<td>System tags associated with the change model record. Data type: String</td>
</tr>
<tr>
<td>result.sys_updated_by</td>
<td>Name of the user that last updated the associated change model record. Maximum length: 40 Data type: String</td>
</tr>
<tr>
<td>result.sys_updated_on</td>
<td>Date and time the associated change model record was last updated. Data type: String</td>
</tr>
<tr>
<td>result.table_name</td>
<td>Table that defines the Choice list field from which to collect choices. For change models, this is always set to &quot;change_request&quot;. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>Data type: String</td>
<td>Maximum length: 80</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/model/c0efda5347c12200e0ef563db
b9a81e3" \n--request GET \n--header "Accept:application/json" \n--user "username":"password"

{
  "result": {
    "record_preset": {
      "display_value": "type=emergency^EQ",
      "value": "type=emergency^EQ"
    },
    "color": {
      "display_value": "#ea3423",
      "value": "#ea3423"
    },
    "default_change_model": {
      "display_value": "false",
      "value": false
    },
    "sys_mod_count": {
      "display_value": "2",
      "value": 2.0
    },
    "description": {
      "display_value": "ITIL Mode 1 Emergency Change",
      "value": "ITIL Mode 1 Emergency Change"
    },
    "active": {
      "display_value": "true",
      "value": true
    },
    "sys_updated_on": {
      "display_value": "2020-09-28 08:53:12",
      "value": "2020-09-28 15:53:12",
      "display_value_internal": "2020-09-28 08:53:12"
    }
  }
}
```
Change Management - GET /sn_chg_rest/change/normal

Retrieves one or more normal change requests based on the specified criteria.

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/normal

Default URL: /api/sn_chg_rest/change/normal

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>api_version</strong></td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>name-value pairs</strong></td>
<td>Name-value pairs to use to filter the result set. The name is the field on which the specified value is filtered. This parameter is mutually exclusive with <em>sysparm_query</em>. For example, instead of using &amp;sysparm_query=active=true, you can simplify the calling statement by using &amp;active=true. You can also use the display value when the field is a choice or reference type field, such as &amp;state=closed instead of &amp;state=7. To specify multiple key-value pairs, separate each with an ampersand, such as &amp;active=true&amp;assigned_to=john.smith. Data type: String</td>
</tr>
<tr>
<td><strong>order</strong></td>
<td>Field by which to sort the returned change requests. Default: number Data type: String</td>
</tr>
<tr>
<td><strong>sysparm_limit</strong></td>
<td>Maximum number of records to return. For requests that exceed this number of records, use the <em>sysparm_offset</em> parameter to paginate record retrieval. Data type: Number</td>
</tr>
</tbody>
</table>
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, <code>sysparm_offset</code> is set to &quot;0&quot;. To simply page through all available records, use <code>sysparm_offset=sysparm_offset+sysparm_limit</code>, until you reach the end of all records. Do not pass a negative number in the <code>sysparm_offset</code> parameter. Data type: Number Default: 0</td>
</tr>
</tbody>
</table>
| sysparm_query    | Encoded query used to filter the result set. Syntax: `sysparm_query=<col_name><operator><value>`.  
  • `<col_name>`: Name of the table column to filter against.  
  • `<operator>`: Supports the following values:  
    ◦ `=`: Exactly matches `<value>`.  
    ◦ `!=`: Does not match `<value>`.  
    ◦ `^`: Logically AND multiple query statements.  
    ◦ `^OR`: Logically OR multiple query statements.  
    ◦ `LIKE`: `<col_name>` contains the specified string. Only works for `<col_name>` fields whose data type is string.  
    ◦ `STARTSWITH`: `<col_name>` starts with the specified string. Only works for `<col_name>` fields whose data type is string.  
    ◦ `ENDSWITH`: `<col_name>` ends with the specified string. Only works for `<col_name>` fields whose data type is string.  
  `<value>`: Value to match against.  
  All parameters are case-sensitive. Queries can contain more than one entry as `sysparm_query=<col_name><operator><value>[<operator><col_name><operator><value>]`. For example:  
  ```javascript
  (sysparm_query=caller_id=javascript:gs.getUserID()^active=true)
  ```
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Encoded queries also supports order by functionality. To sort responses based</td>
</tr>
<tr>
<td></td>
<td>on certain fields, use the ORDERBY and ORDERBYDESC clauses in <strong>sysparm_query</strong>.</td>
</tr>
<tr>
<td></td>
<td>Syntax:</td>
</tr>
<tr>
<td></td>
<td>• ORDERBY&lt;col_name&gt;</td>
</tr>
<tr>
<td></td>
<td>• ORDERBYDESC&lt;col_name&gt;</td>
</tr>
<tr>
<td></td>
<td>For example: sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory</td>
</tr>
<tr>
<td></td>
<td>This query filters all active records and orders the results in ascending order</td>
</tr>
<tr>
<td></td>
<td>by number, and then in descending order by category.</td>
</tr>
<tr>
<td></td>
<td>If part of the query is invalid, such as by specifying an invalid field name,</td>
</tr>
<tr>
<td></td>
<td>the instance ignores the invalid part. It then returns rows using only the</td>
</tr>
<tr>
<td></td>
<td>valid portion of the query. You can control this behavior using the property</td>
</tr>
<tr>
<td></td>
<td><strong>glide.invalid_query.returns_no_rows</strong>. Set this property to true to return</td>
</tr>
<tr>
<td></td>
<td>no rows on an invalid query.</td>
</tr>
</tbody>
</table>

**Note:** The **glide.invalid_query.returns_no_rows** property controls the behavior of all queries across the instance, such as in lists, scripts, [GlideRecord.query()](https://docs.servicenow.com/c Platform/cxwikis/), and web service APIs.

<table>
<thead>
<tr>
<th>textSearch</th>
<th>String to use to search all normal change request record fields. This search uses ServiceNow full text search platform functionality and defaults to <strong>IR_AND_OR_QUERY</strong>.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](https://docs.servicenow.com/c Platform/cxwikis/).
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Values for all fields in the associated change request. Data type: Object</td>
</tr>
<tr>
<td>state</td>
<td>State of the change request prior to the delete. Data type: Object</td>
</tr>
</tbody>
</table>

```json
state: {
  display_value: "String",
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state.display_value</td>
<td>State to display in a UI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Internal state value.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id information for the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>sys_id: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>sys_id.display_value</td>
<td>Sys_id of the change request to display in a UI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id.value</td>
<td>Sys_id of the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Type of change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>type: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>type.display_value</td>
<td>Change type to display in a UI.</td>
</tr>
<tr>
<td></td>
<td>Value is always &quot;Normal&quot;.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>type.value</td>
<td>Internal type value.</td>
</tr>
<tr>
<td></td>
<td>Value is always &quot;normal&quot;.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

curl
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/normal?sysparm_query-active=true^ORDERBYnumber" \  
--request GET \  
--header "Accept:application/json" \  
--user "username":"password"

[
  result: [
  
    sys_id: {
      value: "1c87925347c12200e0ef563dbb9a7177",
      display_value: "1c87925347c12200e0ef563dbb9a7177"
    },
    state: {
      value: "-5",
      display_value: "New"
    },
    type: {
      value: "normal",
      display_value: "Normal"
    },
    ... // all valid fields in record, example below
    short_description: {
      value: "Add network switch to cabinet",
      display_value: "Add network switch to cabinet"
    },
  },
  ... // next record found , ... // and so on
  ]
]

Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url =  
  'https://instance.servicenow.com/api/sn_chg_rest/v1/change/normal?sysparm_query-active=true^ORDERBYnumber'

# Set the user credentials
```
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Accept":"application/xml"}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <sys_id>
      <value>1c87925347c12200e0ef563dbb9a7177</value>
      <display_value>1c87925347c12200e0ef563dbb9a7177</display_value>
    </sys_id>
    <state>
      <value>-5</value>
      <display_value>New</display_value>
    </state>
    <type>
      <value>normal</value>
      <display_value>Normal</display_value>
    </type>

    // all valid fields in record, single parameter example below
    <short_description>
      <value>Add network switch to cabinet</value>
      <display_value>Add network switch to cabinet</display_value>
    </short_description>

    // next record found }, ... // and so on
  </result>
</response>
Change Management - GET /sn_chg_rest/change/normal/{sys_id}
Retrieves the normal change request identified by the specified sys_id.

URL format
Versioned URL: /api/sn_chg_rest/{api_version}/change/normal/{sys_id}
Default URL: /api/sn_chg_rest/change/normal/{sys_id}

Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the normal change request to retrieve from Change Request [change_request] table. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <strong>application/json</strong> or <strong>application/xml</strong>. Default: <strong>application/json</strong></td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

**Response body parameters (JSON or XML)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Values for all fields in the associated change request. Data type: Object</td>
</tr>
<tr>
<td>state</td>
<td>State of the change request prior to the delete. Data type: Object</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>state: {</td>
<td>Display state to display in a UI. Data type: String</td>
</tr>
<tr>
<td>display_value:</td>
<td>State to display in a UI. Data type: String</td>
</tr>
<tr>
<td>value: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>state.value</td>
<td>Internal state value. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request. Data type: String</td>
</tr>
<tr>
<td>type: {</td>
<td>Type of change request. Data type: Object</td>
</tr>
<tr>
<td>display_value:</td>
<td>Change type to display in a UI. Value is always &quot;Normal&quot;. Data type: String</td>
</tr>
<tr>
<td>value: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>type.value</td>
<td>Internal type value. Value is always &quot;normal&quot;. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
   "https://instance.servicenow.com/api/sn_chg_rest/v1/change/normal/b0dbda5347c12200e0ef563dbbb9a718f"
   --request GET
   --header "Accept:application/json"
   --user "username":"password"
{
   result: [
   ]
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/normal/b0dbda5347c12200e0ef563dbbb9a718f'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()
```
# Decode the XML response into a dictionary and use the data
print(response.content)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <sys_id>b0dbda5347c12200e0ef563dbb9a718f</sys_id>
    <state>
      <value>-5</value>
      <display_value>New</display_value>
    </state>
    <type>
      <value>normal</value>
      <display_value>Normal</display_value>
    </type>
    // all valid fields in record, single parameter example below
    <short_description>
      <value>Add network switch to cabinet</value>
      <display_value>Add network switch to cabinet</display_value>
    </short_description>
  </result>
</response>
```

Change Management - GET /sn_chg_rest/change/standard
Retrieves one or more standard change requests based on the specified criteria.

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/standard

Default URL: /api/sn_chg_rest/change/standard

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>
## Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Name-value pairs to use to filter the result set. The name is the field on which the specified value is filtered. This parameter is mutually exclusive with <code>sysparm_query</code>. For example, instead of using <code>&amp;sysparm_query=active=true</code>, you can simplify the calling statement by using <code>&amp;active=true</code>. You can also use the display value when the field is a choice or reference type field, such as <code>&amp;state=closed</code> instead of <code>&amp;state=7</code>. To specify multiple key-value pairs, separate each with an ampersand, such as <code>&amp;active=true&amp;assigned_to=john.smith</code>. Data type: String</td>
</tr>
<tr>
<td>order</td>
<td>Field by which to sort the returned change requests. Default: <code>number</code></td>
</tr>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. For requests that exceed this number of records, use the <code>sysparm_offset</code> parameter to paginate record retrieval. Data type: String</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, <code>sysparm_offset</code> is set to &quot;0&quot;. Simply page through all available records, use <code>sysparm_offset=sysparm_offset+sysparm_limit</code>, until you reach the end of all records. Do not pass a negative number in the <code>sysparm_offset</code> parameter. Data type: Number</td>
</tr>
</tbody>
</table>
| sysparm_query   | Encoded query used to filter the result set. Syntax: `sysparm_query=<col_name><operator><value>`.  
- `<col_name>`: Name of the table column to filter against.  
- `<operator>`: Supports the following values:  
  - `=`: Exactly matches `<value>`.  
  - `!`: Does not match `<value>`.  
  - `^`: Logically AND multiple query statements.  
  - `^OR`: Logically OR multiple query statements. |
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIKE: &lt;col_name&gt;</td>
<td>&lt;col_name&gt; contains the specified string. Only works for &lt;col_name&gt; fields whose data type is string.</td>
</tr>
<tr>
<td>STARTSswith: &lt;col_name&gt;</td>
<td>&lt;col_name&gt; starts with the specified string. Only works for &lt;col_name&gt; fields whose data type is string.</td>
</tr>
<tr>
<td>ENDSswith: &lt;col_name&gt;</td>
<td>&lt;col_name&gt; ends with the specified string. Only works for &lt;col_name&gt; fields whose data type is string.</td>
</tr>
<tr>
<td>&lt;value&gt;: Value to match against.</td>
<td></td>
</tr>
</tbody>
</table>

All parameters are case-sensitive. Queries can contain more than one entry, such as:

```
sysparm_query=<col_name><operator><value>[<operator><col_name><operator><value>]
```

For example:

```
{sysparm_query=caller_id=javascript:gs.getUserID()^active=true}
```

Encoded queries also support order by functionality. To sort responses based on certain fields, use the **ORDERBY** and **ORDERBYDESC** clauses in `sysparm_query`.

**Syntax:**

- **ORDERBY**<col_name>
- **ORDERBYDESC**<col_name>

For example: `sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory`

This query filters all active records and orders the results in ascending order by number, and then in descending order by category.

If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property **glide.invalid_query.returns_no_rows**. Set this property to true to return no rows for an invalid query.

**Note:** The **glide.invalid_query.returns_no_rows** property controls the behavior of all queries across the instance, such as in lists, scripts (`GlideRecord.query()`), and web service APIs.

**Data type:** String

**textSearch**

String to use to search all standard change request record fields. This search uses ServiceNow full text search platform functionality and defaults to **IR_AND_OR_QUERY**.

---

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>All fields (key) with their associated values for the identified change request. Data type: Object</td>
</tr>
<tr>
<td>state</td>
<td>State of the change request prior to the delete. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>state: { display_value: &quot;String&quot;, value: &quot;String&quot; }</td>
</tr>
<tr>
<td>state.display_value</td>
<td>State to display in a UI. Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Internal state value. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id information for the change request. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>sys_id: { display_value: &quot;String&quot;, value: &quot;String&quot; }</td>
</tr>
<tr>
<td>sys_id.display_value</td>
<td>Sys_id of the change request to display in a UI. Data type: String</td>
</tr>
<tr>
<td>sys_id.value</td>
<td>Sys_id of the change request. Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Type of change request.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>type: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>type.display_value</td>
<td>Change type to display in a UI.</td>
</tr>
<tr>
<td></td>
<td>Value is always &quot;Standard&quot;.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>type.value</td>
<td>Internal type value.</td>
</tr>
<tr>
<td></td>
<td>Value is always &quot;standard&quot;.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

```bash
curl "https://instance.service-now.com/api/sn_chg_rest/v1/change/standard?sysparm_query=active=true" ORDERBYnumber" \ 
--request GET \ 
--header "Accept: application/json" \ 
--user "username": "password"

```
Example: Sample Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.service-now.com/api/sn_chg_rest/v1/change/standard?sysparm_query=active=true^ORDERBYnumber'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <sys_id>
      <value>1c87925347c12200e0ef563dbb9a7177</value>
      <display_value>1c87925347c12200e0ef563dbb9a7177</display_value>
    </sys_id>
    <state>
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Change Management - GET /sn_chg_rest/change/standard/{sys_id}
Retrieves the standard change request identified by the specified sys_id.

URL format
Versioned URL: /api/snchg_rest/{api_version}/change/standard/{sys_id}
Default URL: /api/sn_chg_rest/change/standard/{sys_id}

Supported request parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the standard change request to retrieve from the Change Request [change_request] table. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>All fields (key) with their associated values for the identified change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>state</td>
<td>State of the change request prior to the delete.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id information for the change request.</td>
</tr>
<tr>
<td>sys_id.display_value</td>
<td>Sys_id of the change request to display in a UI.</td>
</tr>
<tr>
<td>sys_id.value</td>
<td>Sys_id of the change request to display in a UI.</td>
</tr>
<tr>
<td>state.display_value</td>
<td>State to display in a UI.</td>
</tr>
</tbody>
</table>

```json
state: {
  display_value: "String",
  value: "String"
}
```

```json
sys_id: {
  display_value: "String",
  value: "String"
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id.value</td>
<td>Sys_id of the change request.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```c
curl "https://https://instance.servicenow.com/api/sn_chg_rest/v1/change/standard/b0dbda5347c12200e0ef563dbb9a718f" 
--request GET 
--header "Accept:application/json" 
--user "username":"password"
```

```json
{
    result: [
    {
        sys_id: {
            value: "1c87925347c12200e0ef563dbb9a7177",
            display_value: "1c87925347c12200e0ef563dbb9a7177"
        },
        state: {
            value: "-5",
            display_value: "New"
        },
        ..., // all valid fields in record, example below
        short_description: {
            value: "Add network switch to cabinet",
            display_value: "Add network switch to cabinet"
        }
    },
    ]
}
```

**Change Management - GET /sn_chg_rest/change/standard/template**

Retrieves one or more standard change templates based on the specified criteria.

**URL format**

*Versioned URL:* /api/sn_chg_rest/{api_version}/change/standard/template

*Default URL:* /api/sn_chg_rest/change/standard/template
**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Name-value pairs to use to filter the result set. The name is the field on which the specified value is filtered. This parameter is mutually exclusive with sysparm_query. For example, instead of using &amp;sysparm_query=active=true, you can simplify the calling statement by using &amp;active=true. You can also use the display value when the field is a choice or reference type field, such as &amp;state=closed instead of &amp;state=7. To specify multiple key-value pairs, separate each with an ampersand, such as &amp;active=true&amp;assigned_to=john.smith. Data type: String</td>
</tr>
<tr>
<td>order</td>
<td>Field by which to sort the returned standard change templates. Data type: String Default: number</td>
</tr>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. For requests that exceed this number of records, use the sysparm_offset parameter to paginate record retrieval. Data type: Number Default: 500</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, sysparm_offset is set to &quot;0&quot;. To simply page through all available records, use sysparm_offset=sysparm_offset+sysparm_limit, until you reach the end of all records. Do not pass a negative number in the sysparm_offset parameter. Data type: Number</td>
</tr>
</tbody>
</table>
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default: 0</td>
<td>Encoded query used to filter the result set.</td>
</tr>
<tr>
<td>sysparm_query</td>
<td>Syntax: <code>sysparm_query=&lt;col_name&gt;&lt;operator&gt;&lt;value&gt;</code>.</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;col_name&gt;</code>: Name of the table column to filter against.</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;operator&gt;</code>: Supports the following values:</td>
</tr>
<tr>
<td></td>
<td>◦ <code>=</code>: Exactly matches <code>&lt;value&gt;</code>.</td>
</tr>
<tr>
<td></td>
<td>◦ <code>!=</code>: Does not match <code>&lt;value&gt;</code>.</td>
</tr>
<tr>
<td></td>
<td>◦ <code>^</code>: Logically AND multiple query statements.</td>
</tr>
<tr>
<td></td>
<td>◦ <code>^OR</code>: Logically OR multiple query statements.</td>
</tr>
<tr>
<td></td>
<td>◦ <code>LIKE</code>: <code>&lt;col_name&gt;</code> contains the specified string. Only works for <code>&lt;col_name&gt;</code> fields whose data type is string.</td>
</tr>
<tr>
<td></td>
<td>◦ <code>STARTSWITH</code>: <code>&lt;col_name&gt;</code> starts with the specified string. Only works for <code>&lt;col_name&gt;</code> fields whose data type is string.</td>
</tr>
<tr>
<td></td>
<td>◦ <code>ENDSWITH</code>: <code>&lt;col_name&gt;</code> ends with the specified string. Only works for <code>&lt;col_name&gt;</code> fields whose data type is string.</td>
</tr>
<tr>
<td></td>
<td>◦ <code>&lt;value&gt;</code>: Value to match against.</td>
</tr>
</tbody>
</table>

All parameters are case-sensitive. Queries can contain more than one entry as
```
sysparm_query=<col_name><operator><value>[<operator><col_name><operator><value>]
```

For example:
```
(sysparm_query=caller_id=javascript:gs.getUserID()^active=true)
```

Encoded queries also supports order by functionality. To sort responses based on certain fields, use the `ORDERBY` and `ORDERBYDESC` clauses in `sysparm_query`.

Syntax:
- `ORDERBY<col_name>`
- `ORDERBYDESC<col_name>`

For example: `sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory`

This query filters all active records and orders the results in ascending order by number, and then in descending order by category.

If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>valid portion of the query.</td>
<td>You can control this behavior using the property <code>glide.invalid_query.returns_no_rows</code>. Set this property to true to return no rows on an invalid query.</td>
</tr>
</tbody>
</table>

Note: The `glide.invalid_query.returns_no_rows` property controls the behavior of all queries across the instance, such as in lists, scripts (GlideRecord.query()), and web service APIs.

Data type: String

<table>
<thead>
<tr>
<th>textSearch</th>
<th>String to use to search all standard change request record fields. This search uses ServiceNow full text search platform functionality and defaults to <code>IR_AND_OR_QUERY</code>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type</td>
<td>String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>All fields with their associated values for the identified standard change template. Data type: Object</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id information for the change request. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>sys_id: { display_value: &quot;String&quot;, value: &quot;String&quot; }</td>
</tr>
<tr>
<td>sys_id.display_value</td>
<td>Sys_id of the change request to display in a UI. Data type: String</td>
</tr>
<tr>
<td>sys_id.value</td>
<td>Sys_id of the change request. Data type: String</td>
</tr>
<tr>
<td>template</td>
<td>Information about the standard change template. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>template: { display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------</td>
</tr>
<tr>
<td>value: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>template.display_value</td>
<td>Template information to display in the UI.</td>
</tr>
<tr>
<td>template.value</td>
<td>Template sys_id.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/standard/template?sysparm_query=
  active=true"ORDERBYnumber"
--request GET \
--header "Accept:application/json" \n--user "username":"password"
```

```json
{
  result: [
    {
      sys_id: {
        value: "92b8544047810200e90d87e8dee490b0",
        display_value: "92b8544047810200e90d87e8dee490b0"
      },
      template : {
        value: "1c8e02ec47410200e90d87e8dee49057",
        display_value: "Add network switch to datacenter cabinet"
      },
      // all valid fields in record, example below
      short_description: {
        value: "Standard change template to add network switch to cabinet",
        display_value: "Standard change template to add network switch to cabinet"
      }
    },
    // next record found }, ... // and so on
  ]
}
```

**Example: Python request**

```python
# Install requests package for python
import requests
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/standard/template?sysparm_query=
active=true&ORDERBYnumber'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
    response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<?xml version="1.0" encoding="UTF-8"?><response><result><sys_id><value>92b8544047810200e90d87e8dee490b0</value><display_value>92b8544047810200e90d87e8dee490b0</display_value></sys_id><template><value>1c8e02ec47410200e90d87e8dee49057</value><display_value>Add network switch to datacenter cabinet</display_value></template>// all valid fields in record, single parameter example below
<short_description><value>Standard change template to add network switch to cabinet</value><display_value>Standard change template to add network switch to cabinet</display_value></short_description>
Change Management - GET /sn_chg_rest/change/standard/template/{sys_id}
Retrieves the standard change template identified by the specified sys_id.

URL format
Versioned URL: /api/sn_chg_rest/{api_version}/change/standard/template/
{sys_id}
Default URL: /api/sn_chg_rest/change/standard/template/{sys_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the standard change template to retrieve from the [std_change_record_producer] table. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>
Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>All fields with their associated values for the identified standard change template.</td>
<td>Object</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request.</td>
<td>String</td>
</tr>
<tr>
<td>template</td>
<td>Information about the standard change template.</td>
<td>Object</td>
</tr>
<tr>
<td>template.display_value</td>
<td>Template information to display in a UI.</td>
<td>String</td>
</tr>
<tr>
<td>template.value</td>
<td>Template sys_id.</td>
<td>String</td>
</tr>
</tbody>
</table>

Example: cURL request

```bash
curl "https://instance.servicenow.com/api/sn_chg_rest/v1/change/standard/template/92b8544047810200e90d87e8dee490b0" \
--request GET \
--header "Accept:application/json" \
--user "username":"password"
```

```json
{
    result: [
        {
            sys_id: "92b8544047810200e90d87e8dee490b0",
            template: {
                value: "1c8e02ec47410200e90d87e8dee49057",
                display_value: "Add network switch to datacenter cabinet"
            }
        },
        ...
    ]
}
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/standard/template/92b8544047810200e90d87e8dee490b0'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```
<?xml version="1.0" encoding="UTF-8"?>
<response>
    <result>
        <sys_id>92b8544047810200e90d87e8dee490b0</sys_id>
        <template>
            <value>1c8e02ec47410200e90d87e8dee49057</value>
            <display_value>Add network switch to datacenter cabinet</display_value>
        </template>
    </result>
</response>
```
Change Management - GET /sn_chg_rest/change/worker/{sys_id}
Retrieves the current status, information, and errors for the specified asynchronous worker.

URL format
Versioned URL: /api/sn_chg_rest/{api_version}/change/worker/{sys_id}
Default URL: /api/sn_chg_rest/change/worker/{sys_id}

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>messages</td>
<td>Message information. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;messages&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;errorMessages&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;infoMessages&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;warningMessages&quot;: [Array]</td>
</tr>
</tbody>
</table>
| messages.errorMessages | Error messages encountered while processing the request. For example:  
|                    |   Invalid CMDB_CI sys_id provided                                                                                                           |
| messages.infoMessages | Information messages encountered while processing the request. For example:  
|                    |   CMDB_CI sys_id already associated to provided.                                                                                             |
| messages.warningMessages | Warning messages encountered while processing the request. For example:  
|                    |   Invalid CMDB_CI sys_id provided.                                                                                                          |
| payload            | Unique payload provided when using a worker sys_id from the successful response body of a scheduling endpoint.  
<p>|                    | • GET /ci/{cmdb_ci_sys_id}/schedule                                                                                                          |
|                    | • GET /sn_chg_rest/change/{change_sys_id}/schedule                                                                                           |
|                    | &quot;payload&quot;: {                                                                                                                              |
|                    |   &quot;spans&quot;: [Array]                                                                                                                          |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_type: Object</td>
<td>If a timespan is available within the duration provided in the schedule endpoint, the worker API lists the available time spans.</td>
</tr>
<tr>
<td>payload.spans</td>
<td></td>
</tr>
<tr>
<td>data_type: Array</td>
<td></td>
</tr>
<tr>
<td>payload.spans.start</td>
<td></td>
</tr>
<tr>
<td>data_type: Object</td>
<td></td>
</tr>
<tr>
<td>payload.spans.start.value</td>
<td>Date and time that the change request is planned to start implementation.</td>
</tr>
<tr>
<td>data_type: String</td>
<td></td>
</tr>
<tr>
<td>payload.spans.start.display_value</td>
<td>Displays the value of the change request start time.</td>
</tr>
<tr>
<td>data_type: String</td>
<td>Time format: yyyy-mm-dd hh:mm:ss</td>
</tr>
<tr>
<td>payload.spans.end</td>
<td></td>
</tr>
<tr>
<td>data_type: Object</td>
<td></td>
</tr>
<tr>
<td>payload.spans.end.value</td>
<td>Date and time that the change request is planned for completion.</td>
</tr>
<tr>
<td>data_type: String</td>
<td>Time format: yyyy-mm-dd hh:mm:ss</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>payload.spans.end.display_value</td>
<td>Displays the value of the change request completion time. Data type: String</td>
</tr>
<tr>
<td>request</td>
<td>Original endpoint request. Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>Information on the current state of the worker. Data type: Object</td>
</tr>
<tr>
<td>state.display_value</td>
<td>Display value of the state of the worker. These values directly correlate to the state.value parameter. Possible values: Complete, Error, In-Progress, Waiting. Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Numeric value of the state of the worker. Possible values: 1, 2, 3, 4. Data type: Number</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>type</td>
<td>Type of association between the CMDB CI and the change request. Data type: String</td>
</tr>
<tr>
<td>worker</td>
<td>Information about the associated worker. Data type: Object</td>
</tr>
<tr>
<td>worker.link</td>
<td>URL to retrieve the status of the associated worker and other worker pertinent information. Data type: String</td>
</tr>
<tr>
<td>worker.sysId</td>
<td>Sys_id of the worker associated with the change request. Data type: String</td>
</tr>
<tr>
<td>&lt;other_params&gt;</td>
<td>Other parameters that are process-specific, such as ignored_cmdb_ci_sys_ids.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl "https://instance.servicenow.com/api/sn_chg_rest/v1/change/worker/0644cd02dbec330084f07ffb9619c1" 
  --request GET 
  --header "Accept:application/json" 
  --user "username":"password"

{
  "result": {
    "worker": {
      "sysId": "0644cd02dbec330084f07ffb9619c1",
      "link": "https://instance.service-now.com/api/sn_chg_rest/change/worker/0644cd02dbec330084f07ffb9619c1"
    },
    "request": "{\"task\":\"c286d61347c12200e0ef563db9a71df\"}"
  }
}
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/worker/0644cd02dbec330084f07ffdbf9619c1'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/json'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)
```
Change Management - PATCH /sn_chg_rest/change/{sys_id}

Updates the change request identified by the specified sys_id with the key-value pairs in the request body or URL.

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/{sys_id}

Default URL: /api/sn_chg_rest/change/{sys_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request to modify. Located in the Change Request [change_request] table. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Name-value pairs for the fields to update. Request body parameters override URL parameters. However, required parameters must be specified in the URL. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| data | Name-value pairs for the field(s) to update in the associated change request. For example, to update the short description file, enter a name-value pair similar to the following:  
```bash
--data "{"short_description": "my short desc" }"
```
Data type: String |

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json OR application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json OR application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Updated change request record. Each element in this object corresponds to a field in the record in the Change Request [change_request] table. All elements contain value and display_value name-value pairs. Some contain display_value_internal name-value pairs. Data type: Object</td>
</tr>
</tbody>
</table>

Example: cURL request

```bash
curl "https://instance.servicenow.com/api/sn_chg_rest/v1/change/b0dbda5347c12200e0ef563dbb9a718f" \ --request PATCH \ --header "Accept: application/json" \ --header "Content-Type: application/json" \ --data "{"short_description": "Reboot the server at 6 am\" }" \ --user "username":"password"
```


{ "reason": { "display_value": "", "value": "" }, "parent": { "display_value": "", "value": "" }, "watch_list": { "display_value": "", "value": "" }, "proposed_change": { "display_value": "", "value": "" }, "upon_reject": { "display_value": "Cancel all future Tasks", "value": "cancel" }, "sys_updated_on": { "display_value": "2015-07-06 11:59:27", "value": "2015-07-06 18:59:27", "display_value_internal": "2015-07-06 11:59:27" }, "type": { "display_value": "Standard", "value": "standard" }, "approval_history": { "display_value": "", "value": "" }, "skills": { "display_value": "", "value": "" }, "test_plan": { "display_value": "--Confirm that there are no monitoring alerts for the router", "value": "--Confirm that there are no monitoring alerts for the router" }, "number": { "display_value": "CHG0000024", "value": "" } }
"value": "CHG0000024",
},
"is_bulk": {
  "display_value": "false",
  "value": false
},
"cab_delegate": {
  "display_value": "",
  "value": ""
},
"requested_by_date": {
  "display_value": "",
  "value": "",
  "display_value_internal": ""
},
"ci_class": {
  "display_value": "cmdb_ci",
  "value": "cmdb_ci"
},
"state": {
  "display_value": "Closed",
  "value": 3.0
},
"sys_created_by": {
  "display_value": "admin",
  "value": "admin"
},
"knowledge": {
  "display_value": "false",
  "value": false
},
"order": {
  "display_value": "",
  "value": ""
},
"phase": {
  "display_value": "Requested",
  "value": "requested"
},
"cmdb_ci": {
  "display_value": "",
  "value": ""
},
"delivery_plan": {
null
"requested_by": {
    "display_value": "System Administrator",
    "value": "6816f79cc0a8016401c5a33be04be441"
},
"business_duration": {
    "display_value": "",
    "value": ""
},
"group_list": {
    "display_value": "",
    "value": ""
},
"change_plan": {
    "display_value": "",
    "value": ""
},
"approval_set": {
    "display_value": "",
    "value": "",
    "display_value_internal": ""
},
"wf_activity": {
    "display_value": "",
    "value": ""
},
"implementation_plan": {
    "display_value": "-- Place router into maintenance mode in the monitoring platform\n-- Logon to router through SSH\n-- Run the following command\n  router(config-router)#router bgp 12345\n  neighbor {neighbor ip} soft-reconfig [inbound]\n  router#clear ip bgp {neighbor ip} soft in\n-- Confirm the sessions have been cleared\n-- Place router back into operational mode in the monitoring platform",
    "value": "-- Place router into maintenance mode in the monitoring platform\n-- Logon to router through SSH\n-- Run the following command\n  router(config-router)#router bgp 12345\n  neighbor {neighbor ip} soft-reconfig [inbound]\n  router#clear ip bgp {neighbor ip} soft in\n-- Confirm the sessions have been cleared\n-- Place router back into operational mode in the monitoring platform"
},
"universal_request": {
    "display_value": "",
    "value": ""
},
"end_date": {
"display_value": "",
"value": "",
"display_value_internal": ""
},
"short_description": {
"display_value": "Reboot the server at 6 am",
"value": "Reboot the server at 6 am"
},
"correlation_display": {
"display_value": "",
"value": ""
},
"work_start": {
"display_value": "2015-07-06 11:56:04",
"value": "2015-07-06 18:56:04",
"display_value_internal": "2015-07-06 11:56:04"
},
"delivery_task": {
"display_value": "",
"value": ""
},
"outside_maintenance_schedule": {
"display_value": "false",
"value": false
},
"additional_assignee_list": {
"display_value": "",
"value": ""
},
"std_change_producer_version": {
"display_value": "Clear BGP sessions on a Cisco router - 1",
"value": "16c2273c47010200e90d87e8dee49006"
},
"sys_class_name": {
"display_value": "Change Request",
"value": "change_request"
},
"service_offering": {
"display_value": "",
"value": ""
},
"closed_by": {
"display_value": "System Administrator",
"value": "6816f79cc0a8016401c5a33be04be441"
{
    "follow_up": {
        "display_value": "",
        "value": "",
        "display_value_internal": ""
    },
    "review_status": {
        "display_value": "",
        "value": ""
    },
    "reassignment_count": {
        "display_value": "2",
        "value": 2.0
    },
    "start_date": {
        "display_value": "",
        "value": "",
        "display_value_internal": ""
    },
    "assigned_to": {
        "display_value": "",
        "value": ""
    },
    "variables": {
        "display_value": "variable_pool",
        "value": "variable_pool"
    },
    "sla_due": {
        "display_value": "UNKNOWN",
        "value": "",
        "display_value_internal": ""
    },
    "comments_and_work_notes": {
        "display_value": "",
        "value": ""
    },
    "escalation": {
        "display_value": "Normal",
        "value": 0.0
    },
    "upon_approval": {
        "display_value": "Proceed to Next Task",
        "value": "proceed"
    }
}
"correlation_id": {
    "display_value": "",
    "value": ""
},
"made_sla": {
    "display_value": "true",
    "value": true
},
"backout_plan": {
    "display_value": "Due to the limited number of commands in the implementation plan it is not possible to backout the change.\r\n\nIf required you are authorized to reboot the router if BGP fails to work",
    "value": "Due to the limited number of commands in the implementation plan it is not possible to backout the change.\r\n\nIf required you are authorized to reboot the router if BGP fails to work"
},
"conflict_status": {
    "display_value": "Not Run",
    "value": "Not Run"
},
"task_effective_number": {
    "display_value": "CHG0000024",
    "value": "CHG0000024"
},
"sys_updated_by": {
    "display_value": "admin",
    "value": "admin"
},
"opened_by": {
    "display_value": "System Administrator",
    "value": "6816f79cc0a8016401c5a33be04be441"
},
"user_input": {
    "display_value": "",
    "value": ""
},
"sys_created_on": {
    "display_value": "2015-07-06 11:55:46",
    "value": "2015-07-06 18:55:46",
    "display_value_internal": "2015-07-06 11:55:46"
},
"on_hold_task": {
    "display_value": "",
    "value": ""
"phase_state": {
  "display_value": "Open",
  "value": "open"
},
"cab_date": {
  "display_value": "",
  "value": "",
  "display_value_internal": ""
},
"work_notes": {
  "display_value": "",
  "value": ""
},
"close_code": {
  "display_value": "Successful",
  "value": "successful"
},
"assignment_group": {
  "display_value": "Network",
  "value": "287ebd7da9fe198100f92cc8d1d2154e"
},
"description": {
  "display_value": "Resend the complete BGP table to neighboring routers. Both neighbors need to support soft reset route refresh capability. Stores complete BGP table of you neighbor in router memory. Not a good idea on a peering router with full feed, due to the memory requirements."
},
"on_hold_reason": {
  "display_value": "",
  "value": ""
},
"calendar_duration": {
  "display_value": "",
  "value": ""
},
"close_notes": {
  "display_value": "Completed without issues",
  "value": "Completed without issues"
}
"display_value_internal": "",
},
"sys_mod_count": {
    "display_value": "10",
    "value": 10.0
},
"on_hold": {
    "display_value": "false",
    "value": false
},
"sys_tags": {
    "display_value": "",
    "value": ""
},
"conflict_last_run": {
    "display_value": "",
    "value": "",
    "display_value_internal": ""
},
"risk_value": {
    "display_value": "",
    "value": ""
},
"unauthorized": {
    "display_value": "false",
    "value": false
},
"risk": {
    "display_value": "Moderate",
    "value": 3.0
},
"location": {
    "display_value": "",
    "value": ""
},
"category": {
    "display_value": "Other",
    "value": "Other"
},
"risk_impact_analysis": {
    "display_value": "",
    "value": ""}
Change Management - PATCH /sn_chg_rest/change/{sys_id}/approvals
Allows the current user to approve or reject a change request approval record for the specified change request.

URL format
Versioned URL: /api/sn_chg_rest/{api_version}/change/{sys_id}/approvals
Default URL: /api/sn_chg_rest/change/{sys_id}/approvals

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request for which the approval/rejection applies. Located in the Change Request [change_request] table. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>comments</td>
<td>Required if state is rejected. Reason that the change was rejected. Data type: String</td>
</tr>
</tbody>
</table>
| state | Required. Approval state. For example: --data "{"state": "approved"}"

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• approved</td>
</tr>
<tr>
<td></td>
<td>• rejected</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad request. Indicates a bad request type such as the user not having the authority to approve or reject the change request.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Values for all fields in the associated change request. Data type: Object</td>
</tr>
<tr>
<td>state</td>
<td>Current state of the change request. Data type: Object</td>
</tr>
<tr>
<td></td>
<td><code>state: {</code>&lt;br&gt;</td>
</tr>
<tr>
<td>state.display_value</td>
<td>State to display in a UI. Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Internal state value. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request being approved/rejected. Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Type of change request. Data type: Object</td>
</tr>
<tr>
<td></td>
<td><code>type: {</code>&lt;br&gt;</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>type.display_value</td>
<td>Change type to display in a UI. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Emergency</td>
</tr>
<tr>
<td></td>
<td>• Normal</td>
</tr>
<tr>
<td></td>
<td>• Standard</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>type.value</td>
<td>Internal type value. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• emergency</td>
</tr>
<tr>
<td></td>
<td>• normal</td>
</tr>
<tr>
<td></td>
<td>• standard</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6ee11a9fe/approvals"
  --request POST 
  --header "Accept: application/json" 
  --header "Content-Type: application/json"
  --data "{"state": "approved"}"
  --user "username":"password"
```

```json
{
  result: [
    {
      sys_id: "0f4ac6c4b750230096c3e4f6ee11a9fe",
      state: {
        value: "-2",
        display_value: "Scheduled"
      },
      type: {
        value: "normal",
        display_value: "Normal"
      }
    }
  ]
}
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.service-now.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6ee11a9fe/approvals'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Content-Type': 'application/json', 'Accept': 'application/xml'}

# Make the HTTP request
response = requests.patch(url, auth=(user, pwd), headers=headers, data='{"state": "approved"}')

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <sys_id>0f4ac6c4b750230096c3e4f6ee11a9fe</sys_id>
  </result>
</response>
```
Change Management - PATCH /sn_chg_rest/change/{change_sys_id}/schedule/first_available

Updates the planned start and end times of a change request using the first available time slot found.

Role required: sn_change_writer

ℹ️ Note: Use the link provided in the response body `worker.link` property to view the schedule status.

See also:
- GET /sn_chg_rest/change/ci/{cmdb_ci_sys_id}/schedule
- GET /sn_chg_rest/change/{change_sys_id}/schedule

URL format

Versioned URL: /api/sn_chg_rest/{api_version}/change/{change_sys_id}/schedule/first_available

Default URL: /api/sn_chg_rest/change/{change_sys_id}/schedule/first_available
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>change_sys_id</td>
<td>Sys_id of the change request on which to update with the next available time slot. Located in the Change Requests [change_request] table. The selected change request must have a configuration item (cmdb_ci).</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>duration_in_seconds</td>
<td>Duration of change in seconds, that is, how much time is required to complete the change request task. Data type: Integer</td>
</tr>
<tr>
<td>planned_start_time</td>
<td>Optional. Date and time that the change request is planned to start implementation in UTC. Retrieve the available time slot start at or later than this time. If not provided, the system uses the current time as the start time. Time format: yyyy-mm-dd hh:mm:ss Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>202</td>
<td>System accepted the request.</td>
</tr>
</tbody>
</table>
| 400         | Bad Request. A bad request type or malformed request was detected. Possible issues:  
• The duration_in_seconds body parameter value is invalid or was not provided.  
• Invalid planned_start_time body parameter value provided.  
• The specified change request does not have an associated configuration item (cmdb_ci).  
• User does not have read access to the fields of the change request. |
| 403         | Forbidden. User does not have write access to the planned start and end date values of the change request. |
| 404         | Not Found. The specified record could not be found. Possible issues:  
• System cannot find the change request based on information provided.  
• User does not have read access to the record. |
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Information on any errors encountered while processing the endpoint request. Data type: Object</td>
</tr>
<tr>
<td>error.detail</td>
<td>Additional information about the error. Data type: String</td>
</tr>
<tr>
<td>error.message</td>
<td>Message that identifies the error. Data type: String</td>
</tr>
<tr>
<td>messages</td>
<td>Message information. Data type: Object</td>
</tr>
<tr>
<td>messages.errorMessages</td>
<td>Error messages encountered while processing the request. Data type: Array</td>
</tr>
<tr>
<td>messages.infoMessages</td>
<td>Information messages encountered while processing the request. Data type: Array</td>
</tr>
<tr>
<td>messages.warningMessages</td>
<td>Warning messages encountered while processing the request. Data type: Array</td>
</tr>
<tr>
<td>request</td>
<td>Original endpoint request. Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>Information on the current state of the worker.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Data type: Object</td>
<td></td>
</tr>
<tr>
<td>state:</td>
<td></td>
</tr>
<tr>
<td>display_value:</td>
<td>Display value of the state of the worker. These values directly correlate</td>
</tr>
<tr>
<td>value: Number</td>
<td>to the <code>state.value</code> parameter.</td>
</tr>
<tr>
<td>state.display_value</td>
<td></td>
</tr>
<tr>
<td>state.value</td>
<td>Numeric value of the state of the worker.</td>
</tr>
<tr>
<td>type</td>
<td>Indicates the type of request.</td>
</tr>
<tr>
<td>worker</td>
<td>Information about the associated worker.</td>
</tr>
</tbody>
</table>

Possible values:
- Complete
- Error
- In-Progress
- Waiting

Data types:
- String
- Number

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>worker.link</td>
<td>Link for retrieving change request schedule status. Use the sys_id in GET /sn_chg_rest/change/worker/{sys_id} to view results.</td>
<td>String</td>
</tr>
<tr>
<td>worker.sysId</td>
<td>Sys_id of the worker associated with the change request.</td>
<td>String</td>
</tr>
<tr>
<td>status</td>
<td>Only appears if an error is encountered. Status of the endpoint processing. Possible values: • failure</td>
<td>String</td>
</tr>
</tbody>
</table>

**Get change request schedule status**

Use the value provided in the **worker.link** to determine if the change record has been successfully scheduled for the first available time slot. The value is in the following format:

https://instance.service-now.com/api/sn_chg_rest/change/worker/<worker_sys_id>

Use the worker.link details to run the provided sys_id in GET /sn_chg_rest/change/worker/{sys_id} to view results.

The response body contains the status and provides results when processing is complete.

If an available time slot is found, the system updates the change request with the first available slot. When the state is complete, the **messages.infoMessages** reveals that the first available time slot has been set.

One of the following scheduling response values for **messages.infoMessages** are provided in the response body:
• Change has been updated – Change requested has been updated for time slot.
• No slots found for <number> days from now – No time slots available for change request duration provided within the number of days defined in the schedule window.

**Note:** The change request scheduling time slot default value is 90 days. To change this value, modify the `change.conflict.next_available.schedule_window` property. For more information, see Configure conflict analysis properties.

The following GET `/sn_chg_rest/change/worker/{sys_id}` example shows output provided using the ID provided in the worker.link detail. The results indicate that processing is complete and the change request has been updated with the first available time slot.

```json
{
  "result": {
    "worker": {
      "sysId": "355c62e0a4c87010f87712198fe9cacf",
      "link": "https://instance.service-now.com/api/sn_chg_rest/change/worker/355c62e0a4c87010f87712198fe9cacf"
    },
    "request": {
      "change_sys_id": "87ae5e900a0a2c3e263e8304e727c646",
      "duration_in_seconds": 10800,
      "timezone": "America/Los_Angeles"
    },
    "state": {
      "value": 3,
      "display_value": "Complete"
    },
    "type": "schedule",
    "messages": {
      "errorMessages": [],
      "warningMessages": [],
      "infoMessages": [
        "Change has been updated"
      ]
    }
  }
}
```
Example: cURL request

curl
  "https://instance.service-now.com/api/sn_chg_rest/change/87ae5e900a0a2c3e263e8304e727c646/schedule/first_available" \
  --request PATCH \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --data "{"duration_in_seconds":"10800"}" \
  --user "username":"password"

Results include worker.link details you can use to run the provided sys_id in the GET /sn_chg_rest/change/worker/ endpoint.

```json
{
  "result": {
    "worker": {
      "sysId": "355c62e0a4c87010f87712198fe9cacf",
      "link": "https://instance.service-now.com/api/sn_chg_rest/change/worker/355c62e0a4c87010f87712198fe9cacf"
    },
    "request": {
      "change_sys_id": "87ae5e900a0a2c3e263e8304e727c646",
      "duration_in_seconds": 10800,
      "timezone": "America/Los_Angeles"
    },
    "state": {
      "value": 1,
      "display_value": "Waiting"
    },
    "type": "schedule",
    "messages": {
      "errorMessages": [],
      "warningMessages": [],
      "infoMessages": []
    }
  }
}
```

Change Management - PATCH /sn_chg_rest/change/{change_sys_id}/task/{task_sys_id}

Updates the change request task identified by the specified sys_ids with the key-value pairs in the request body or the URL.
**URL format**

Versioned URL: `/api/sn_chg_rest/{api_version}/change/{change_sys_id}/task/{task_sys_id}`

Default URL: `/api/sn_chg_rest/change/{change_sys_id}/task/{task_sys_id}`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>change_sys_id</td>
<td>Sys_id of the change request to which the task is associated. Verifies whether the specified task is associated with the specified change request. Located in the Change Request [change_request] table. Data type: String</td>
</tr>
<tr>
<td>task_sys_id</td>
<td>Sys_id of the task to modify. Located in the Change Task [change_task] table. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key-value pairs</td>
<td>Name-value pairs for the fields to update. Request body parameters override URL parameters. However, required parameters must be specified in the URL. Data type: String</td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Name-value pairs for the field(s) to update in the associated change request. For example, to update the short description file, enter a name-value pair similar to the following: <code>--data &quot;{&quot;short_description&quot;: &quot;my short desc&quot;}&quot;</code></td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
</tbody>
</table>
Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>All fields (key) with their associated values for the identified change request task prior to the delete. Data type: Object</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id information for the change request task. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>sys_id: { display_value: &quot;String&quot;, value: &quot;String&quot; }</td>
</tr>
<tr>
<td>sys_id.display_value</td>
<td>Sys_id of the change request task to display in a UI. Data type: String</td>
</tr>
<tr>
<td>sys_id.value</td>
<td>Sys_id of the change request task. Data type: String</td>
</tr>
<tr>
<td>parent</td>
<td>Unique identifier information for the change request associated to this task. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>parent: { display_value: &quot;String&quot;, value: &quot;String&quot; }</td>
</tr>
<tr>
<td>parent.display_value</td>
<td>Task information to display in a UI.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>parent.value</td>
<td>Sys_id of the parent task.</td>
</tr>
<tr>
<td>__meta.ignoredFields</td>
<td>Key-value pairs that were passed in the call, but were not applied to the change request as they either do not exist in the base record or the fields are read-only.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl -X PATCH "https://instance.servicenow.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6ee11a9fe/task/12629ec4b750230096c3e4f6ee11a9d5?short_description=Retire both nodes" --data "{"state": "assess", "no_such_field": "this will be ignored\" }" --user "username":"password"
```

```
{
  result: [
    {
      sys_id: {
        value: "12629ec4b750230096c3e4f6ee11a9d5",
        display_value: "12629ec4b750230096c3e4f6ee11a9d5"
      },
      parent: {
        value: "0f4ac6c4b750230096c3e4f6ee11a9fe",
        display_value: "CHG0033046 "
      },
      ... // all valid fields in record, example below
      short_description: {
        value: "Retire both nodes",
        display_value: "Retire both nodes"
      }
      __meta: {
        ignoredFields: ["no_such_field"]
      }
    }
  ]
}```
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.service-now.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6ee11a9fe/task/12629ec4b750230096c3e4f6ee11a9d5?short_description=Retire both nodes'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Content-Type":"application/json","Accept":"application/xml"}

# Make the HTTP request
response = requests.patch(url, auth=(user, pwd), headers=headers, data="{"state":"assess","no_such_field":"this will be ignored" }")

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <sys_id>
      <value>12629ec4b750230096c3e4f6ee11a9d5</value>
      <display_value>12629ec4b750230096c3e4f6ee11a9d5</display_value>
    </sys_id>
    <parent>
      <value>0f4ac6c4b750230096c3e4f6ee11a9fe</value>
      <display_value>CHG0033046</display_value>
    </parent>
  </result>
// all valid fields in record, single parameter example below
Change Management - PATCH /sn_chg_rest/change/emergency/{sys_id}

Updates the emergency change request identified by the specified sys_id with the key-value pairs in the request body or the URL.

URL format

Versioned URL: /api/sn_chg_rest/{api_version}/change/emergency/{sys_id}

Default URL: /api/sn_chg_rest/change/emergency/{sys_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request to modify. Located in the [change_request] table. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Name-value pairs for the fields to update. Request body parameters override URL parameters. However, required parameters must be specified in the URL. Data type: String</td>
</tr>
</tbody>
</table>
**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| data | Name-value pairs for the field(s) to update in the associated change request. For example, to update the short description file, enter a name-value pair similar to the following: `--data "{"short_description":"my short desc" }"`.

Data type: String

---

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see **Supported REST API headers**.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

---

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see **REST API HTTP response codes**.

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

#### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Values for all fields in the associated change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>Current state of the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>state.display_value</td>
<td>State to display in a UI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Internal state value.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Type of change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
state: {
  display_value: "String",
  value: "String"
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>type.display_value</td>
<td>Change type to display in a UI. Value is always &quot;Emergency&quot;. Data type: String</td>
</tr>
<tr>
<td>type.value</td>
<td>Internal type value. Value is always &quot;emergency&quot;. Data type: String</td>
</tr>
<tr>
<td>__meta.ignoredFields</td>
<td>Name-value pairs that were passed in the call, but were not applied to the change request as they either do not exist in the base record or the fields are read-only. Data type: Array</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/emergency/b0dbda5347c12200e0ef563dbb9a718f"
  --request PATCH
  --header "Accept: application/json"
  --header "Content-Type: application/json"
  --data "{"no_such_field": "this will be ignored", }"
  --user "username":"password"
```

```json
{
  result: [  
    {
      sys_id: "b0dbda5347c12200e0ef563dbb9a718f",
    },
    state: {
      value: "-4",
      display_value: "Assess"
    },
    type: {
      value: "emergency",
      display_value: "Emergency"
    },
    ..., // all valid fields in record, example below
    short_description: {
      value: "Reboot server",
    }
  ]
}
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/emergency/b0dbda5347c12200e0ef563dbb9a718f?state=assess'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Content-Type': 'application/json', 'Accept': 'application/xml'}

# Do the HTTP request
response = requests.patch(url, auth=(user, pwd), headers=headers, data='{"state": "assess", "no_such_field": "this will be ignored" }')

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <sys_id>b0dbda5347c12200e0ef563dbb9a718f</sys_id>
    <state>
```

---

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Change Management - PATCH /sn_chg_rest/change/normal/{sys_id}

Updates the normal change request identified by the specified sys_id with the parameters in the request body or the URL.

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/normal/{sys_id}

Default URL: /api/sn_chg_rest/change/normal/{sys_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request to modify. Located in the [change_request] table. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Name-value pairs for the fields to update. Request body parameters override URL parameters. However, required parameters must be specified in the URL. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Name-value pairs for the field(s) to update in the associated change request. For example, to update the short description file, enter a name-value pair similar to the following: <code>--data &quot;{&quot;short_description&quot;: &quot;my short desc&quot; }&quot;</code>. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

**Response body parameters (JSON or XML)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Values for all fields in the associated change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>state</td>
<td>Current state of the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>state: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>state.display_value</td>
<td>State to display in a UI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Internal state value.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Type of change request.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data type: Object</strong></td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>Change type to display in a UI.</td>
</tr>
<tr>
<td></td>
<td>Value is always &quot;Normal&quot;.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type: String</strong></td>
</tr>
<tr>
<td>type.display_value</td>
<td>Change type to display in a UI.</td>
</tr>
<tr>
<td></td>
<td>Value is always &quot;Normal&quot;.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type: String</strong></td>
</tr>
<tr>
<td>type.value</td>
<td>Internal type value.</td>
</tr>
<tr>
<td></td>
<td>Value is always &quot;normal&quot;.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type: String</strong></td>
</tr>
<tr>
<td>__meta.ignoredFields</td>
<td>Name-value pairs that were passed in the call, but were not applied to the change request as they either do not exist in the base record or the fields are read-only.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/normal/b0dbda5347c12200e0ef563dbb9a718f?state=assess" \
  --request PATCH \
  --header "Accept: application/json" \
  --header "Content-Type: application/json" \
  --data "{"state": "assess" \n      "no_such_field": "this will be ignored" }" \
  --user "username":"password"
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/normal/b0dbda5347c12200e0ef563dbb9a718f7/state=assess'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Content-Type': 'application/json', 'Accept': 'application/xml'}

# Make the HTTP request
response = requests.patch(url, auth=(user, pwd), headers=headers, data='{"state": "assess", "no_such_field": "this will be ignored" }')

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```
Change Management - PATCH /sn_chg_rest/change/standard/{sys_id}
Updates the standard change request identified by the specified sys_id with the parameters in the request body or in the URL.

**URL format**
- **Versioned URL**: /api/sn_chg_rest/{api_version}/change/standard/{sys_id}
- **Default URL**: /api/sn_chg_rest/change/standard/{sys_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request to modify. Located in the [change_request] table.</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Name-value pairs for the fields to update. Request body parameters override URL parameters. However, required parameters must be specified in the URL.</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Name-value pairs for the field(s) to update in the associated change request. For example, to update the short description file, enter a name-value pair similar to the following: <code>--data &quot;{short_description&quot;: &quot;my short desc&quot;}</code>.</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Request headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>All fields (key) with their associated values for the identified change request. Data type: Object</td>
</tr>
<tr>
<td>state</td>
<td>State of the change request.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Data type: Object</td>
<td>state: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

**state.display_value**  
State to display in a UI.  
Data type: String

**state.value**  
Internal state value.  
Data type: String

**sys_id**  
Sys_id information for the change request.  
Data type: Object

**sys_id.display_value**  
Sys_id of the change request to display in a UI.  
Data type: String

**sys_id.value**  
Sys_id of the change request.  
Data type: String

---

**Example: cURL request**

```bash
curl 
"https://instance.servicenow.com/api/sn_chg_rest/v1/change/standard/1c87925347c12200e0ef563dbb9a7177?description=Reboot my email server" 
--request PATCH 
--header "Accept:application/json" 
--header "Content-Type:application/json" 
--data "{"short_description": "my short desc"}" 
--user "username":"password"
```

```json
{
  result: [
    {
      sys_id: {
        value: "1c87925347c12200e0ef563dbb9a7177",
        display_value: "1c87925347c12200e0ef563dbb9a7177"
      }
    }
  ]
}
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/standard/1c87925347c12200e0ef53dbb9a7177?description=Reboot my email server'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Content-Type': 'application/json', 'Accept': 'application/xml'}

# Make the HTTP request
response = requests.patch(url, auth=(user, pwd), headers=headers,
                           data='{"short_description": "my short desc" }')

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```
Change Management - PATCH /sn_chg_rest/change/standard/{sys_id}/risk
Calculates the risk and impact of the specified standard change based on an
evaluation of the risk conditions.

If the Change Risk Assessment plugin is installed, it also calculates the cumulative
highest risk once the risk assessment is complete.

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/{sys_id}/risk
Default URL: /api/sn_chg_rest/change/{sys_id}/risk

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>Sys_id of the standard change to evaluate. Located in the Change Request [change_request] table. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
## Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Risk assessment completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Risk assessment failed. Details of the type of failure are included in the error data.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>impact</td>
<td>Impact associated with the specified standard change.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>impact: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>impact.display_value</td>
<td>Impact information to display in a UI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>impact.value</td>
<td>Internal impact value.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>name-value pairs</td>
<td>All valid fields within the standard change record.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>risk</td>
<td>Risk calculated for the specified standard change.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>risk: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td>risk.display_value</td>
<td>Risk information to display in a UI.</td>
</tr>
<tr>
<td>risk.value</td>
<td>Internal risk value.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id information for the standard change request.</td>
</tr>
<tr>
<td>sys_id.display_value</td>
<td>Sys_id of the change request to display in a UI.</td>
</tr>
<tr>
<td>sys_id.value</td>
<td>Sys_id of the change request.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl  
"https://instance.servicenow.com/api/sn_chg_rest/v1/change/1c87925347c12200e0ef563dbb9a7177/risk" 
--request PATCH 
--header "Accept:application/json" 
--user "username":"password"

{
  sys_id: {
    value: "1c87925347c12200e0ef563dbb9a7177",
    display_value: "1c87925347c12200e0ef563dbb9a7177"
  },
  risk: {
    value: "4",
    display_value: "Low"
  },
  impact: {
    value: "3",
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/standard/1c87925347c12200e0ef563dbb9a7177/risk'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.patch(url, auth=(user, pwd), headers=headers,
data='{"short_description": "my short desc" }')

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<response>
  <result>
    <error>
      <message>Complete Risk Assessment to calculate risk</message>
      <detail></detail>
    </error>
    <status>failure</state>
  </result>
</response>
```
Change Management - POST /sn_chg_rest/change

Creates a change request record based on the default change request record. Multiple change request creations within a single call is not supported.

To create a change request for a desired model or type, other than the default of Normal, you must pass in either the model or type parameter. You can obtain the list of available change models using the Change Management - GET /sn_chg_rest/change/model or Change Management - GET /sn_chg_rest/change/model/{sys_id} endpoints.

**URL format**

**Versioned URL:** /api/sn_chg_rest/{api_version}/change

**Default URL:** /api/sn_chg_rest/change

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value</td>
<td>Name-value pairs for the fields to update. Request body parameters override URL parameters. However, required parameters must be specified in the URL.</td>
</tr>
<tr>
<td>value pairs</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** To create a change request for a desired model or type, rather than the default of "Normal", you must pass in either the model or type parameter. For example, chg_model=Normal or type=Normal.

Data type: String
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Name-value pairs for the field(s) to update in the associated change request. For example, to update the short description file, enter a name-value pair similar to the following: <code>--data &quot;{&quot;short_description&quot;:&quot;my short desc&quot; }&quot;</code> . Data type: String</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>New change request record. The elements of this object correspond to the record format in the Change Request [change_request] table. All values that are not specified in the request are set to their defaults or are empty/null.</td>
</tr>
</tbody>
</table>

Data type: Object

### Example: cURL request

```bash
curl "https://instance.servicenow.com/api/sn_chg_rest/v1/change" \
--request PATCH \
--header "Accept: application/json" \
--header "Content-Type: application/json" \
--data "{\"short_description\": \"Reboot server on scheduled interval\", }" \
--user "username":"password"
```

```json
{
  "result": {
    "reason": {
      "display_value": "",
      "value": ""
    },
    "parent": {
      "display_value": "",
      "value": ""
    }
  }
}
```
"display_value_internal": "",
"ci_class": {
    "display_value": "cmdb_ci",
    "value": "cmdb_ci"
},
"state": {
    "display_value": "New",
    "value": -5.0
},
"sys_created_by": {
    "display_value": "jane.doe",
    "value": "jane.doe"
},
"knowledge": {
    "display_value": "false",
    "value": false
},
"order": {
    "display_value": "",
    "value": ""
},
"phase": {
    "display_value": "Requested",
    "value": "requested"
},
"cmdb_ci": {
    "display_value": "",
    "value": ""
},
"delivery_plan": {
    "display_value": "",
    "value": ""
},
"impact": {
    "display_value": "3 - Low",
    "value": 3.0
},
"contract": {
    "display_value": "",
    "value": ""
},
"active": {
    "display_value": "true",
    "value": "true"}
"short_description": {
  "display_value": "Reboot server on scheduled interval",
  "value": "Reboot server on scheduled interval"
},
"correlation_display": {
  "display_value": "",
  "value": ""
},
"work_start": {
  "display_value": "",
  "value": "",
  "display_value_internal": ""
},
"delivery_task": {
  "display_value": "",
  "value": ""
},
"outside_maintenance_schedule": {
  "display_value": "false"
"value": false,
},
"additional_assignee_list": {
    "display_value": "",
    "value": ""
},
"std_change_producer_version": {
    "display_value": "",
    "value": ""
},
"sys_class_name": {
    "display_value": "Change Request",
    "value": "change_request"
},
"service_offering": {
    "display_value": "",
    "value": ""
},
"closed_by": {
    "display_value": "",
    "value": ""
},
"follow_up": {
    "display_value": "",
    "value": "",
    "display_value_internal": ""
},
"review_status": {
    "display_value": "",
    "value": ""
},
"reassignment_count": {
    "display_value": "0",
    "value": 0.0
},
"start_date": {
    "display_value": "",
    "value": "",
    "display_value_internal": ""
},
"assigned_to": {
    "display_value": "",
    "value": ""}
}
"variables": {
   "display_value": "variable_pool",
   "value": "variable_pool"
},
"sla_due": {
   "display_value": "UNKNOWN",
   "value": "",
   "display_value_internal": ""
},
"comments_and_work_notes": {
   "display_value": "",
   "value": ""
},
"escalation": {
   "display_value": "Normal",
   "value": 0.0
},
"upon_approval": {
   "display_value": "Proceed to Next Task",
   "value": "proceed"
},
"correlation_id": {
   "display_value": "",
   "value": ""
},
"made_sla": {
   "display_value": "true",
   "value": true
},
"backout_plan": {
   "display_value": "",
   "value": ""
},
"conflict_status": {
   "display_value": "Not Run",
   "value": "Not Run"
},
"task_effective_number": {
   "display_value": "CHG0030003",
   "value": "CHG0030003"
},
"sys_updated_by": {
   "display_value": "jane.doe",
   "value": "jane.doe"
"opened_by": {
    "display_value": "Jane Doe",
    "value": "ae30a9ef1b182010593876a61a4bcb7b"
},
"user_input": {
    "display_value": "",
    "value": ""
},
"sys_created_on": {
    "display_value": "2020-11-11 09:54:57",
    "value": "2020-11-11 17:54:57",
    "display_value_internal": "2020-11-11 09:54:57"
},
"on_hold_task": {
    "display_value": "",
    "value": ""
},
"sys_domain": {
    "display_value": "global",
    "value": "global"
},
"route_reason": {
    "display_value": "",
    "value": ""
},
"closed_at": {
    "display_value": "",
    "value": "",
    "display_value_internal": ""
},
"review_comments": {
    "display_value": "",
    "value": ""
},
"business_service": {
    "display_value": "",
    "value": ""
},
"time_worked": {
    "display_value": "",
    "value": ""
},
"chg_model": {"}
"comments": {
    "display_value": "",
    "value": ""
},
"approval": {
    "display_value": "Not Yet Requested",
    "value": "not requested"
},
"due_date": {
    "display_value": "",
    "value": "",
    "display_value_internal": ""
},
"sys_mod_count": {
    "display_value": "0",
    "value": 0.0
},
"on_hold": {
    "display_value": "false",
    "value": false
},
"sys_tags": {
    "display_value": "",
    "value": ""
},
"conflict_last_run": {
    "display_value": "",
    "value": "",
    "display_value_internal": ""
},
"risk_value": {
    "display_value": "",
    "value": ""
},
"unauthorized": {
    "display_value": "false",
    "value": false
},
"risk": {
    "display_value": "Moderate",
    "value": 3.0
},
"location": {
    "display_value": "",
    "value": ""
Change Management - POST /sn_chg_rest/change/{sys_id}/ci

Creates the association between a change request and Configuration Management Database (CMDB) configuration items (CI).

The creation of the association is done asynchronously, which means that a response is provided immediately and contains details for the worker. The worker does the actual work after the response.

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/{sys_id}/ci
Default URL: /api/sn_chg_rest/change/{sys_id}/ci

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>api_version</strong></td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td><strong>sys_id</strong></td>
<td>Sys_id of the change request to associate with the CMDB CI. Data type: String</td>
</tr>
</tbody>
</table>
### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>association_type</td>
<td>Required. Type of association between the CMDB CI and the change request.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• affected: CIs that are affected by the change request</td>
</tr>
<tr>
<td></td>
<td>• impacted: Services impacted by the change request</td>
</tr>
<tr>
<td></td>
<td>• offering: Impacted service offerings</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>cmdb_ci_sys_ids</td>
<td>Required. List of CMDB CI sys_ids to associate to the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array or comma-separated string</td>
</tr>
<tr>
<td>refresh_impacted_services</td>
<td>Flag used when association_type=affected to populate impacted services based on the list of affected CIs.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Populate impacted services based on the list of affected CIs</td>
</tr>
<tr>
<td></td>
<td>• false: Do not automatically populate impacted services</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>202</td>
<td>Accepted. The request was accepted for processing.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected. The error response contains pertinent messages to help troubleshoot the problem.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| messages           | Message information. Data type: Object  
|                    | "messages": {  
|                    | "errorMessages": [Array],  
|                    | "infoMessages": [Array],  
|                    | "warningMessages": [Array]  
|                    | } |
| messages.errorMessages | Error messages encountered while processing the request. For example:  
|                        | Invalid CMDB_CI sys_id provided  
|                        | Data type: Array |
| messages.infoMessages | Information messages encountered while processing the request. For example:  
|                        | CMDB_CI sys_id already associated to provided.  
|                        | Data type: Array |
| messages.warningMessages | Warning messages encountered while processing the request. For example:  
|                         | Invalid CMDB_CI sys_id provided.  
<p>|                         | Data type: Array |
| request             | Original endpoint request. Data type: String |
| state               | Information on the current state of the worker. |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>Display value of the state of the worker. These values directly correlate to the state.value element. Possible values: • Complete • Error • In-Progress • Waiting Data type: String</td>
</tr>
<tr>
<td>state.display_value</td>
<td>Numeric value of the state of the worker. Possible values: • 1 • 2 • 3 • 4 Data type: Number</td>
</tr>
<tr>
<td>type</td>
<td>Type of association between the CMDB CI and the change request. Data type: String</td>
</tr>
<tr>
<td>worker</td>
<td>Information about the associated worker. Data type: Object</td>
</tr>
<tr>
<td>worker.link</td>
<td>URL to retrieve the status of the associated worker and other worker pertinent information.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>worker.sysId</td>
<td>Sys_id of the worker associated with the change request.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/c286d61347c12200e0ef563dbb9a71df/ci'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Content-Type': 'application/json', 'Accept': 'application/json'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers,
    data='{"cmdb_ci_sys_ids":['caf043a3b7fb23000999e4f6ee11a9c0,06f043a3b7fb23000999e4f6ee11a9c1',
    "association_type":'affected'}
}

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
    response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "worker": {
            "sysId": "f490f4c6dbac330084f07ffdbf961952",
            "link": "instance.service-now.com/api/sn_chg_rest/change/worker/f490f4c6dbac330084f07ffdbf961952",
        },
        "request": {
            "cmdb_ci_sys_ids": ["caf043a3b7fb23000999e4f6ee11a9c0",
            "06f043a3b7fb23000999e4f6ee11a9c1"],
            "association_type": "affected",
            "task": "c286d61347c12200e0ef563dbb9a71df",
            "state": {
                "value": 1,
            }
    }
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Change Management - POST /sn_chg_rest/change/{sys_id}/conflict

Starts a change request conflict checking process for the specified change request (sys_id).

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/{sys_id}/conflict

Default URL: /api/sn_chg_rest/change/{sys_id}/conflict

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change management request for which to start the conflict checking process. Located in the Change Request [change_request] table. For additional information on the conflict checking process, see Conflict detection. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

#### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. Request was unable to start due to unresolvable errors. The returned message may provide additional details.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
</tbody>
</table>
Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Sys_id of the change request conflict checking process. Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

```
curl
   "https://instance.servicenow.com/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6ee11a9fe/conflict" \
   --request POST \
   --header "Accept:application/json" \
   --user "username":"password"
{
   result: "c0b5afe4b710230096c3e4f6ee11a93f"
}
```

Example: Python request

```
# Install requests package for python
import requests

# Set the request parameters
url = 'https://<instance.servicenow.com>/api/sn_chg_rest/v1/change/0f4ac6c4b750230096c3e4f6ee11a9fe/conflict'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Accept":"application/xml"}
```
# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

```xml
<response>
    <result>b0dbda5347c12200e0ef563dbb9a718f</result>
</response>
```

**Change Management - POST /sn_chg_rest/change/emergency**

Creates one emergency change request based on the default emergency change request record. Multiple emergency change request creations within a single call is not supported.

**URL format**

Versioned URL: /api/sn_chg_rest/{api_version}/change/emergency

Default URL: /api/sn_chg_rest/change/emergency

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
</tbody>
</table>

Data type: String
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| key-value pairs    | Key-value pairs of fields to modify when creating the request. The key is the field name within the template and the value is the information to populate in the field. Fields that cannot be modified and are ignored if passed in:  
  • Business rules  
  • Read-only fields as defined in ACLs  
  • Fields that do not exist  
  Data type: String |

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
</table>
| Accept   | Data format of the response body. Supported types: application/json or application/xml.  
  Default: application/json |

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Name-value pairs of the fields that were created in the emergency change request. Data type: Object</td>
</tr>
<tr>
<td>state</td>
<td>State of the change request prior to the delete. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>state: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td>state.display_value</td>
<td>State to display in a UI. Value is always &quot;New&quot;. Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Internal state value. Value is always &quot;-5&quot;. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the newly created emergency change request. Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Type of change request.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Data type: Object</td>
<td></td>
</tr>
<tr>
<td>type:</td>
<td></td>
</tr>
<tr>
<td>display_value: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>value: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>type.display_value</td>
<td>Change type to display in a UI.</td>
</tr>
<tr>
<td>Value is always &quot;Emergency&quot;</td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>type.value</td>
<td>Internal type value.</td>
</tr>
<tr>
<td>Value is always &quot;emergency&quot;</td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>__meta.ignoredFields</td>
<td>Key-value pairs that were passed in the call, but were not applied to the</td>
</tr>
<tr>
<td></td>
<td>change request as they either do not exist in the base record or the fields</td>
</tr>
<tr>
<td></td>
<td>are read-only.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash

curl
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/emergency?no_such_field=something&description=test&short_description=Reboot server" \
  --request POST \
  --header "Accept:application/json" \
  --user "username":"password"
```

```json
{
  result: [
    {
      sys_id: "b0dbda5347c12200e0ef563dbb9a718f",
      state: {
        value: "-5",
        display_value: "New"
      },
      type: {
        value: "emergency",
        display_value: "Emergency"
      }
    },
    ...
  // all valid fields in record, example below
  ]
}
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/emergency?no_such_field=something&description=test&short_description=Reboot server'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <sys_id>b0dbda5347c12200e0ef563dbb9a718f</sys_id>
  </result>
</response>
```
<state>
    <value>-5</value>
    <display_value>New</display_value>
</state>
<type>
    <value>emergency</value>
    <display_value>Emergency</display_value>
</type>
// all valid fields in record, single parameter example below
<short_description>
    <value>Reboot server</value>
    <display_value>Reboot server</display_value>
</short_description>

<meta>
    <ignoredFields>no_such_field</ignoredFields>
</meta>
</result>
</response>

Change Management - POST /sn_chg_rest/change/normal

Creates one normal change request based on the default normal change request record. Multiple normal change request creations within a single call is not supported.

URL format

Versioned URL: /api/sn_chg_rest/{api_version}/change/normal
Default URL: /api/sn_chg_rest/change/normal

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| key-value pairs | Fields to modify when creating the request. The key is the field name within the template and the value is the information to populate in the field. Fields that cannot be modified and are ignored if passed in:  
- Business rules  
- Read-only fields as defined in ACLs  
- Fields that do not exist  
Data type: String |

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Values for all fields in the associated change request. Data type: Object</td>
</tr>
<tr>
<td>state</td>
<td>State of the newly created change request. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>state: {</td>
</tr>
<tr>
<td></td>
<td>display_value: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>value: &quot;String&quot;</td>
</tr>
<tr>
<td>state.display_value</td>
<td>State to display in a UI. Value is always &quot;New&quot;. Data type: String</td>
</tr>
<tr>
<td>state.value</td>
<td>Internal state value. Value is always &quot;,-5&quot;. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the newly created normal change request. Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Type of change request. Data type: Object</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>type: display_value</td>
<td>Change type to display in a UI. Value is always &quot;Normal&quot;. Data type: String</td>
</tr>
<tr>
<td>type: value</td>
<td>Internal type value. Value is always &quot;normal&quot;. Data type: String</td>
</tr>
<tr>
<td>__meta.ignoredFields</td>
<td>Key-value pairs that were passed in the call, but were not applied to the change request as they either do not exist in the base record or the fields are read-only. Data type: Array</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/normal?no_such_field=something&
  description=test&short_description=Remove server" \
--request POST \
--header "Accept:application/json" \
--user "username":"password"

{
  result: [
    {
      sys_id: "b0dbda5347c12200e0ef563dbb9a718f",
      state: {
        value: "-5",
        display_value: "New"
      },
      type: {
        value: "normal",
        display_value: "Normal"
      },
      ... // all valid fields in record, example below
      short_description: {
        value: "Remove server",
      }
    },
    ...
  ]
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url =
'https://instance.servicenow.com/api/sn_chg_rest/v1/change/normal?no_such_field=something&
description=test&short_description=Remove server'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Set the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <sys_id>b0dbda5347c12200e0ef563dbb9a718f</sys_id>
    <state>
      <value>-5</value>
    </state>
  </result>
</response>
```
Change Management - POST /sn_chg_rest/change/{sys_id}/refresh_impacted_services

Populates the impacted services/configuration items (CIs) related list based on the primary CI.

The primary CI appears on the change request form and affected CI related list.

Note: All work items for this endpoint are performed asynchronously.

URL format

Versioned URL: /api/sn_chg_rest/{api_version}/change/{sys_id}/refresh_impacted_services

Default URL: /api/sn_chg_rest/change/{sys_id}/refresh_impacted_services

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request to use to refresh the impacted services.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml.</td>
</tr>
<tr>
<td></td>
<td>Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml.</td>
</tr>
<tr>
<td></td>
<td>Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>200</strong></td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td><strong>401</strong></td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td><strong>404</strong></td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td><strong>500</strong></td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

**Response body parameters (JSON or XML)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>messages</td>
<td>Message information. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;messages&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;errorMessages&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;infoMessages&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;warningMessages&quot;: [Array]</td>
</tr>
<tr>
<td>messages.errorMessages</td>
<td>Error messages encountered while processing the request. For example:</td>
</tr>
<tr>
<td></td>
<td>Invalid CMDB_CI sys_id provided</td>
</tr>
<tr>
<td>messages.infoMessages</td>
<td>Information messages encountered while processing the request. For example:</td>
</tr>
<tr>
<td></td>
<td>CMDB_CI sys_id already associated to provided.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>messages.warningMessages</td>
<td>Warning messages encountered while processing the request. For example:</td>
</tr>
<tr>
<td></td>
<td><em>Invalid CMDB_CI sys_id provided.</em></td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>request</td>
<td>Original endpoint request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>Information on the current state of the worker.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
</tbody>
</table>
|                           |     *state: {
|                           |     display_value: "String",
|                           |     value: "String"
<p>|                           | }*                                                                         |
| state.display_value       | Display value of the state of the worker. These values directly correlate to |
|                           | the <em>state.value</em> element.                                                   |
|                           | Possible values:                                                            |
|                           |     • Complete                                                               |
|                           |     • Error                                                                  |
|                           |     • In-Progress                                                            |
|                           |     • Waiting                                                                |
|                           | Data type: String                                                           |
| state.value               | Numeric value of the state of the worker.                                   |
|                           | Possible values:                                                            |
|                           |     • 1                                                                      |
|                           |     • 2                                                                      |
|                           |     • 3                                                                      |
|                           |     • 4                                                                      |
|                           | Data type: Number                                                           |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Type of association between the CMDB CI and the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>worker</td>
<td>Information about the associated worker.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;worker&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;link&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sysId&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>worker.link</td>
<td>URL to retrieve the status of the associated worker and other worker pertinent information.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>worker.sysId</td>
<td>Sys_id of the worker associated with the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
  "https://instance.servicenow.com/api/sn_chg_rest/v1/change/c286d61347c12200e0ef563dbb9a71df/refresh_impacted_services" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --user "username":"password"
```

```
{
  result: {
    worker: {
      sysId: "aa31c308b75033000999e4f6ee1a9c2",
      link: "http://instance.service-now.com/api/sn_chg_rest/change/worker/aa31c308b75033000999e4f6ee1a9c2"
    },
    request: "",
    state: {
      value: 1,
      display_value: "Waiting"
    }
  }
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_chg_rest/v1/change/c286d61347c12200e0ef563dbb9a71df/refresh_impacted_services'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Content-Type': 'application/json', 'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
    <result>
        <request>{"task":"c286d61347c12200e0ef563dbb9a71df"}</request>
        <messages></messages>
        <state>
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Change Management - POST /sn_chg_rest/change/standard/{standard_change_template_id}

Creates one standard change request based on an existing standard change template as specified by the passed-in template sys_id. Multiple standard change request creations within a single call is not supported.

URL format

Versioned URL: /api/sn_chg_rest/{api_version}/change/standard/{standard_change_template_id}

Default URL: /api/sn_chg_rest/change/standard/{standard_change_template_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>standard_change_template_id</td>
<td>Sys_id of the standard change template on which to base the new standard change request. Located in the Standard Change Template [std_change_record_producer] table.</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Fields within the specified standard change template to modify when creating the request. The key is the field name within the template and the value is the information to populate in the field. Fields that cannot be modified and are ignored if passed in:</td>
</tr>
<tr>
<td></td>
<td>• Description</td>
</tr>
<tr>
<td></td>
<td>• Backout plan</td>
</tr>
<tr>
<td></td>
<td>• Test plan</td>
</tr>
<tr>
<td></td>
<td>• Implementation plan</td>
</tr>
<tr>
<td></td>
<td>• Read-only fields as defined in ACLs</td>
</tr>
<tr>
<td></td>
<td>• Fields that do not exist in the specified standard change template</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body
The API returns these JSON or XML elements in the response body.

Elements returned in the response body

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Name-value pairs of the fields that were created in the standard change request. Data type: Object</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id information of the newly created standard change request. Data type: Object</td>
</tr>
<tr>
<td>sys_id.display_value</td>
<td>Sys_id of the standard change request to display in a UI. Data type: String</td>
</tr>
<tr>
<td>sys_id.value</td>
<td>Sys_id of the standard change request.</td>
</tr>
</tbody>
</table>
Elements returned in the response body (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

| __meta.ignoredFields | Key-value pairs that were passed in the call, but were not applied to the change request as they either do not exist in the base record or the fields are read-only. |
| Data type: Array |

Example: Sample cURL request

```bash
curl "https://instance.servicenow.com" \
--request POST \
--header "Accept:application/json" \
--user "username":"password"

{
    result: [{
        sys_id: {
            value: "1c87925347c12200e0ef563dbb9a7177",
            display_value: "1c87925347c12200e0ef563dbb9a7177"
        },
        ..., // all valid fields in record, example below
        short_description: {
            value: "Add network switch to cabinet",
            display_value: "Add network switch to cabinet"
        }
        __meta: {
            ignoredFields: ["no_such_field"]
        }
    }
}
```

Example: Sample Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com'
```
# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept':'application/xml'}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <sys_id>
      <value>1c87925347c12200e0ef563dbb9a7177</value>
      <display_value>1c87925347c12200e0ef563dbb9a7177</display_value>
    </sys_id>

    // all valid fields in record, single parameter example below
    <short_description>
      <value>Add network switch to cabinet</value>
      <display_value>Add network switch to cabinet</display_value>
    </short_description>

  </result>
</response>

Change Management - POST /sn_chg_rest/change/{change_sys_id}/task

Creates one change request task based on the default change request task record and associates it with the specified change request. Multiple change request task creations within a single call is not supported.
URL format
Versioned URL: /api/sn_chg_rest/{api_version}/change/{change_sys_id}/task
Default URL: /api/sn_chg_rest/change/{change_sys_id}/task

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>change_sys_id</td>
<td>Sys_id of the change request to which to associate this task. Located in the Change Request [change_request] table. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key-value pairs</td>
<td>Fields to modify when creating the request. The key is the field name within the template and the value is the information to populate in the field. Fields that cannot be modified and are ignored if passed in:</td>
</tr>
<tr>
<td></td>
<td>• Business rules</td>
</tr>
<tr>
<td></td>
<td>• Read-only fields as defined in ACLs</td>
</tr>
<tr>
<td></td>
<td>• Fields that do not exist</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes
The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Name-value pairs of the fields that were created in the change request task.</td>
<td>Object</td>
</tr>
<tr>
<td>parent</td>
<td>Information for the change request associated to the task.</td>
<td>Object</td>
</tr>
<tr>
<td>parent.display_value</td>
<td>Information to display in the UI for the change request associated to the task.</td>
<td>String</td>
</tr>
<tr>
<td>parent.value</td>
<td>Sys_id of the change request associated to the task.</td>
<td>String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id information of the newly created change request task.</td>
<td>Object</td>
</tr>
<tr>
<td>sys_id.display_value</td>
<td>Sys_id of the change request task to display in a UI.</td>
<td>String</td>
</tr>
<tr>
<td>__meta.ignoredFields</td>
<td>Key-value pairs that were passed in the call, but were not applied to the change request as they either do not exist in the base record or the fields are read-only.</td>
<td>Array</td>
</tr>
</tbody>
</table>
Example: cURL request

curl
  "https://instance.servicenow.com/api/now/change/0f4ac6c4b750230096c3e4f6ee1a9fe/task?short_description=Retire node&no_such_field=test" \
  --request POST \
  --header "Accept:application/json" \ 
  --user "username":"password"

{
  result: [

  {
    sys_id: {
      value: "12629ec4b750230096c3e4f6ee1a9d5",
      display_value: "12629ec4b750230096c3e4f6ee1a9d5"
    },
    parent: {
      value: "0f4ac6c4b750230096c3e4f6ee1a9fe ",
      display_value: "CHG0033046 "
    },
    ..., // all valid fields in record, example below
    short_description: {
      value: "Retire node",
      display_value: "Retire node"
    }
  }__meta.ignoredFields": ["no_such_field"]
  ]
}

Example: Python request

# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/
api/now/change/0f4ac6c4b750230096c3e4f6ee1a9fe/task?short_description=Retire node&no_such_field=test'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers

headers = {'Accept':'application/xml'}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print(f'Status: {response.status_code}, Headers: {response.headers}, Error Response: {response.content}
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<!xml version="1.0" encoding="UTF-8">
<response>
    <result>
        <sys_id>
            <value>12629ec4b750230096c3e4f6ee11a9d5</value>
            <display_value>12629ec4b750230096c3e4f6ee11a9d5</display_value>
        </sys_id>
        <parent>
            <value>0f4ac6c4b750230096c3e4f6ee11a9fe</value>
            <display_value>CHG0033046</display_value>
        </parent>

        // all valid fields in record, single parameter example below
        <short_description>
            <value>Retire node</value>
            <display_value>Retire node</display_value>
        </short_description>
    </result>
</response>

**CI Lifecycle Management API**

The CI Lifecycle Management API provides the ability to manipulate configuration item (CI) operational states and apply CI actions.

The API interfaces adhere to restrictions and allowances specified by not allowed CI actions, compatible CI actions, and not allowed operational transitions. If an interface attempts a restricted operation, the operation is blocked, an error is logged, and a task is created if appropriate.
CI Lifecycle Management - DELETE /now/cilifecyclemgmt/actions
Removes a configuration item (CI) action for a list of CIs.

URL format
Versioned URL: /api/now/{api_version}/cilifecyclemgmt/actions
Default URL: /api/now/cilifecyclemgmt/actions

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>Required. Configuration item action name. Data type: String</td>
</tr>
<tr>
<td>requestorId</td>
<td>Required. Sys_id of a workflow context, or an operator user ID returned from the CI Lifecycle Management - POST /now/cilifecyclemgmt/operators endpoint. Operator user IDs are located in the CI State Registered Users [statemgmt_register_users] table. Data type: String</td>
</tr>
<tr>
<td>sysIds</td>
<td>Required. Comma-separated list of CI sys_ids. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json, application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed by the endpoint. Review the response body <code>result.result</code> parameter to verify the outcome of the operation. If the endpoint encounters errors while processing the request, error codes and messages appear in the response body <code>result.errors</code> parameter.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Object encapsulating the result of the request. Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.errors</td>
<td>List of objects in which each object represents an error encountered while processing the request. Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.errors.error</td>
<td>Identifier for a state management error encountered while processing the request. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_ALREADY_SET</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_NOT_SET</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_CI_ACTION_RECORD</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_OPS_STATE_RECORD</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_SYS_ID</td>
</tr>
<tr>
<td></td>
<td>• INCOMPATIBLE_CI_ACTION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_CI_ACTION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_DELETE</td>
</tr>
<tr>
<td></td>
<td>• INVALID_INPUT_PARAMETERS</td>
</tr>
<tr>
<td></td>
<td>• INVALID_LEASETIME</td>
</tr>
<tr>
<td></td>
<td>• INVALID_OPS_STATE</td>
</tr>
<tr>
<td></td>
<td>• INVALID_OPS_STATE_TRANSITION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_REQUESTOR</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• INVALID_REQUESTOR_FOR_CI</td>
<td></td>
</tr>
<tr>
<td>• INVALID_SYS_ID</td>
<td></td>
</tr>
<tr>
<td>• MUTEX_UNAVAILABLE</td>
<td></td>
</tr>
<tr>
<td>• NOT_ALLOWED_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>• OPS_STATE_NOT_SET</td>
<td></td>
</tr>
<tr>
<td>• UNPRIORITYzed_OPS_STATE</td>
<td></td>
</tr>
<tr>
<td>• UNSUPPORTED_SYS_ID</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

result.errors.message | Message providing details on the associated error. Data type: String

result.result | Flag indicating whether the CI action was successfully removed. Possible values: Data type: String
• true: CI action was removed.
• false: CI action was not removed.

Example: cURL request
Remove a CI action from two CIs.

```
curl --request DELETE \\
'https://instance.service-now.com/api/now/cilifecyclemgmt/actions?actionName=Patching&requestorId=621b5a09309e5010f877773aa7167c0a&sysIds=00a9a80d3790200044e0bfc8bcbe5d1c,d0fdbc8437201000deabfc8bcbe5d33' \\
--header 'Accept: application/json' \\
--user "username":"password"
```

The response body confirms that the CI action was removed from the CIs.

```
{
  "result": {
    "result": true
  }
}
```
Example: Python request
Remove a CI action from two CIs.

```python
# Need to install requests package for python
import requests

import json

# Set the API endpoint URL for the request
url = "https://instance.service-now.com/api/now/cilifecyclemgmt/actions/"

# Set credentials
user = "username"
pwd = "password"

# Set query parameters
params = {
    "actionName": "Provisioning",
    "requestorId": "621b5a09309e5010f877773aa7167c0a",
    "sysIds": "00a9a80d3790200044e0bfc8bcbe5d1c,d0fdbc8437201000deeabfc8bcbe5d33"
}

# Set HTTP headers
headers = {
    "Accept": "application/json"
}

# Issue the HTTP request
response = requests.delete(url, auth=(user, pwd), params=params, headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status:", response.status_code, "Headers:", response.headers, "Error Response:", response.content)
    exit()

# Display the JSON response
print(json.dumps(response.json()))
```

The response body shows that the specified CI action is not set for the CIs.

```json
{
    "result": {
        "result": false,
        "errors": {
            
```
CI Lifecycle Management - DELETE /now/cilifecyclemgmt/operators/{req_id}
Unregisters an operator for non-workflow users.

**URL format**

Versioned URL: /api/now/{api_version}/cilifecyclemgmt/operators/{req_id}
Default URL: /api/now/cilifecyclemgmt/operators/{req_id}

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>req_id</td>
<td>Sys_id of a workflow context, or an operator user ID returned from the CI Lifecycle Management - POST /now/cilifecyclemgmt/operators endpoint.</td>
</tr>
</tbody>
</table>

Operator user IDs are located in the CI State Registered Users [statemgmt_register_users] table.
Data type: String
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed by the endpoint. Review the response body result.result parameter to verify the outcome of the operation.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>If the endpoint encounters errors while processing the request, error codes and messages appear in the response body <code>result.errors</code> parameter.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Object encapsulating the result of the request.</td>
</tr>
<tr>
<td>Data type: Object</td>
<td></td>
</tr>
</tbody>
</table>

```
"result": {
    "errors": [Array],
    "result": "String"
}
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.errors</td>
<td>List of objects in which each object represents an error encountered while processing the request.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
</tbody>
</table>

```
"errors": [
    {
        "error": "String",
        "message": "String"
    }
]
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.errors.error</td>
<td>Identifier for a state management error encountered while processing the request. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_ALREADY_SET</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_NOT_SET</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_CI_ACTION_RECORD</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DUPLICATE_OPS_STATE_RECORD</td>
<td></td>
</tr>
<tr>
<td>DUPLICATE_SYS_ID</td>
<td></td>
</tr>
<tr>
<td>INCOMPATIBLE_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>INVALID_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>INVALID_DELETE</td>
<td></td>
</tr>
<tr>
<td>INVALID_INPUT_PARAMETERS</td>
<td></td>
</tr>
<tr>
<td>INVALID_LEASETIME</td>
<td></td>
</tr>
<tr>
<td>INVALID_OPS_STATE</td>
<td></td>
</tr>
<tr>
<td>INVALID_OPS_STATE_TRANSITION</td>
<td></td>
</tr>
<tr>
<td>INVALID_REQUESTOR</td>
<td></td>
</tr>
<tr>
<td>INVALID_REQUESTOR_FOR_CI</td>
<td></td>
</tr>
<tr>
<td>INVALID_SYS_ID</td>
<td></td>
</tr>
<tr>
<td>MUTEX_UNAVAILABLE</td>
<td></td>
</tr>
<tr>
<td>NOT_ALLOWED_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>OPS_STATE_NOT_SET</td>
<td></td>
</tr>
<tr>
<td>UNPRIORITYED_OPS_STATE</td>
<td></td>
</tr>
<tr>
<td>UNSUPPORTED_SYS_ID</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

result.errors.message | Message providing details on the associated error. Data type: String

result.result | Flag indicating whether the operator was successfully unregistered. Possible values:  
\* true: Operator was successfully unregistered.  
\* false: Operator was not successfully unregistered. Data type: String

**Example: cURL request**

Unregister an operator by user ID.
curl --request DELETE \
"https://instance.service-now.com/api/now/cilifecyclemgmt/operators/3cf1b3cc30121010f877773aa7167c6e" \
--header "Accept: application/json" \
--user "username":"password"

The response body verifies that the operator was unregistered.

```
{
    "result": {
        "result": true
    }
}
```

Example: Python request

Unregister an operator by user ID.

```python
# Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request
url = "https://instance.service-now.com/api/now/cilifecyclemgmt/operators/0c7f674830521010f877773aa7167cdf"

# Set credentials
user = "username"
pwd = "password"

# Set HTTP headers
headers = {
    "Accept": "application/json"
}

# Issue the HTTP request
response = requests.delete(url, auth=(user, pwd), headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status:"+ response.status_code, "Headers:"+ response.headers, "Error Response:"+ response.content)
    exit()
```
The response body shows that the operator could not be unregistered. Verify that the `req_id` path parameter contains a valid operator user ID field value.

```json
{
    "result": {
        "result": false,
        "errors": [
            {
                "message": "Requestor [0c7f674830521010f877773aa7167cdf] is not registered",
                "error": "INVALID_REQUESTOR"
            }
        ]
    }
}
```

**CI Lifecycle Management - GET /now/cilifecyclemgmt/actions/{sys_id}**

Returns a list of active configuration item (CI) actions for the specified CI.

**URL format**

**Versioned URL:** `/api/now/{api_version}/cilifecyclemgmt/actions/{sys_id}`

**Default URL:** `/api/now/cilifecyclemgmt/actions/{sys_id}`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the CI for which to return associated CI actions. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed by the endpoint. Review the response body result.ciActions parameter to verify the outcome of the operation.</td>
</tr>
</tbody>
</table>
Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>If the endpoint encounters errors while processing the request, error codes and messages appear in the response body <code>result.errors</code> parameter.</td>
<td></td>
</tr>
</tbody>
</table>

401 Unauthorized. The user credentials are incorrect or have not been passed.

500 Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Object encapsulating the result of the request. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.ciActions</td>
<td>Comma-separated list of active CI actions for the CI, or no_active_action if none were found. Data type: String</td>
</tr>
<tr>
<td>result.errors</td>
<td>List of objects in which each object represents an error encountered while processing the request. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.errors.error</td>
<td>Identifier for a state management error encountered while processing the request. Possible values:</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CI_ACTION_ALREADY_SET</td>
<td></td>
</tr>
<tr>
<td>CI_ACTION_NOT_SET</td>
<td></td>
</tr>
<tr>
<td>DUPLICATE_CI_ACTION_RECORD</td>
<td></td>
</tr>
<tr>
<td>DUPLICATE_OPS_STATE_RECORD</td>
<td></td>
</tr>
<tr>
<td>DUPLICATE_SYS_ID</td>
<td></td>
</tr>
<tr>
<td>INCOMPATIBLE_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>INVALID_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>INVALID_DELETE</td>
<td></td>
</tr>
<tr>
<td>INVALID_INPUT_PARAMETERS</td>
<td></td>
</tr>
<tr>
<td>INVALID_LEASETIME</td>
<td></td>
</tr>
<tr>
<td>INVALID_OPS_STATE</td>
<td></td>
</tr>
<tr>
<td>INVALID_OPS_STATE_TRANSITION</td>
<td></td>
</tr>
<tr>
<td>INVALID_REQUESTOR</td>
<td></td>
</tr>
<tr>
<td>INVALID_REQUESTOR_FOR_CI</td>
<td></td>
</tr>
<tr>
<td>INVALID_SYS_ID</td>
<td></td>
</tr>
<tr>
<td>MUTEX_UNAVAILABLE</td>
<td></td>
</tr>
<tr>
<td>NOT_ALLOWED_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>OPS_STATE_NOT_SET</td>
<td></td>
</tr>
<tr>
<td>UNPRIORITIZED_OPS_STATE</td>
<td></td>
</tr>
<tr>
<td>UNSUPPORTED_SYS_ID</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

result.errors.message | Message providing details on the associated error. |
Data type: String

result.result        | Flag indicating whether the list of active CI actions was returned for all specified CIs. Possible values: |
|                    | • true: List of active CI actions was returned. |
|                    | • false: List of active CI actions was not returned. |
Data type: String
Example: cURL request

curl --request GET \
'https://instance.service-now.com/api/now/cilifecyclemgmt/actions/00a9a80d3790200044e0bfc8bcbe5d1c' \
--header 'Accept: application/json' \
--user "username":"password"

The response body shows that no CI actions are active for the specified CI.

{
  "result": {
    "ciActions": [
      "no_active_action"
    ],
    "result": true
  }
}

Example: Python request

#Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request
url = 
  "https://instance.service-now.com/api/now/cilifecyclemgmt/actions/d0fdbc8437201000deeabfc8bcbe5d33"

# Set credentials
user = "username"
pwd = "password"

# Set HTTP headers
headers = { 
  "Accept": "application/json"
}

# Issue the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
The response body shows that a CI action with name "Patching" is active for the specified CI.

```
{
  "result": {
    "ciActions": ["Patching"],
    "result": true
  }
}
```

**CI Lifecycle Management - GET /now/cilifecyclemgmt/compatActions**

Determines whether two specified configuration item (CI) actions are compatible.

**URL format**

Versioned URL: `/api/now/{api_version}/cilifecyclemgmt/compatActions`  
Default URL: `/api/now/cilifecyclemgmt/compatActions`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| api_version| Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.  
Data type: String |

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>Required. Name of CI action to compare.</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>otherActionName</td>
<td>Required. Name of CI action to compare.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed by the endpoint.</td>
</tr>
</tbody>
</table>
Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Review the response body <code>result.result</code> parameter to verify the outcome of the operation. If the endpoint encounters errors while processing the request, error codes and messages appear in the response body <code>result.errors</code> parameter.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| result | Flag indicating whether the two specified CI actions are compatible. Possible values:  
• `true`: CI actions are compatible.  
• `false`: CI actions are not compatible.  
Data type: String |

Example: cURL request

Check compatibility of Patching and Provisioning CI actions.

```
curl --request GET  
'https://instance.service-now.com/api/now/cilifecyclemgmt/compatActions?actionName=Patching 
&otherActionName=Provisioning' 
--header 'Accept: application/json' 
--user "username":"password"
```

The response body confirms that the two CI actions are compatible.

```
{
    "result": true
}
```
Example: Python request

Check compatibility of Patching and Disposing CI actions.

```python
# Need to install requests package for python
import requests

# Set the API endpoint URL for the request
url =
  "https://instance.service-now.com/api/now/cilifecyclemgmt/compatActions?actionName=Patchin
g&otherActionName=Disposing"

# Set credentials
user = "username"
pwd = "password"

# Set HTTP headers
headers = {
  "Accept": "application/json"
}

# Issue the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status:", response.status_code, "Headers:", response.headers, "Error Response:", response.content)
    exit()

# Display the JSON response
print(json.dumps(response.json()))
```

The response body shows that the two CI actions are incompatible.

```json
{
  "result": false
}
```

CI Lifecycle Management - GET /now/cilifecyclemgmt/leases/{sys_id}/expired

Determines whether the lease has expired for the requestor of a configuration item (CI) action applied to the specified CI.
## URL format

- **Versioned URL:** `/api/now/{api_version}/cilifecyclemgmt/leases/{sys_id}/expired`
- **Default URL:** `/api/now/cilifecyclemgmt/leases/{sys_id}/expired`

## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, <code>v1</code> or <code>v2</code>. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the CI that is the target of the applied CI action.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>Required. Name of the CI action applied to the specified CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>requestorId</td>
<td>Required. Sys_id of a workflow context, or an operator user ID returned from the CI Lifecycle Management - POST /now/cilifecyclemgmt/operators endpoint. Operator user IDs are located in the CI State Registered Users [statemgmt_register_users] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed by the endpoint. Review the response body result.result parameter to verify the outcome of the operation. If the endpoint encounters errors while processing the request, error codes and messages appear in the response body result.errors parameter.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Object encapsulating the result of the request. Data type: Object</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>result</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.errors</td>
<td>List of objects in which each object represents an error encountered while processing the request. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.errors.error</td>
<td>Identifier for a state management error encountered while processing the request. Possible values: CI_ACTION_ALREADY_SET, CI_ACTION_NOT_SET, DUPLICATE_CI_ACTION_RECORD, DUPLICATE_OPS_STATE_RECORD, DUPLICATE_SYS_ID, INCOMPATIBLE_CI_ACTION, INVALID_CI_ACTION, INVALID_DELETE, INVALID_INPUT_PARAMETERS, INVALID_LEASETIME, INVALID_OPS_STATE, INVALID_OPS_STATE_TRANSITION, INVALID_REQUESTOR, INVALID_REQUESTOR_FOR_CI, INVALID_SYS_ID, MUTEX_UNAVAILABLE, NOT_ALLOWED_CI_ACTION</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.errors.message</td>
<td>Message providing details on the associated error.</td>
</tr>
<tr>
<td>result.result</td>
<td>Flag indicating whether the lease for the requestor of the applied CI action has expired. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Requestor's lease has expired.</td>
</tr>
<tr>
<td></td>
<td>• false: Requestor's lease has not expired.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl --request GET \\ 'https://instance.service-now.com/api/now/cilifecyclemgmt/leases/00a9a80d3790200044e0bfc8bbe5dc/expired?actionName=Patching&requestorId=621b5a09309e5010f877773aa7167c0a' \\
--header 'Accept: application/json' \\
--user "username":"password"
```

The response body shows that the lease for the requestor of the applied CI action has not expired.

```
{
    "result": {
        "result": false
    }
}
```

**Example: Python request**

```
# Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request
```
url =
  "https://instance.service-now.com/api/now/cilifecyclemgmt/leases/affd3c8437201000deabfc8bcbe5dc3/expired?actionName=Patching&requestorId=4cab9b5301a9010f877773aa7167ca9"

# Set credentials
user = "username"
pwd = "password"

# Set HTTP headers
headers = {
    "Accept": "application/json"
}

# Issue the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status:", response.status_code, "Headers:", response.headers, "Error Response:", response.content)
    exit()

# Display the JSON response
print(json.dumps(response.json()))

The response body shows that the lease for the requestor of the applied CI action has expired.

{
    "result": {
        "result": true
    }
}

**CI Lifecycle Management - GET /now/cilifecyclemgmt/notAllowedAction**

Determines whether a specified configuration item (CI) action is not allowed for a CI of a specified class when in a specified operational state.

**URL format**

Versioned URL: /api/now/{api_version}/cilifecyclemgmt/notAllowedAction

Default URL: /api/now/cilifecyclemgmt/notAllowedAction
**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>Required. Name of the CI action to check.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>ciClass</td>
<td>Required. CI class.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>opsLabel</td>
<td>Required. Operational state to check.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed by the endpoint. Review the response body result.result parameter to verify the outcome of the operation. If the endpoint encounters errors while processing the request, error codes and messages appear in the response body result.errors parameter.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

**Response body parameters (JSON or XML)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Flag indicating whether the CI action is restricted (not allowed) for CIs of the specified type when in the specified operational state. Possible values: • true: CI action is restricted (not allowed). • false: CI action is unrestricted (allowed). Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

Check whether the Patching CI action is restricted (not allowed) for computers in the DR Standby operational state.

curl --request GET \
"https://instance.service-now.com/api/now/cilifecyclemgmt/notAllowedAction?actionName=Patch\ning&ciClass=cmdb_ci_computer&opsLabel=DR+Standby" \
--header "Accept: application/json" \
--user "username":"password"

The response body confirms that this action is restricted (not allowed) for the specified CI class when in the specified operational state.

```
{
  "result": true
}
```

Example: Python request

Check whether the Provisioning action is restricted (not allowed) for network firewall hardware CIs when in the Retired operational state.

```python
# Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request

# Set credentials
user = "username"
pwd = "password"

# Set HTTP headers
headers = {
    "Accept": "application/json"
}

# Issue the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
```
print("Status": response.status_code, "Headers": response.headers, "Error Response":, response.content)
exit()

# Display the JSON response
print(json.dumps(response.json()))

The response body confirms that this action is unrestricted (allowed) for the specified CI class when in the specified operational state.

{
  "result": false
}

CI Lifecycle Management - GET /now/cilifecyclemgmt/notAllowedOpsTransition

Determines whether a configuration item (CI) from a specified class can transition from a specified operational state to a second specified operational state.

**URL format**

Versioned URL: /api/now/{api_version}/cilifecyclemgmt/notAllowedOpsTransition

Default URL: /api/now/cilifecyclemgmt/notAllowedOpsTransition

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ciClass</td>
<td>Required. CI class to check transition restrictions for. Data type: String</td>
</tr>
<tr>
<td>opsLabel</td>
<td>Required. Label of current CI operational state.</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>transitionOpsLabel</td>
<td>Required. Label of operational state that is the target of the transition. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed by the endpoint. Review the response body <code>result.result</code> parameter to verify the outcome of the operation. If the endpoint encounters errors while processing the request, error codes and messages appear in the response body <code>result.errors</code> parameter.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| result | Flag indicating whether the specified operational state transition is restricted (not allowed) for the specified CI class. Possible values:  
• true: Operational state transition is restricted (not allowed).  
• false: Operational state transition is unrestricted (allowed).  
Data type: String |

### Example: cURL request

Check whether transition from Retired operational state to DR Standby operational state is restricted (not allowed) for Linux server CIs.

```
curl --request GET "https://instance.service-now.com/api/now/cilifecyclemgmt/notAllowedOpsTransition?ciClass=cmdb_ci_computer&opsLabel=Retired&transitionOpsLabel=DR+Standby" 
--header "Accept: application/json" 
--user "username":"password"
```

The response body shows that the specified operational state transition is unrestricted (allowed) for this CI class.
Example: Python request

Check whether transition from Repair in Progress operational state to DR Standby operational state is restricted (not allowed) for Linux server CIs.

```python
# Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request
url =
    "https://instance.service-now.com/api/now/cilifecyclemgmt/notAllowedOpsTransition?ciClass=cmdb_ci_linux_server&opsLabel=Repair+in+Progress&transitionOpsLabel=DR+Standby"

# Set credentials
user = "username"
pwd = "password"

# Set HTTP headers
headers = {
    "Accept": "application/json"
}

# Issue the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status:", response.status_code, "Headers:", response.headers, "Error Response:", response.content)
    exit()

# Display the JSON response
print(json.dumps(response.json()))
```

The response body shows that the specified operational state transition is restricted (not allowed) for this CI class.

```json
{
    "result": true
}
```
**CI Lifecycle Management - GET /now/cilifecyclemgmt/requestors/{req_id}/valid**

Determines whether the specified active workflow user or registered user is a valid requestor.

**URL format**

Versioned URL: `/api/now/{api_version}/cilifecyclemgmt/requestors/{req_id}/valid`

Default URL: `/api/now/cilifecyclemgmt/requestors/{req_id}/valid`

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>api_version</strong></td>
<td>Optional. Version of the endpoint to access. For example, <code>v1</code> or <code>v2</code>. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td><strong>req_id</strong></td>
<td>Sys_id of a workflow context, or an operator user ID returned from the CI Lifecycle Management - POST /now/cilifecyclemgmt/operators endpoint. Operator user IDs are located in the CI State Registered Users [statemgmt_register_users] table. Data type: String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Request body parameters (XML or JSON)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes
The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed by the endpoint. Review the response body result.result parameter to verify the outcome of the operation. If the endpoint encounters errors while processing the request, error codes and messages appear in the response body result.errors parameter.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Object encapsulating the result of the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.errors</td>
<td>List of objects in which each object represents an error encountered while</td>
</tr>
<tr>
<td></td>
<td>processing the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.errors.error</td>
<td>Identifier for a state management error encountered while processing the request.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_ALREADY_SET</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_NOT_SET</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_CI_ACTION_RECORD</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_OPS_STATE_RECORD</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_SYS_ID</td>
</tr>
<tr>
<td></td>
<td>• INCOMPATIBLE_CI_ACTION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_CI_ACTION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_DELETE</td>
</tr>
<tr>
<td></td>
<td>• INVALID_INPUT_PARAMETERS</td>
</tr>
<tr>
<td></td>
<td>• INVALID_LEASETIME</td>
</tr>
<tr>
<td></td>
<td>• INVALID_OPS_STATE</td>
</tr>
<tr>
<td></td>
<td>• INVALID_OPS_STATE_TRANSITION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_REQUESTOR</td>
</tr>
</tbody>
</table>
## Name | Description
--- | ---
• INVALID_REQUESTOR_FOR_CI | 
• INVALID_SYS_ID | 
• MUTEX_UNAVAILABLE | 
• NOT_ALLOWED_CI_ACTION | 
• OPS_STATE_NOT_SET | 
• UNPRIORITIZED_OPS_STATE | 
• UNSUPPORTED_SYS_ID | 

Data type: String

result.errors.message | Message providing details on the associated error.
Data type: String

result.result | Flag indicating whether the specified active workflow user or registered user is a valid requestor.
Possible values:
• true: Valid requestor.
• false: Invalid requestor.
Data type: String

### Example: cURL request
```bash
curl --request GET \\
'https://instance.service-now.com/api/now/cilifecyclemgmt/requestors/4cab9b95301a9010f87773aa7167ca9/valid' \\
--header 'Accept: application/json' \\
--user "username":"password"

```

```
{
    "result": {
        "result": true
    }
}
```

### Example: Python request
```python
# Need to install requests package for python
import requests
import json
```
# Set the API endpoint URL for the request
url = "https://instance.service-now.com/api/now/cilifecyclemgmt/requestors/621b5a09309e5010f877773aa7167c0a/valid"

# Set credentials
user = "username"
pwd = "password"

# Set HTTP headers
headers = {
    "Accept": "application/json"
}

# Issue the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status: ", response.status_code, "Headers:", response.headers, "Error Response:", response.content)
    exit()

# Display the JSON response
print(json.dumps(response.json()))

{
    "result": {
        "result": false
    }
}

**CI Lifecycle Management - GET /now/cilifecyclemgmt/statuses/{sys_id}**

Returns the current operational state for the specified configuration item (CI).

**URL format**

Versioned URL: /api/now/{api_version}/cilifecyclemgmt/statuses/{sys_id}
Default URL: /api/now/cilifecyclemgmt/statuses/{sys_id}
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed by the endpoint. Review the response body <code>result.result</code> parameter to verify the outcome of the operation. If the endpoint encounters errors while processing the request, error codes and messages appear in the response body <code>result.errors</code> parameter.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Object encapsulating the result of the request. Data type: Object</td>
</tr>
<tr>
<td></td>
<td><code>&quot;result&quot;: {</code></td>
</tr>
<tr>
<td></td>
<td><code>&quot;errors&quot;: [Array],</code></td>
</tr>
<tr>
<td></td>
<td><code>&quot;result&quot;: &quot;String&quot;</code></td>
</tr>
<tr>
<td>result.errors</td>
<td>List of objects in which each object represents an error encountered while processing the request. Data type: Array</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>&quot;errors&quot;: [</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>result.errors.error</td>
<td>Identifier for a state management error encountered while processing the request. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_ALREADY_SET</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_NOT_SET</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_CI_ACTION_RECORD</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_OPS_STATE_RECORD</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_SYS_ID</td>
</tr>
<tr>
<td></td>
<td>• INCOMPATIBLE_CI_ACTION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_CI_ACTION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_DELETE</td>
</tr>
<tr>
<td></td>
<td>• INVALID_INPUT_PARAMETERS</td>
</tr>
<tr>
<td></td>
<td>• INVALID_LEASETIME</td>
</tr>
<tr>
<td></td>
<td>• INVALID_OPS_STATE</td>
</tr>
<tr>
<td></td>
<td>• INVALID_OPS_STATE_TRANSITION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_REQUESTOR</td>
</tr>
<tr>
<td></td>
<td>• INVALID_REQUESTOR_FOR_CI</td>
</tr>
<tr>
<td></td>
<td>• INVALID_SYS_ID</td>
</tr>
<tr>
<td></td>
<td>• MUTEX_UNAVAILABLE</td>
</tr>
<tr>
<td></td>
<td>• NOT_ALLOWED_CI_ACTION</td>
</tr>
<tr>
<td></td>
<td>• OPS_STATE_NOT_SET</td>
</tr>
<tr>
<td></td>
<td>• UNPRIORITIZED_OPS_STATE</td>
</tr>
<tr>
<td></td>
<td>• UNSUPPORTED_SYS_ID</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.errors.message</td>
<td>Message providing details on the associated error.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.operationalStates</td>
<td>Operational state for the specified CI or unknown if no state is specified.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl --request GET
'https://instance.service-now.com/api/now/cilifecyclemgmt/statuses/aac0b1213784200044e0bfc8bcbe5de3' \
--header 'Accept: application/json'
--user "username":"password"
```

```
{
    "result": {
        "operationalState": "Repair in Progress",
        "result": true
    }
}
```

**Example: Python request**

```
# Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request
url =
    "https://instance.service-now.com/api/now/cilifecyclemgmt/statuses/aac0b1213784200044e0bfc8bcbe5de3"

# Set credentials
user = "username"
pwd = "password"

# Set HTTP headers
headers = {
    "Accept": "application/json"
}

# Issue the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)
```
# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status:", response.status_code, "Headers:", response.headers, "Error Response:",
          response.content)
    exit()

# Display the JSON response
print(json.dumps(response.json()))

{
    "result": {
        "operationalState": "Ready",
        "result": true
    }
}

**CI Lifecycle Management - PATCH /now/cilifecyclemgmt/leases/{sys_id}**

Extends the specified configuration item (CI) action's lease expiration time for the specified user.

If the previous lease has expired, the new lease time starts immediately.

**URL format**

*Versioned URL:* /api/now/{api_version}/cilifecyclemgmt/leases/{sys_id}

*Default URL:* /api/now/cilifecyclemgmt/leases/{sys_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the CI for which to extend the lease. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>Required. Name of the associated CI action.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>leaseTime</td>
<td>Required. Time duration for which the lease is valid for the CI action specified in <code>actionName</code>.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Format: HH:MM:SS</td>
</tr>
<tr>
<td>requestorId</td>
<td>Required. Sys_id of a workflow context, or an operator user ID returned from the CI Lifecycle Management - POST /now/cilifecyclemgmt/operators endpoint.</td>
</tr>
<tr>
<td></td>
<td>Operator user IDs are located in the CI State Registered Users [statemgmt_register_users] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>.</td>
</tr>
<tr>
<td></td>
<td>Default: <code>application/json</code></td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: <code>application/json</code> or <code>application/xml</code>.</td>
</tr>
<tr>
<td></td>
<td>Default: <code>application/json</code></td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed by the endpoint. Review the response body <code>result.result</code> parameter to verify the outcome of the operation. If the endpoint encounters errors while processing the request, error codes and messages appear in the response body <code>result.errors</code> parameter.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

**Response body parameters (JSON or XML)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Object encapsulating the result of the request. Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.errors</td>
<td>List of objects in which each object represents an error encountered while processing the request. Data type: Array</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.errors.error</td>
<td>Identifier for a state management error encountered while processing the request. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_ALREADY_SET</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_NOT_SET</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_CI_ACTION_RECORD</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_OPS_STATE_RECORD</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_SYS_ID</td>
</tr>
<tr>
<td></td>
<td>• INCOMPATIBLE_CI_ACTION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_CI_ACTION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_DELETE</td>
</tr>
<tr>
<td></td>
<td>• INVALID_INPUT_PARAMETERS</td>
</tr>
<tr>
<td></td>
<td>• INVALID_LEASETIME</td>
</tr>
<tr>
<td></td>
<td>• INVALID_OPS_STATE</td>
</tr>
<tr>
<td></td>
<td>• INVALID_OPS_STATE_TRANSITION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_REQUESTOR</td>
</tr>
<tr>
<td></td>
<td>• INVALID_REQUESTOR_FOR_CI</td>
</tr>
<tr>
<td></td>
<td>• INVALID_SYS_ID</td>
</tr>
<tr>
<td></td>
<td>• MUTEX_UNAVAILABLE</td>
</tr>
<tr>
<td></td>
<td>• NOT_ALLOWED_CI_ACTION</td>
</tr>
<tr>
<td></td>
<td>• OPS_STATE_NOT_SET</td>
</tr>
<tr>
<td></td>
<td>• UNPRIORITYED_OPS_STATE</td>
</tr>
<tr>
<td></td>
<td>• UNSUPPORTED_SYS_ID</td>
</tr>
<tr>
<td>result.errors.message</td>
<td>Message providing details on the associated <strong>error</strong>.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.result</td>
<td>Flag indicating whether the lease time was extended. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Lease time was extended.</td>
</tr>
<tr>
<td></td>
<td>• false: Lease time was not extended.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Example: cURL request

Extend the duration of a Patching CI action requestor lease by 24 hours.

```bash
curl --request PATCH
  'https://instance.service-now.com/api/now/cilifecyclemgmt/leases/affd3c8437201000deeabfc8bcbe5dc3?actionName=Patching&leaseTime=24:00:00&requestorId=e7c3402d305a9010f877773aa7167c2a'
    --header 'Accept: application/json'
    --header 'Content-Type: application/json'
    --user "username":"password"

{
  "result": {
    "result": true
  }
}
```

### Example: Python request

Extend the duration of a Provisioning CI action requestor lease by 90 minutes.

```python
# Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request
url =
  "https://instance.service-now.com/api/now/cilifecyclemgmt/leases/affd3c8437201000deeabfc8bcbe5dc3?actionName=Provisioning&leaseTime=01:30:00&requestorId=e7c3402d305a9010f877773aa7167c2a"

# Set credentials
user = "username"
pwd = "password"
```
# Set HTTP headers
headers = {
    "Accept": "application/json"
}

# Issue the HTTP request
response = requests.patch(url, auth=(user, pwd), headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status:", response.status_code, "Headers:", response.headers, "Error Response:", response.content)
    exit()

# Display the JSON response
print(json.dumps(response.json()))

{
    "result": {
        "result": true
    }
}
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actionName</td>
<td>Required. Name of the CI action to add to the list.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>leaseTime</td>
<td>Time duration for which the lease is valid for specified CI action.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Format: HH:MM:SS</td>
</tr>
<tr>
<td>oldActionNames</td>
<td>Comma-separated list of old CI actions that all CIs should be in.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>requestorId</td>
<td>Required. Sys_id of a workflow context, or an operator user ID returned from</td>
</tr>
<tr>
<td></td>
<td>the CI Lifecycle Management - POST /now/cilifecyclemgmt/operators endpoint.</td>
</tr>
<tr>
<td></td>
<td>Operator user IDs are located in the CI State Registered Users [statemgmt_register_users] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sysIds</td>
<td>Required. Comma-separated list of CI sys_ids to which to add the CI action.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
## Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types:</td>
</tr>
<tr>
<td></td>
<td><em>application/json</em> or <em>application/xml</em>.</td>
</tr>
<tr>
<td></td>
<td>Default: <em>application/json</em></td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types:</td>
</tr>
<tr>
<td></td>
<td><em>application/json</em> or <em>application/xml</em>.</td>
</tr>
<tr>
<td></td>
<td>Default: <em>application/json</em></td>
</tr>
</tbody>
</table>

## Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

## Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed by the endpoint.</td>
</tr>
<tr>
<td></td>
<td>Review the response body <code>result.result</code> parameter to verify the outcome of the operation.</td>
</tr>
<tr>
<td></td>
<td>If the endpoint encounters errors while processing the request, error codes and messages appear</td>
</tr>
<tr>
<td></td>
<td>in the response body <code>result.errors</code> parameter.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response</td>
</tr>
<tr>
<td></td>
<td>contains additional information about the error.</td>
</tr>
</tbody>
</table>
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Object encapsulating the result of the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><em>result</em>: {</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.errors</td>
<td>List of objects in which each object represents an error encountered while processing the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>result.errors.error</td>
<td>Identifier for a state management error encountered while processing the request.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_ALREADY_SET</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_NOT_SET</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_CI_ACTION_RECORD</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_OPS_STATE_RECORD</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_SYS_ID</td>
</tr>
<tr>
<td></td>
<td>• INCOMPATIBLE_CI_ACTION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_CI_ACTION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_DELETE</td>
</tr>
<tr>
<td></td>
<td>• INVALID_INPUT_PARAMETERS</td>
</tr>
<tr>
<td></td>
<td>• INVALID_LEASETIME</td>
</tr>
<tr>
<td></td>
<td>• INVALID_OPS_STATE</td>
</tr>
<tr>
<td></td>
<td>• INVALID_OPS_STATE_TRANSITION</td>
</tr>
<tr>
<td></td>
<td>• INVALID_REQUESTOR</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>INVALID_REQUESTOR_FOR_CI</td>
<td></td>
</tr>
<tr>
<td>INVALID_SYS_ID</td>
<td></td>
</tr>
<tr>
<td>MUTEX_UNAVAILABLE</td>
<td></td>
</tr>
<tr>
<td>NOT_ALLOWED_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>OPS_STATE_NOT_SET</td>
<td></td>
</tr>
<tr>
<td>UNPRIORITIZED_OPS_STATE</td>
<td></td>
</tr>
<tr>
<td>UNSUPPORTED_SYS_ID</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

result.errors.message Message providing details on the associated error.
Data type: String

result.result Flag indicating whether the CI action was set for all CIs in the list.
Possible values:
- true: CI actions were set.
- false: CI actions were not set.
Data type: String

**Example: cURL request**

Add a Patching CI action to two CIs.

curl --request POST \
"https://instance.service-now.com/api/now/cilifecyclemgmt/actions?actionName=Patching&reques\storId=621b5a09309e5010f877773aa7167c0a&sysIds=00a9a80d3790200044e0bf\c8bcbe5d1c,d0fdbc8437201000deebfc8bcbe5d33" \
   --header "Accept: application/json" \
   --header "Content-Type: application/json" \
   --user "username":"password"

```json
{
  "result": {
    "result": true
  }
}
```
## Example: Python request

Add a Provisioning CI action to the two CIs from the cURL example above.

```python
# Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request
url = "https://instance.service-now.com/api/now/cilifecyclemgmt/actions/"

# Set credentials
user = "username"
pwd = "password"

# Set query parameters
params = {
    "actionName": "Provisioning",
    "requestorId": "621b5a09309e5010f877773aa7167c0a",
    "sysIds": "00a9a80d3790200044e0bfc8bcbe5d1c,d0fdbc8437201000deeabfc8bcbe5d33"
}

# Set HTTP headers
headers = {
    "Accept": "application/json",
    "Content-Type": "application/json"
}

# Issue the HTTP request
response = requests.post(url, auth=(user, pwd), params=params, headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status:" , response.status_code, "Headers:" , response.headers, "Error Response:" , response.content)
    exit()

# Display the JSON response
print(json.dumps(response.json()))
```

The request is processed successfully, but the Provisioning CI action is not added to the two CIs because it is incompatible with the Patching CI action.

```json
{
  "result": {
    "result": false,
  }
}
```
"errors": [
  {
    "message": "Incompatible CI Actions [Provisioning] and [Patching] for sys_id 00a9a80d3790200044e0bfc8bcbe5d1c",
    "error": "INCOMPATIBLE_CI_ACTION"
  },
  {
    "message": "Incompatible CI Actions [Provisioning] and [Patching] for sys_id d0fdbc8437201000deeabfc8bcbe5d33",
    "error": "INCOMPATIBLE_CI_ACTION"
  }
]

CI Lifecycle Management - POST /now/cilifecyclemgmt/operators
Registers an operator for a non-workflow user.

URL format
Versioned URL: /api/now/{api_version}/cilifecyclemgmt/operators
Default URL: /api/now/cilifecyclemgmt/operators

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed by the endpoint. Review the response body result.result parameter to verify the outcome of the operation.</td>
</tr>
</tbody>
</table>
Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If the endpoint encounters errors while processing the request, error codes and messages appear in the response body <code>result.errors</code> parameter.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>result</code></td>
<td>Object encapsulating the result of the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><code>result.errors</code></td>
<td>List of objects in which each object represents an error encountered while processing the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td><code>result.errors.error</code></td>
<td>Identifier for a state management error encountered while processing the request.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_ALREADY_SET</td>
</tr>
<tr>
<td></td>
<td>• CI_ACTION_NOT_SET</td>
</tr>
<tr>
<td></td>
<td>• DUPLICATE_CI_ACTION_RECORD</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>• DUPLICATE_OPS_STATE_RECORD</td>
<td></td>
</tr>
<tr>
<td>• DUPLICATE_SYS_ID</td>
<td></td>
</tr>
<tr>
<td>• INCOMPATIBLE_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>• INVALID_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>• INVALID_DELETE</td>
<td></td>
</tr>
<tr>
<td>• INVALID_INPUT_PARAMETERS</td>
<td></td>
</tr>
<tr>
<td>• INVALID_LEASETIME</td>
<td></td>
</tr>
<tr>
<td>• INVALID_OPS_STATE</td>
<td></td>
</tr>
<tr>
<td>• INVALID_OPS_STATE_TRANSITION</td>
<td></td>
</tr>
<tr>
<td>• INVALID_REQUESTOR</td>
<td></td>
</tr>
<tr>
<td>• INVALID_REQUESTOR_FOR_CI</td>
<td></td>
</tr>
<tr>
<td>• INVALID_SYS_ID</td>
<td></td>
</tr>
<tr>
<td>• MUTEX_UNAVAILABLE</td>
<td></td>
</tr>
<tr>
<td>• NOT_ALLOWED_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>• OPS_STATE_NOT_SET</td>
<td></td>
</tr>
<tr>
<td>• UNPRIORITIZED_OPS_STATE</td>
<td></td>
</tr>
<tr>
<td>• UNSUPPORTED_SYS_ID</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

<table>
<thead>
<tr>
<th>result.errors.message</th>
<th>Message providing details on the associated error.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>result.requestorId</th>
<th>GUID for the registered operator. Use this value to set CI actions and operational states.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>result.result</th>
<th>Flag indicating whether the operator was successfully registered. Possible values:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• true: Operator was successfully registered.</td>
<td></td>
</tr>
<tr>
<td>• false: Operator was not successfully registered.</td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>
Example: cURL request

curl --request POST "https://instance.service-now.com/api/now/cilifecyclemgmt/operators" --header "Accept: application/json" --header "Content-Type: application/json" --user "username":"password"

{
  "result": {
    "result": true,
    "requestorId": "6f992b8430121010f877773aa7167c42"
  }
}

Example: Python request

# Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request
url = "https://instance.service-now.com/api/now/cilifecyclemgmt/operators"

# Set credentials
user = "username"
pwd = "password"

# Set HTTP headers
headers = {
    "Accept": "application/json",
    "Content-Type": "application/json"
}

# Issue the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status:", response.status_code, "Headers:", response.headers, "Error Response:", response.content)
    exit()

# Display the JSON response
print(json.dumps(response.json()))
**CI Lifecycle Management - POST /now/cilifecyclemgmt/statuses**

Sets the operational state for a specified list of configuration items (CIs).

**URL format**

Versioned URL: `/api/now/{api_version}/cilifecyclemgmt/statuses`

Default URL: `/api/now/cilifecyclemgmt/statuses`

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oldOpsLabels</td>
<td>Comma-separated list of old CI operational states in which all CIs should be included.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>opsLabel</td>
<td>Required. Label of the operational state to set for the specified CIs.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>requestorId</td>
<td>Required. Sys_id of a workflow context, or an operator user ID returned from the CI Lifecycle Management - POST /now/cilifecyclemgmt/operators endpoint. Operator user IDs are located in the CI State Registered Users [statemgmt_register_users] table.</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sysIds</td>
<td>Required. Comma-separated list of sys_ids of CIs for which the operational state is to be set.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed by the endpoint. Review the response body <code>result.result</code> parameter to verify the outcome of the operation. If the endpoint encounters errors while processing the request, error codes and messages appear in the response body <code>result.errors</code> parameter.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>result</code></td>
<td>Object encapsulating the result of the request. Data type: Object</td>
</tr>
<tr>
<td></td>
<td><code>&quot;result&quot;: {</code></td>
</tr>
<tr>
<td></td>
<td><code>&quot;errors&quot;: [Array],</code></td>
</tr>
<tr>
<td></td>
<td><code>&quot;result&quot;: &quot;String&quot;</code></td>
</tr>
<tr>
<td></td>
<td><code>}</code></td>
</tr>
<tr>
<td><code>result.errors</code></td>
<td>List of objects in which each object represents an error encountered while processing the request. Data type: Array</td>
</tr>
<tr>
<td></td>
<td><code>&quot;errors&quot;: [</code></td>
</tr>
<tr>
<td></td>
<td><code>{</code></td>
</tr>
<tr>
<td></td>
<td><code>&quot;error&quot;: &quot;String&quot;,</code></td>
</tr>
<tr>
<td></td>
<td><code>&quot;message&quot;: &quot;String&quot;</code></td>
</tr>
<tr>
<td></td>
<td><code>}</code></td>
</tr>
<tr>
<td></td>
<td><code>]</code></td>
</tr>
<tr>
<td><code>result.errors.error</code></td>
<td>Identifier for a state management error encountered while processing the request. Possible values:</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>• CI_ACTION_ALREADY_SET</td>
<td></td>
</tr>
<tr>
<td>• CI_ACTION_NOT_SET</td>
<td></td>
</tr>
<tr>
<td>• DUPLICATE_CI_ACTION_RECORD</td>
<td></td>
</tr>
<tr>
<td>• DUPLICATE_OPS_STATE_RECORD</td>
<td></td>
</tr>
<tr>
<td>• DUPLICATE_SYS_ID</td>
<td></td>
</tr>
<tr>
<td>• INCOMPATIBLE_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>• INVALID_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>• INVALID_DELETE</td>
<td></td>
</tr>
<tr>
<td>• INVALID_INPUT_PARAMETERS</td>
<td></td>
</tr>
<tr>
<td>• INVALID_LEASETIME</td>
<td></td>
</tr>
<tr>
<td>• INVALID_OPS_STATE</td>
<td></td>
</tr>
<tr>
<td>• INVALID_OPS_STATE_TRANSITION</td>
<td></td>
</tr>
<tr>
<td>• INVALID_REQUESTOR</td>
<td></td>
</tr>
<tr>
<td>• INVALID_REQUESTOR_FOR_CI</td>
<td></td>
</tr>
<tr>
<td>• INVALID_SYS_ID</td>
<td></td>
</tr>
<tr>
<td>• MUTEX_UNAVAILABLE</td>
<td></td>
</tr>
<tr>
<td>• NOT_ALLOWED_CI_ACTION</td>
<td></td>
</tr>
<tr>
<td>• OPS_STATE_NOT_SET</td>
<td></td>
</tr>
<tr>
<td>• UNPRIORITIZED_OPS_STATE</td>
<td></td>
</tr>
<tr>
<td>• UNSUPPORTED_SYS_ID</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

<table>
<thead>
<tr>
<th>result.errors.message</th>
<th>Message providing details on the associated error.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>result.result</th>
<th>Flag indicating whether the desired state was set for all CIs included in the request. Possible values:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• true: Operational state set.</td>
</tr>
<tr>
<td></td>
<td>• false: Operational state not set.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

Set the Repair in Progress operational state for two CIs.

curl --request POST "https://instance.service-now.com/api/now/cilifecyclemgmt/statuses?opsLabel=Repair%20in%20Progress&requestorId=23d58bc030121010f877773aa7167c83&sysIds=affd3c8437201000deabfc8bce5dc3,aac0b1213784200044e0bfc8bce5de3" --header "Accept: application/json" --header "Content-Type: application/json" --user "username":"password"

The response body shows that the desired operational state was set for both CIs specified in the request.

```
{
  "result": {
    "result": true
  }
}
```

Example: Python request

Set the Retired operational state for a CI.

```python
# Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request
url = "https://instance.service-now.com/api/now/cilifecyclemgmt/statuses"

# Set credentials
user = "username"
pwd = "password"

# Set query parameters
params = {
    "opsLabel": "Retired",
    "requestorId": "06168bc030121010f87773aa7167c86",
    "sysIds": "780b8ce8735423002728660c4cf6a7e7"
}

# Set HTTP headers
headers = {
    "Accept": "application/json",

```
The response indicates that the specified `requestorId` is not a valid workflow context `sys_id` or operator user ID.

```json
{
  "result": {  
    "result": false,
    "errors": [  
      {  
        "message": "Requestor [06168bc030121010f877773aa7167c86] is not registered",
        "error": "INVALID_REQUESTOR"
      }  
    ]
  }
}
```

**CMDB Data Ingestion API**

The CMDB Data Ingestion API enables the batch ingestion of an array of objects into an Import Set table.

**Note:** With the release of Quebec, the Import Set API has added the `insertMultiple` endpoint. This new endpoint is strongly preferred over the CMDB Data Ingestion API as it has benefits for both scale and performance. From Quebec onward, any usage of the CMDB Data Ingestion API should be migrated to use the `insertMultiple` endpoint.

This API is activated through the Configuration Management Database (CMDB) (com.snc.cmdb) plugin.
CMDB Data Ingestion - POST /cmdb/ingest/{data_source_sys_id}

Inserts records into the Import Set table associated with the data source record identified by the passed-in sys_id.

The request body must contain the JSON array of objects (payload) to insert in the Import Set table. Each object equates to a row in the table, each name-value pair equates to a column. The JSON payload must leverage the field names from the import set without the "u_" prefix. For example, the field name "u_matching_record" should be "matching_record" in the request body payload. If the Import Set table exists, the endpoint appends the rows (objects) to the existing Import Set table. No checking for duplicates or updating of existing records is performed.

If you are initially building an application, you must first create the associated data source record in your instance before calling this endpoint. If you are just using this endpoint to add records to an existing Import Set table, you do not need to create the data source record, but you must know its sys_id. The data source record describes the Import Set table in which to insert the specified payload. This table must extend the Import Set Rows [sys_import_set_row] table. Also, the data source must be set to Attachment and the format set to JSON. For more information on data sources, see Data sources.

If the Import Set table defined in the data source record does not exist, the endpoint attaches the passed-in payload to the data source record. To create the initial Import Set table, you must manually import the data into the Import Set table. To import the data, on the associated Data Source form, click the Test Load 20 Records or Load All Records link in the Related Links section. Once the Import Set table is created, you cannot add columns to the table using this endpoint. If name-value pairs are later passed in that do not exist in the Import Set table, they are ignored without warning. If you need to modify the columns within the Import Set table, you can manually add them to the table. You can also delete or rename the Import Set table, and call the endpoint again using the new payload.

You must have the cmdb_import_api_admin role to access this endpoint.

**URL format**

Versioned URL: /api/now/{api_version}/cmdb/ingest/{data_source_sys_id}

Default URL: /api/now/cmdb/ingest/{data_source_sys_id}
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>data_source_sys_id</td>
<td>Sys_id of the data source record.</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>Free-form array of objects that describe the data to append to the associated Import Set table. Each object in the array defines a row in the Import Sets table; each name-value pair a column.</td>
</tr>
</tbody>
</table>

**Note:** This array must be named, such as "{"records":{"hostname": "Hostname1", "serialnumber": "2acd3873-7fc5-454c-8844-e7769e4d6cfc", "model": "Model Id"},{"vendor": "ABC Co"}}".

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json OR application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created. An attachment was added to the data source.</td>
</tr>
<tr>
<td>202</td>
<td>Accepted. Rows were added to the Import Set table.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>409</td>
<td>Conflict. Attachment already exists on the data source.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
<tr>
<td>501</td>
<td>Not Implemented. Request format is not supported.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Describes an encountered error. Data type: Object</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>error</td>
<td>Additional information about the error.</td>
</tr>
<tr>
<td>error.message</td>
<td>Message describing the error.</td>
</tr>
<tr>
<td>import_set</td>
<td>Name of the Import Set table to which the payload was appended.</td>
</tr>
<tr>
<td>staged_row_count</td>
<td>Number of rows appended to the Import Set table.</td>
</tr>
<tr>
<td>staging_table</td>
<td>Name of the data source record used to stage the payload.</td>
</tr>
<tr>
<td>status</td>
<td>Error status.</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

```bash
curl "instance.service-now.com/api/now/cmdb/ingest/4dd9686d1b9800103d374087bc4bcb3d" \
--request POST \
--header "Accept: application/json" \
--header "Content-Type:application/json" \
--data "{"records": [{"hostname": "Hostname1", "serialnumber": "2acd3873-7fc5-454c-8844-e7769e4d6cfc", "model": "Model 5100"},{"vendor": "ABC Co"}],
{"hostname": "Hostname2", "serialnumber": "3adb3873-7fc5-564d-8844-e7769e4d6ded", "model": "Model 5200"},{"vendor": "ACME Co"}]"
--user "username":"password"
```

**Successful**

```json
{
    "result": {
        "staged_row_count": 2,
        ...
    }
}
```
Example: Sample Python request

```python
# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/cmdb/ingest/4dd9686d1b9800103d374087bc4bcb3d'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Content-Type': 'application/json', 'Accept': 'application/xml'}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, data='{"records": [{"hostname": "Hostname1", "serialnumber": "2acd3873-7fc5-454c-8844-e7769e4d6cfc", "model": "Model 5100"}, {"vendor": "ABC Co"}], [{"hostname": "Hostname2", "serialnumber": "3adb3873-7fc5-564d-8844-e7769e4d6ded", "model": "Model 5200"}, {"vendor": "ACME Co"}]}')

# Check for HTTP codes other than 202
if response.status_code != 202:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

Error response

{
    "error": {
        "message": "No request body provided",
        "detail": "Confirm your request body is correct"
    }
}
```
CMDB Instance API

Performs create, read, update, and delete operations on existing CMDB tables. Only one record can be inserted, updated, or deleted at a time.

This API requires the user have the ITIL role.

CMDB Instance - GET /now/cmdb/instance/{classname}

Returns the available configuration items (CI) for a specified Configuration Management Database (CMDB) class (table).

URL format

Versioned URL: /api/now/v1/cmdb/instance/{className}
Default URL: /api/now/cmdb/instance/{className}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>className</td>
<td>CMDB class name. This is the name of the table that contains the desired CI records, such as cmdb_ci_linux_server or cmdb_ci_apache_web_server. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. For requests that exceed this number of records, use the sysparm_offset parameter to paginate record retrieval. This limit is applied before ACL evaluation. If no records return, including records you have access to, rearrange the record order so records you have access to return first. Note: Unusually large sysparm_limit values can impact system performance.</td>
</tr>
<tr>
<td>Data type: Number</td>
<td></td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default: 1000</strong></td>
<td></td>
</tr>
<tr>
<td><strong>sysparm_offset</strong></td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, <code>sysparm_offset</code> is set to &quot;0&quot;. To simply page through all available records, use <code>sysparm_offset=sysparm_offset + sysparm_limit</code>, until you reach the end of all records. Do not pass a negative number in the <code>sysparm_offset</code> parameter. Data type: Number Default: 0</td>
</tr>
<tr>
<td><strong>sysparm_query</strong></td>
<td>Encoded query used to filter the result set. Syntax: <code>sysparm_query=&lt;col_name&gt;&lt;operator&gt;&lt;value&gt;</code>.</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;col_name&gt;</code>: Name of the table column to filter against.</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;operator&gt;</code>: Supports the following values:</td>
</tr>
<tr>
<td></td>
<td>◦ <code>=</code>: Exactly matches <code>&lt;value&gt;</code>.</td>
</tr>
<tr>
<td></td>
<td>◦ <code>!</code>: Does not match <code>&lt;value&gt;</code>.</td>
</tr>
<tr>
<td></td>
<td>◦ <code>^</code>: Logically AND multiple query statements.</td>
</tr>
<tr>
<td></td>
<td>◦ <code>^OR</code>: Logically OR multiple query statements.</td>
</tr>
<tr>
<td></td>
<td>◦ LIKE: <code>&lt;col_name&gt;</code> contains the specified string. Only works for <code>&lt;col_name&gt;</code> fields whose data type is string.</td>
</tr>
<tr>
<td></td>
<td>◦ <code>STARTSWITH</code>: <code>&lt;col_name&gt;</code> starts with the specified string. Only works for <code>&lt;col_name&gt;</code> fields whose data type is string.</td>
</tr>
<tr>
<td></td>
<td>◦ <code>ENDSWITH</code>: <code>&lt;col_name&gt;</code> ends with the specified string. Only works for <code>&lt;col_name&gt;</code> fields whose data type is string.</td>
</tr>
<tr>
<td></td>
<td><code>&lt;value&gt;</code>: Value to match against.</td>
</tr>
<tr>
<td></td>
<td>All parameters are case-sensitive. Queries can contain more than one entry, as <code>sysparm_query=&lt;col_name&gt;&lt;operator&gt;&lt;value&gt;&lt;operator&gt;&lt;col_name&gt;&lt;operator&gt;&lt;value&gt;</code>.</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td><code>{sysparm_query=caller_id=javascript:gs.getUserID()^active=true}</code></td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Encoded queries also supports order by functionality. To sort responses based on certain fields, use the ORDERBY and ORDERBYDESC clauses in sysparm_query.</td>
</tr>
<tr>
<td></td>
<td>Syntax:</td>
</tr>
<tr>
<td></td>
<td>• ORDERBY&lt;col_name&gt;</td>
</tr>
<tr>
<td></td>
<td>• ORDERBYDESC&lt;col_name&gt;</td>
</tr>
<tr>
<td></td>
<td>For example: sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory</td>
</tr>
<tr>
<td></td>
<td>This query filters all active records and orders the results in ascending order by number, and then in descending order by category.</td>
</tr>
<tr>
<td></td>
<td>If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property glide.invalid_query.returns_no_rows. Set this property to true to return no rows on an invalid query.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The glide.invalid_query.returns_no_rows property controls the behavior of all queries across the instance, such as in lists, scripts (GlideRecord.query()), and web service APIs.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml.</td>
</tr>
</tbody>
</table>
Request headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of a CI within the specified class.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Example: Sample cURL request

```
curl "https://instance.servicenow.com/api/now/cmdb/instance/cmdb_ci_linux_server" \
--request GET \
--header "Accept:application/json" \
--user 'username':'password'
```

"result": [
  {
    "sys_id": "3a290cc60a0a0bb400000bdb386af1cf",
    "name": "PS LinuxApp01"
  },
  {
    "sys_id": "3a5dd3dbc0a8ce0100655f1ec66ed42c",
    "name": "PS LinuxApp02"
  }
]

CMDB Instance - DELETE /now/cmdb/instance/{classname}/{sys_id}/relation/{rel_sys_id}

Deletes the relation for the specified configuration item (CI).

**URL format**

Versioned URL: /api/now/v1/cmdb/instance/{className}/{sys_id}/relation/{rel_sys_id}

Default URL: /api/now/cmdb/instance/{className}/{sys_id}/relation/{rel_sys_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>className</td>
<td>CMDB class name. This is the name of the table that contains the desired CI records, such as cmdb_ci_linux_server or cmdb_ci_apache_web_server. Data type: String</td>
</tr>
<tr>
<td>rel_sys_id</td>
<td>Sys_id of the relation to remove. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the CI.</td>
</tr>
</tbody>
</table>
### Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Example: Sample cURL request

```bash
curl "https://instance.servicenow.com/api/now/cmdb/instance/cmdb_ci_linux_server/0f4ac6c4b750230096c3e4f6ee11a9fe/relation/b0dbda5347c12200e0ef56d9a718f" \
--request DELETE \
--user 'username':'password'
```

### CMDB Instance - GET /now/cmdb/instance/{classname}/{sys_id}

Returns attributes and relationship information for a specified configuration item (CI) record.

### URL format

- **Versioned URL**: `/api/now/{api_version}/cmdb/instance/{className}/{sys_id}`
- **Default URL**: `/api/now/cmdb/instance/{className}/{sys_id}`
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>className</td>
<td>CMDB class name. This is the name of the table that contains the desired CI records, such as cmdb_ci_linux_server or cmdb_ci_apache_web_server. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the CI record to retrieve. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json Or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

Response parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Data attributes currently in the CI record. The available attributes depend on the specified CMDB class. You can locate the available attributes in the associated CMDB table which typically begins with &quot;cmdb_ci&quot;, such as cmdb_ci_linux_server or cmdb_ci_mfp_printer.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
</tbody>
</table>
## Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Information on any errors encountered while processing the endpoint request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;detail&quot;: &quot;String&quot;,</td>
</tr>
</tbody>
</table>
|                         |   "message": "String"
<p>| error.detail             | Additional information about the error.                                    |
|                          | Data type: String                                                           |
| error.message            | Message that identifies the error.                                         |
|                          | Data type: String                                                           |
| inbound_relations        | List of objects that define the inbound relations associated with the CI.   |
|                          | Data type: Array                                                            |
|                         | &quot;inbound_relations&quot;: [                                                     |
|                         |   {                                                                        |
|                         |     &quot;sys_id&quot;: &quot;String&quot;,                                                   |
|                         |     &quot;target&quot;: {Object},                                                   |
|                         |     &quot;type&quot;: {Object}                                                      |
|                         | }                                                                          |
| inbound_relations.sys_id | Sys_id of the inbound CI relationship. Located in the CI Relationship [cmdb_rel_ci] table. |
|                          | Data type: String                                                           |
| inbound_relations.target | Information that describes the target inbound relation.                    |
|                          | Data type: Object                                                           |
|                         | &quot;target&quot;: {                                                                |
|                         |   &quot;display_value&quot;: &quot;String&quot;,                                              |
|                         |   &quot;display_value&quot;: &quot;String&quot;                                               |
|                         |   &quot;display_value&quot;: &quot;String&quot;                                               |
|                         | }                                                                          |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>inbound_relations.target.display_value</strong></td>
<td>Name of the inbound relation to display in a user interface. Data type: String</td>
</tr>
<tr>
<td><strong>inbound_relations.target.link</strong></td>
<td>CMDB REST endpoint URL to use to retrieve the inbound relation from the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td><strong>inbound_relations.target.value</strong></td>
<td>Sys_id of the inbound CI. Located in the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td><strong>inbound_relations.type</strong></td>
<td>Information about the type of inbound relationship. Data type: Object</td>
</tr>
<tr>
<td><strong>inbound_relations.type.display_value</strong></td>
<td>Name of the inbound type to display in a user interface. Data type: String</td>
</tr>
<tr>
<td><strong>inbound_relations.type.link</strong></td>
<td>Table REST endpoint URL to use to retrieve the inbound relation type from the CI Relationship Type [cmdb_rel_type] table. Data type: String</td>
</tr>
<tr>
<td><strong>inbound_relations.type.value</strong></td>
<td>Sys_id of the type of inbound relationship. Located in the CI</td>
</tr>
</tbody>
</table>
## Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Relationship Type</strong></td>
<td>[cmdb_rel_type] table.</td>
</tr>
<tr>
<td><strong>Data type:</strong> String</td>
<td></td>
</tr>
<tr>
<td><strong>outbound_relations</strong></td>
<td>List of objects that define the outbound relations associated with the CI.</td>
</tr>
<tr>
<td><strong>Data type:</strong> Array</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;outbound_relations&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;target&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: {Object}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td><strong>outbound_relations.sys_id</strong></td>
<td>Sys_id of the outbound CI relationship. Located in the CI Relationship</td>
</tr>
<tr>
<td></td>
<td>[cmdb_rel_ci] table.</td>
</tr>
<tr>
<td><strong>Data type:</strong> String</td>
<td></td>
</tr>
<tr>
<td><strong>outbound_relations.target</strong></td>
<td>Required. Information about the target outbound relation.</td>
</tr>
<tr>
<td><strong>Data type:</strong> Object</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;target&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;display_value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;link&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><strong>outbound_relations.target.display_value</strong></td>
<td>Name of the outbound relation to display in a user interface.</td>
</tr>
<tr>
<td><strong>Data type:</strong> String</td>
<td></td>
</tr>
<tr>
<td><strong>outbound_relations.target.link</strong></td>
<td>CMDB REST endpoint URL to use to retrieve the outbound relation from</td>
</tr>
<tr>
<td></td>
<td>the Configuration Item [cmdb_ci] table.</td>
</tr>
</tbody>
</table>
### Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>outbound_relations.target.value</strong></td>
<td>Sys_id of the outbound CI. Located in the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>outbound_relations.type</strong></td>
<td>Information about the type of outbound relationship.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td><img src="object.png" alt="Object" /></td>
<td></td>
</tr>
<tr>
<td><strong>outbound_relations.type.display_value</strong></td>
<td>Name of the outbound type to display in a user interface.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>outbound_relations.type.link</strong></td>
<td>Table REST endpoint URL to use to retrieve the outbound relation from the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>outbound_relations.type.value</strong></td>
<td>Sys_id of the type of outbound relationship. Located in the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>status</strong></td>
<td>Only appears if an error is encountered. Status of the endpoint processing. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• failure</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Example: Sample cURL request

```
curl
  "https://instance.servicenow.com/api/now/cmdb/instance/cmdb_ci_linux_server/70bc4e11c0a8210d01f074cbe6bd73b4" \n  --request GET \n  --header "Accept:application/json" \n  --user "username":"password"
```
"attributes": {
  "firewall_status": "Intranet",
  "os_address_width": "",
  "attested_date": "",
  "operational_status": "1",
  "os_service_pack": "",
  "cpu_core_thread": "",
  "cpu_manufacturer": "",
  "sys_updated_on": "2020-07-08 11:16:51",
  "discovery_source": "",
  "first_discovered": "",
  "due_in": "",
  "used_for": "Production",
  "invoice_number": "",
  "gl_account": "",
  "sys_created_by": "glide.maint",
  "warranty_expiration": "",
  "ram": "2048",
  "cpu_name": "",
  "cpu_speed": "2800",
  "owned_by": "",
  "checked_out": "",
  "kernel_release": "",
  "sys_domain_path": "/",
  "classification": "Production",
  "disk_space": "40",
  "object_id": "",
  "maintenance_schedule": "",
  "cost_center": "",
  "attested_by": "",
  "dns_domain": "",
  "assigned": "2020-01-04 07:00:00",
  "purchase_date": "",
  "life_cycle_stage": "",
  "short_description": "",
  "cd_speed": "",
  "floppy": "",
  "managed_by": {
    "display_value": "Lynda Caraway"},
}
"os": "Linux Red Hat",
"sys_mod_count": "24",
"monitor": "false",
"model_id": {
  "display_value": "Iris 5875",
  "link": "https://instance.servicenow.com/api/now/table/cmdb_model/5f5fbcc3c0a8010e00f3b27814f3b96b",
  "value": "5f5fbcc3c0a8010e00f3b27814f3b96b"
},
"ip_address": "",
"duplicate_of": "",
"sys_tags": "",
"cost_cc": "USD",
"order_date": "",
"schedule": "",
"environment": "",
"due": "",
"attested": "false",
"location": {
  "display_value": "322 West 52nd Street, New York, NY",
  "link": "https://instance.servicenow.com/api/now/table/cmn_location/25ab9f690a0a0bb3001c5fec1d07bcb",
  "value": "25ab9f690a0a0bb3001c5fec1d07bcb"
},
"category": "Do not migrate to asset",
"fault_count": "0",
"host_name": "",
"lease_id": ""
},
"inbound_relations": [
  {
    "sys_id": "3a5e4d8ac0a8ce010005145af8370818",
    "type": {
      "display_value": "Depends on::Used by",
      "link": "https://instance.servicenow.com/api/now/table/cmdb_rel_type/1a9cb166f1571100a92eb60da2bce5c5",
      "value": "1a9cb166f1571100a92eb60da2bce5c5"
    },
    "target": {
      "display_value": "PS Apache01",
      "link": "https://instance.servicenow.com/api/now/table/cmdb_asset/4f5fbcc3c0a8010e00f3b27814f3b96b"
    }
  }
]
"link": "https://instance.servicenow.com/api/now/cmdb/instance/cmdb_ci/3a27d4370a0a0bb4006316812bf45439",

"value": "3a27d4370a0a0bb4006316812bf45439"
}
},
{
"sys_id": "3a5e4d9cc0a8ce010097f2f5c2f65fd8",
"type": {
"display_value": "Depends on::Used by",
"link": "https://instance.servicenow.com/api/now/table/cmdb_rel_type/1a9cb166f1571100a92eb60da2bce5c5",

"value": "1a9cb166f1571100a92eb60da2bce5c5"
},
"target": {
"display_value": "PS Apache02",
"link": "https://instance.servicenow.com/api/now/cmdb/instance/cmdb_ci/3a27f1520a0a0bb400ecd6ff7acf036",

"value": "3a27f1520a0a0bb400ecd6ff7acf036"
}
},
{
"sys_id": "3a5e4d9fc0a8ce0100a3754fac26fe56",
"type": {
"display_value": "Depends on::Used by",
"link": "https://instance.servicenow.com/api/now/table/cmdb_rel_type/1a9cb166f1571100a92eb60da2bce5c5",

"value": "1a9cb166f1571100a92eb60da2bce5c5"
},
"target": {
"display_value": "PS Apache03",
"link": "https://instance.servicenow.com/api/now/cmdb/instance/cmdb_ci/3a2810c20a0a0bb400268337d6e942ca",

"value": "3a2810c20a0a0bb400268337d6e942ca"
}
}
CMDB Instance - POST /now/cmdb/instance/{classname}
Creates a single configuration item (CI) with the specified outbound and inbound relations within the specified Configuration Management Database (CMDB) table.

⚠️ Note: If the CMDB class has a dependent identification rule defined, then the input payload must contain the sys_id of the parent CI.

URL format
Versioned URL: /api/now/v1/cmdb/instance/{className}
Default URL: /api/now/cmdb/instance/{className}

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>className</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Request body parameters (XML or JSON)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>attributes</td>
</tr>
<tr>
<td>inbound_relations</td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>-------------------------------</td>
</tr>
<tr>
<td><strong>Data type: Array</strong></td>
</tr>
<tr>
<td>&quot;inbound_relations&quot;: [</td>
</tr>
<tr>
<td>{</td>
</tr>
<tr>
<td>&quot;target&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;type&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>}</td>
</tr>
<tr>
<td>inbound_relations.target</td>
</tr>
<tr>
<td>inbound_relations.type</td>
</tr>
<tr>
<td>outbound_relations</td>
</tr>
<tr>
<td>&quot;outbound_relations&quot;: [</td>
</tr>
<tr>
<td>{</td>
</tr>
<tr>
<td>&quot;target&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;type&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>}</td>
</tr>
<tr>
<td>outbound_relations.target</td>
</tr>
<tr>
<td>outbound_relations.type</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>source</td>
<td>Required. Entity that updated the information. This must be one of the choice values specified in the discovery_source field in the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

#### Response parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Data attributes currently in the CI record. The available attributes depend on the specified CMDB class. You can locate the available attributes in the associated CMDB table which typically begins with &quot;cmdb_ci&quot;, such as cmdb_ci_linux_server or cmdb_ci_mfp_printer. Data type: Object</td>
</tr>
<tr>
<td>error</td>
<td>Information on any errors encountered while processing the endpoint request. Data type: Object</td>
</tr>
</tbody>
</table>

```
"error": {
    "detail": "String",
    "message": "String"
}
```
### Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error.detail</td>
<td>Additional information about the error. Data type: String</td>
</tr>
<tr>
<td>error.message</td>
<td>Message that identifies the error. Data type: String</td>
</tr>
<tr>
<td>inbound_relations</td>
<td>List of objects that define the inbound relations associated with the CI. Data type: Array</td>
</tr>
<tr>
<td>&quot;inbound_relations&quot;: [</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;target&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: {Object}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>inbound_relations.sys_id</td>
<td>Sys_id of the inbound CI relationship. Located in the CI Relationship [cmdb_rel_ci] table. Data type: String</td>
</tr>
<tr>
<td>inbound_relations.target</td>
<td>Information that describes the target inbound relation. Data type: Object</td>
</tr>
<tr>
<td>&quot;target&quot;: {</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;display_value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;link&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>inbound_relations.target.display_value</td>
<td>Name of the inbound relation to display in a user interface. Data type: String</td>
</tr>
<tr>
<td>inbound_relations.target.link</td>
<td>CMDB REST endpoint URL to use to retrieve the inbound relation from</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>inbound_relations.target.value</td>
<td>Sys_id of the inbound CI. Located in the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>inbound_relations.type</td>
<td>Information about the type of inbound relationship.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;display_value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;link&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>inbound_relations.type.display_value</td>
<td>Name of the inbound type to display in a user interface.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>inbound_relations.type.link</td>
<td>Table REST endpoint URL to use to retrieve the inbound relation type from the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>inbound_relations.type.value</td>
<td>Sys_id of the type of inbound relationship. Located in the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>outbound_relations</td>
<td>List of objects that define the outbound relations associated with the CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>
### Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>outbound_relations.sys_id</code></td>
<td>Sys_id of the outbound CI relationship. Located in the CI Relationship [cmdb_rel_ci] table. Data type: String</td>
</tr>
<tr>
<td><code>outbound_relations.target</code></td>
<td>Required. Information about the target outbound relation. Data type: Object</td>
</tr>
<tr>
<td><code>outbound_relations.target.display_value</code></td>
<td>Name of the outbound relation to display in a user interface. Data type: String</td>
</tr>
<tr>
<td><code>outbound_relations.target.link</code></td>
<td>CMDB REST endpoint URL to use to retrieve the outbound relation from the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td><code>outbound_relations.target.value</code></td>
<td>Sys_id of the outbound CI. Located in the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td><code>outbound_relations.type</code></td>
<td>Information about the type of outbound relationship.</td>
</tr>
</tbody>
</table>
### Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;display_value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;link&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>outbound_relations.type.display_value</td>
<td>Name of the outbound type to display in a user interface.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>outbound_relations.type.link</td>
<td>Table REST endpoint URL to use to retrieve the outbound relation from the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>outbound_relations.type.value</td>
<td>Sys_id of the type of outbound relationship. Located in the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>Only appears if an error is encountered. Status of the endpoint processing. Possible values:</td>
</tr>
<tr>
<td></td>
<td>failure</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Example: Sample cURL request

```bash
curl "https://instance.servicenow.com/api/now/cmdb/instance/cmdb_ci_linux_server" \
    --request POST \
    --header "Accept:application/json" \
    --header "Content-Type:application/json"\n    --user "username":"password" \
    --data "outbound_relations": [ 
```
"type": {
"display_value": "Exchanges data with::Exchanges data with",
"link": "https://instance.servicenow.com/api/now/table/cmdb_rel_type/607ad1b2c0a8010e01941856b365af90",
"value": "607ad1b2c0a8010e01941856b365af90"
},
"target": {
"display_value": "PS ORA01",
"link": "https://instance.servicenow.com/api/now/cmdb_instance/cmdb_ci/3a307c930a0a0bb400353965d0b8861f",
"value": "3a307c930a0a0bb400353965d0b8861f"
}
},
"attributes": {
"firewall_status": "Intranet",
"os_address_width": "",
"attested_date": "",
"operational_status": "1",
"os_service_pack": "",
"cpu_core_thread": "",
"cpu_manufacturer": "",
"sys_updated_on": "2020-07-13 20:27:28",
"discovery_source": "ServiceNow",
"first_discovered": "2020-07-13 20:27:28",
"due_in": "",
"used_for": "Production",
"invoice_number": "",
"gl_account": "",
"sys_created_by": "dora.gray",
"warranty_expiration": "",
"ram": "",
"cpu_name": "",
"cpu_speed": "",
"owned_by": "",
"checked_out": "",
"kernel_release": "",
"sys_domain_path": "/",
"classification": "Production",
"disk_space": "",
"object_id": "",
"maintenance_schedule": ""
"cost_center": "",
"attested_by": "",
"dns_domain": "",
"assigned": "",
"purchase_date": "",
"life_cycle_stage": "",
"short_description": "",
"cd_speed": "",
"floppy": "",
"managed_by": "",
"os_domain": "",
"last_discovered": "2020-07-13 20:27:28",
"can_print": "false",
"sys_class_name": "cmdb_ci_linux_server",
"manufacturer": "",
"cpu_count": "",
"vendor": "",
"life_cycle_stage_status": "",
"model_number": "",
"assigned_to": "",
"start_date": "",
"os_version": "",
"serial_number": "",
"cd_rom": "false",
"support_group": "",
"unverified": "false",
"correlation_id": "",
"attributes": "",
"asset": "",
"form_factor": "",
"cpu_core_count": "",
"skip_sync": "false",
"attestation_score": "",
"sys_updated_by": "dora.gray",
"sys_created_on": "2020-07-13 20:27:28",
"sys_domain": { 
  "display_value": "global",
  "link": 
  "https://instance.servicenow.com/api/now/table/sys_user_group/global",
  "value": "global"
 },
"cpu_type": "",
"install_date": "",
"asset_tag": ""
```
},
"inbound_relations": [
{
  "sys_id": "c03ff2641b425010593876a61a4bcb49",
  "type": {
    "display_value": "Depends on::Used by",
    "link": "https://instance.servicenow.com/api/now/table/cmdb_rel_type/1a9cb166f1571100a92eb60da2bce5c5",
    "value": "1a9cb166f1571100a92eb60da2bce5c5"
  },
  "target": {
    "display_value": "PS Apache01",
    "link": "https://instance.servicenow.com/api/now/cmdb/instance/cmdb_ci/3a27d4370a0a0bb4006316812bf45439",
    "value": "3a27d4370a0a0bb4006316812bf45439"
  }
}
]
}

CMDB Instance - PUT /now/cmdb/instance/{classname}/{sys_id}

Updates the specified configuration item (CI) record with the specified attribute name-value pairs. This endpoint does not create, update, or delete the associated relations.

⚠️ Note: This endpoint does not conform to the HTML specifications. Standard PUT endpoints are destructive in that they remove any elements that are not included in the request. This endpoint does not remove non-specified elements.

**URL format**

Versioned URL: /api/now/v1/cmdb/instance/{className}/{sys_id}

Default URL: /api/now/cmdb/instance/{className}/{sys_id}
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>className</td>
<td>CMDB class name. This is the name of the table that contains the desired CI records, such as cmdb_ci_linux_server or cmdb_ci_apache_web_server.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the CI record to update.</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Data attributes to replace in the CI record. The available attributes depend on the specified CMDB class. You can locate the available attributes in the associated CMDB table which typically begins with &quot;cmdb_ci&quot;, such as cmdb_ci_linux_server or cmdb_ci_mfp_printer.</td>
</tr>
<tr>
<td>source</td>
<td>Required. Entity that updated the information. This must be one of the choice values specified in the discovery_source field in the Configuration Item [cmdb_ci] table.</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
# Response body parameters (JSON or XML)

## Response parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>attributes</code></td>
<td>Data attributes currently in the CI record. The available attributes depend on the specified CMDB class. You can locate the available attributes in the associated CMDB table which typically begins with &quot;cmdb_ci&quot;, such as cmdb_ci_linux_server or cmdb_ci_mfp_printer. Data type: Object</td>
</tr>
<tr>
<td><code>error</code></td>
<td>Information on any errors encountered while processing the endpoint request. Data type: Object</td>
</tr>
<tr>
<td><code>error.detail</code></td>
<td>Additional information about the error. Data type: String</td>
</tr>
<tr>
<td><code>error.message</code></td>
<td>Message that identifies the error. Data type: String</td>
</tr>
<tr>
<td><code>inbound_relations</code></td>
<td>List of objects that define the inbound relations associated with the CI. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"error": {
  "detail": "String",
  "message": "String"
}
```
### Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inbound_relations.sys_id</td>
<td>Sys_id of the inbound CI relationship. Located in the CI Relationship [cmdb_rel_ci] table. Data type: String</td>
</tr>
<tr>
<td>inbound_relations.target</td>
<td>Information that describes the target inbound relation. Data type: Object</td>
</tr>
<tr>
<td>inbound_relations.target.display_value</td>
<td>Name of the inbound relation to display in a user interface. Data type: String</td>
</tr>
<tr>
<td>inbound_relations.target.link</td>
<td>CMDB REST endpoint URL to use to retrieve the inbound relation from the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>inbound_relations.target.value</td>
<td>Sys_id of the inbound CI. Located in the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>inbound_relations.type</td>
<td>Information about the type of inbound relationship. Data type: Object</td>
</tr>
</tbody>
</table>
## Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| inbound_relations.type.display_value | Name of the inbound type to display in a user interface.  
Data type: String               |
| inbound_relations.type.link       | Table REST endpoint URL to use to retrieve the inbound relation type from the CI Relationship Type [cmdb_rel_type] table. 
Data type: String                |
| inbound_relations.type.value      | Sys_id of the type of inbound relationship. Located in the CI Relationship Type [cmdb_rel_type] table. 
Data type: String                |
| outbound_relations                | List of objects that define the outbound relations associated with the CI.  
Data type: Array                  |
| outbound_relations.sys_id         | Sys_id of the outbound CI relationship. Located in the CI Relationship [cmdb_rel_ci] table. 
Data type: String                |
| outbound_relations.target         | Required. Information about the target outbound relation.  
Data type: Object                 |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>outbound_relations.target.display_value</td>
<td>Name of the outbound relation to display in a user interface.</td>
<td>String</td>
</tr>
<tr>
<td>outbound_relations.target.link</td>
<td>CMDB REST endpoint URL to use to retrieve the outbound relation from the Configuration Item [cmdb_ci] table.</td>
<td>String</td>
</tr>
<tr>
<td>outbound_relations.target.value</td>
<td>Sys_id of the outbound CI. Located in the Configuration Item [cmdb_ci] table.</td>
<td>String</td>
</tr>
<tr>
<td>outbound_relations.type</td>
<td>Information about the type of outbound relationship.</td>
<td>Object</td>
</tr>
<tr>
<td>outbound_relations.type.display_value</td>
<td>Name of the outbound type to display in a user interface.</td>
<td>String</td>
</tr>
<tr>
<td>outbound_relations.type.link</td>
<td>Table REST endpoint URL to use to retrieve the outbound relation from the CI Relationship Type [cmdb_rel_type] table.</td>
<td>String</td>
</tr>
</tbody>
</table>
Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>outbound_relations.type.value</code></td>
<td>Sys_id of the type of outbound relationship. Located in the CI Relationship Type [cmdb_rel_type] table. Data type: String</td>
</tr>
<tr>
<td><code>status</code></td>
<td>Only appears if an error is encountered. Status of the endpoint processing. Possible values: • failure Data type: String</td>
</tr>
</tbody>
</table>

Example: Sample cURL request

```bash
curl
  "https://instance.servicenow.com/api/now/cmdb/instance/cmdb_ci_linux_server/70bc4e11c0a8210d01f074cbe6bd73b4" \n  --request PUT \n  --header "Accept:application/json" \n  --header "Content-Type:application/json"\n  --user "username":"password" \n  --data {
    "attributes": {
      "firewall_status": "Extranet",
      "discovery_source": "ServiceNows",
      "name": "Linux299",
      "used_for": "Pre-production",
      "container": "Box"
    },
    "source": "ServiceNow"
  }

{
  "result": {
    "outbound_relations": [
      {
        "sys_id": "403ff2641b425010593876a61a4bcb4b",
        "type": {
          ...
```
"display_value": "Depends on::Used by",
"link":
"https://demonightlyus.service-now.com/api/now/table/cmdb_rel_type/1a9cb166f15711100a92eb60da2bce5c5",
"value": "1a9cb166f15711100a92eb60da2bce5c5"
"target": {
"display_value": "PS ORA01",
"link":
"https://demonightlyus.service-now.com/api/now/cmdb/instance/cmdb_ci/3a307c930a0a0bb400353965d0b8861f",
"value": "3a307c930a0a0bb400353965d0b8861f"
}
"sys_id": "443ff2641b425010593876a61a4bcb4c",
"type": {
"display_value": "Exchanges data with::Exchanges data with",
"link":
"https://demonightlyus.service-now.com/api/now/table/cmdb_rel_type/607ad1b2c0a8010e01941856b365af90",
"value": "607ad1b2c0a8010e01941856b365af90"
"target": {
"display_value": "PS ORA01",
"link":
"https://demonightlyus.service-now.com/api/now/cmdb/instance/cmdb_ci/3a307c930a0a0bb400353965d0b8861f",
"value": "3a307c930a0a0bb400353965d0b8861f"
}
"attributes": {
"firewall_status": "Extranet",
"os_address_width": "",
"attested_date": "",
"operational_status": "1",
"os_service_pack": "",
"cpu_core_thread": "",
"cpu_manufacturer": "",
"sys_updated_on": "2020-07-13 20:59:40",
"discovery_source": "ServiceNow",
"first_discovered": "2020-07-13 20:27:28",
"due_in": ""}
"asset": {
    "display_value": "Unknown",
    "link": "https://demonightlyus.service-now.com/api/now/table/alm_asset/21cfba641b425010593876a61a4bcb52",
    "value": "21cfba641b425010593876a61a4bcb52",
    "form_factor": "",
    "cpu_core_count": "",
    "skip_sync": "false",
    "attestation_score": "",
    "sys_updated_by": "dora.gray",
    "sys_created_on": "2020-07-13 20:27:28",
    "sys_domain": {
        "display_value": "global",
        "link": "https://demonightlyus.service-now.com/api/now/table/sys_user_group/global",
        "value": "global"
    },
    "cpu_type": "",
    "install_date": "",
    "asset_tag": "",
    "dr_backup": "",
    "hardware_substatus": "",
    "fqdn": "",
    "change_control": "",
    "internet_facing": "true",
    "delivery_date": "",
    "hardware_status": "installed",
    "install_status": "1",
    "supported_by": "",
    "name": "Linux299",
    "subcategory": "Computer",
    "default_gateway": "",
    "chassis_type": "",
    "virtual": "false",
    "assignment_group": "",
    "managed_by_group": "",
    "sys_id": "0c3ff2641b425010593876a61a4bcb39",
    "po_number": "",
    "checked_in": "",
    "sys_class_path": "/!!/!2/!(/!!/!0",
    "mac_address": "",
    "company": "",
    "sys_domain": "global",
    "value": "global"
}
"
"justification": "",
"department": "",
"cost": "",
"comments": "",
"os": "",
"sys_mod_count": "3",
"monitor": "false",
"model_id": {
   "display_value": "Unknown",
   "link": "https://demonightlyus.service-now.com/api/now/table/cmdb_model/0ce0a2681b4e1010593876a61a4bcba6",
   "value": "0ce0a2681b4e1010593876a61a4bcba6"
},
"ip_address": "",
"duplicate_of": "",
"sys_tags": "",
"cost_cc": "USD",
"order_date": "",
"schedule": "",
"environment": "",
"due": "",
"attested": "false",
"location": "",
"category": "Hardware",
"fault_count": "0",
"host_name": "",
"lease_id": ""
},
"inbound_relations": [ 
{
   "sys_id": "c03ff2641b425010593876a61a4bcb49",
   "type": { 
      "display_value": "Depends on::Used by",
      "link": "https://demonightlyus.service-now.com/api/now/table/cmdb_rel_type/1a9cb166f1571100a92eb60da2bce5c5",
   "value": "1a9cb166f1571100a92eb60da2bce5c5"
},
   "target": { 
      "display_value": "PS Apache01",
      "link": "https://demonightlyus.service-now.com/api/now/cmdb/instance/cmdb_ci/3a27d4370a0a0bb4006316812bf45439",
}];
CMDB Instance - POST /now/cmdb/instance/{classname}/{sys_id}/relation

Adds an inbound and/or outbound relation to the specified configuration item (CI). The specified CI is the parent.

URL format

Versioned URL: /api/now/v1/cmdb/instance/{className}/{sys_id}/relation

Default URL: /api/now/cmdb/instance/{className}/{sys_id}/relation

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>className</td>
<td>CMDB class name. This is the name of the table that contains the desired CI records, such as cmdb_ci_linux_server or cmdb_ci_apache_web_server.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the CI record to update.</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inbound_relations</td>
<td>List of objects that define the inbound relations to associate with the CI.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>inbound_relations</strong></td>
<td>Required if <code>inbound_relations</code> is specified. Sys_id of the target inbound relation to associate with the specified CI. Located in the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td><strong>inbound_relations.target</strong></td>
<td>Required if <code>inbound_relations</code> is specified. Sys_id of the target inbound relation. Located in the CI Relationship Type [cmdb_rel_type] table. Data type: String</td>
</tr>
<tr>
<td><strong>inbound_relations.type</strong></td>
<td>Required if <code>inbound_relations</code> is specified. Sys_id of the type of inbound relationship. Located in the CI Relationship Type [cmdb_rel_type] table. Data type: String</td>
</tr>
<tr>
<td><strong>outbound_relations</strong></td>
<td>List of objects that define the outbound relations to associate with the CI. Data type: Array</td>
</tr>
<tr>
<td><strong>outbound_relations.target</strong></td>
<td>Required if <code>outbound_relations</code> is specified. Sys_id of the target outbound relation. Located in the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td><strong>outbound_relations.type</strong></td>
<td>Required if <code>outbound_relations</code> is specified. Sys_id of the type of outbound relationship. Located in the CI Relationship Type [cmdb_rel_type] table. Data type: String</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>Required. Entity that updated the information. This must be one of the choice values specified in the discovery_source field in the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

#### Response parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Data attributes currently in the CI record. The available attributes depend on the specified CMDB class. You can locate the available attributes in the associated CMDB table which typically begins with &quot;cmdb_ci&quot;, such as cmdb_ci_linux_server or cmdb_ci_mfp_printer. Data type: Object</td>
</tr>
<tr>
<td>error</td>
<td>Information on any errors encountered while processing the endpoint request. Data type: Object</td>
</tr>
<tr>
<td>error.detail</td>
<td>Additional information about the error.</td>
</tr>
</tbody>
</table>

```json
"error": {
   "detail": "String",
   "message": "String"
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error.message</td>
<td>Message that identifies the error. Data type: String</td>
</tr>
<tr>
<td>inbound_relations</td>
<td>List of objects that define the inbound relations associated with the CI. Data type: Array</td>
</tr>
<tr>
<td>inbound_relations.sys_id</td>
<td>Sys_id of the inbound CI relationship. Located in the CI Relationship [cmdb_rel_ci] table. Data type: String</td>
</tr>
<tr>
<td>inbound_relations.target</td>
<td>Information that describes the target inbound relation. Data type: Object</td>
</tr>
<tr>
<td>inbound_relations.target.display_value</td>
<td>Name of the inbound relation to display in a user interface. Data type: String</td>
</tr>
<tr>
<td>inbound_relations.target.link</td>
<td>CMDB REST endpoint URL to use to retrieve the inbound relation from the Configuration Item [cmdb_ci] table.</td>
</tr>
</tbody>
</table>
## Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>inbound_relations.target.value</code></td>
<td>Sys_id of the inbound CI. Located in the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td><strong>Data type:</strong> String</td>
<td></td>
</tr>
<tr>
<td><code>inbound_relations.type</code></td>
<td>Information about the type of inbound relationship.</td>
</tr>
<tr>
<td><strong>Data type:</strong> Object</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;display_value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;link&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td><code>inbound_relations.type.display_value</code></td>
<td>Name of the inbound type to display in a user interface.</td>
</tr>
<tr>
<td><strong>Data type:</strong> String</td>
<td></td>
</tr>
<tr>
<td><code>inbound_relations.type.link</code></td>
<td>Table REST endpoint URL to use to retrieve the inbound relation type from the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td><strong>Data type:</strong> String</td>
<td></td>
</tr>
<tr>
<td><code>inbound_relations.type.value</code></td>
<td>Sys_id of the type of inbound relationship. Located in the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td><strong>Data type:</strong> String</td>
<td></td>
</tr>
<tr>
<td><code>outbound_relations</code></td>
<td>List of objects that define the outbound relations associated with the CI.</td>
</tr>
<tr>
<td><strong>Data type:</strong> Array</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;outbound_relations&quot;: [</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
</tbody>
</table>
### Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>outbound_relations.sys_id</td>
<td>Sys_id of the outbound CI relationship. Located in the CI Relationship [cmdb_rel_ci] table. Data type: String</td>
</tr>
<tr>
<td>outbound_relations.target</td>
<td>Required. Information about the target outbound relation. Data type: Object</td>
</tr>
<tr>
<td>outbound_relations.target.display_value</td>
<td>Name of the outbound relation to display in a user interface. Data type: String</td>
</tr>
<tr>
<td>outbound_relations.target.link</td>
<td>CMDB REST endpoint URL to use to retrieve the outbound relation from the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>outbound_relations.target.value</td>
<td>Sys_id of the outbound CI. Located in the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>outbound_relations.type</td>
<td>Information about the type of outbound relationship. Data type: Object</td>
</tr>
</tbody>
</table>
Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>outbound_relations.type.display_value</td>
<td>Name of the outbound type to display in a user interface. Data type: String</td>
</tr>
<tr>
<td>outbound_relations.type.link</td>
<td>Table REST endpoint URL to use to retrieve the outbound relation from the CI Relationship Type [cmdb_rel_type] table. Data type: String</td>
</tr>
<tr>
<td>outbound_relations.type.value</td>
<td>Sys_id of the type of outbound relationship. Located in the CI Relationship Type [cmdb_rel_type] table. Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>Only appears if an error is encountered. Status of the endpoint processing. Possible values: • failure. Data type: String</td>
</tr>
</tbody>
</table>

Example: Sample cURL request

```bash
curl -H "https://instance.servicenow.com/api/now/cmdb/instance/cmdb_ci_linux_server/3a290cc60a0a0bab400000b4b386af1cf/relation" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --user "username":"password" \
  --data "
  "outbound_relations": [
  
```
"life_cycle_stage": "",
"short_description": "",
"cd_speed": "",
"floppy": "",
"managed_by": "",
"os_domain": "",
"last_discovered": "2020-07-13 21:24:05",
"can_print": "false",
"sys_class_name": "cmdb_ci_linux_server",
"manufacturer": "",
"cpu_count": "",
"vendor": "",
"life_cycle_stage_status": "",
"model_number": "",
"assigned_to": "",
"start_date": "",
"os_version": "",
"serial_number": "",
"cd_rom": "false",
"support_group": "",
"unverified": "false",
"correlation_id": "",
"attributes": "",
"asset": {
  "display_value": "Unknown",
  "link": "https://instance.servicenow.com/api/now/table/alm_asset/21cfba641b425010593876a61a4bcb52",
  "value": "21cfba641b425010593876a61a4bcb52"
},
"form_factor": "",
"cpu_core_count": "",
"skip_sync": "false",
"attestation_score": "",
"sys_updated_by": "dora.gray",
"sys_created_on": "2020-07-13 20:27:28",
"sys_domain": {
  "display_value": "global",
  "link": "https://instance.servicenow.com/api/now/table/sys_user_group/global",
  "value": "global"
},
"cpu_type": "",
"install_date": ""
"asset_tag": "",
"dr_backup": "",
"hardware_substatus": "",
"fqdn": "",
"change_control": "",
"internet_facing": "true",
"delivery_date": "",
"hardware_status": "installed",
"install_status": "1",
"supported_by": "",
"name": "Linux299",
"subcategory": "Computer",
"default_gateway": "",
"chassis_type": "",
"virtual": "false",
"assignment_group": "",
"managed_by_group": "",
"sys_id": "0c3ff2641b425010593876a61a4bcb39",
"pc_number": "",
"checked_in": "",
"sys_class_path": "/!/!/!/!/!/!/!/0",
"mac_address": "",
"company": "",
"justification": "",
"department": "",
"cost": "",
"comments": "",
"os": "",
"sys_mod_count": "4",
"monitor": "false",
"model_id": {
    "display_value": "Unknown",
    "link": 
    "https://instance.servicenow.com/api/now/table/cmdb_model/0ce0a2681b4e1010593876a61a4bcba6",
    "value": "0ce0a2681b4e1010593876a61a4bcba6"
},
"ip_address": "",
"duplicate_of": "",
"sys_tags": "",
"cost_cc": "USD",
"order_date": "",
"schedule": "",
"environment": ""}
CMDB Instance - PATCH /now/cmdb/instance/{classname}/{sys_id}

Replaces the attributes in the specified configuration item (CI) record with the specified name-value pairs. This endpoint does not create, update, or delete the associated relations.

URL format

Versioned URL: /api/now/v1/cmdb/instance/{className}/{sys_id}
Default URL: /api/now/cmdb/instance/{className}/{sys_id}
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>className</td>
<td>CMDB class name. This is the name of the table that contains the desired CI records, such as cmdb_ci_linux_server or cmdb_ci_apache_web_server. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the CI record to update. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Data attributes to replace in the CI record. The available attributes depend on the specified CMDB class. You can locate the available attributes in the associated CMDB table which typically begins with &quot;cmdb_ci&quot;, such as cmdb_ci_linux_server or cmdb_ci_mfp_printer. Data type: Object</td>
</tr>
<tr>
<td>source</td>
<td>Required. Entity that updated the information. This must be one of the choice values specified in the discovery_source field in the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
## Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

## Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

## Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
# Response body parameters (JSON or XML)

## Response parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Data attributes currently in the CI record. The available attributes depend on the specified CMDB class. You can locate the available attributes in the associated CMDB table which typically begins with &quot;cmdb_ci&quot;, such as cmdb_ci_linux_server or cmdb_ci_mfp_printer. Data type: Object</td>
</tr>
<tr>
<td>error</td>
<td>Information on any errors encountered while processing the endpoint request. Data type: Object</td>
</tr>
<tr>
<td>error.detail</td>
<td>Additional information about the error. Data type: String</td>
</tr>
<tr>
<td>error.message</td>
<td>Message that identifies the error. Data type: String</td>
</tr>
<tr>
<td>inbound_relations</td>
<td>List of objects that define the inbound relations associated with the CI. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"error": {  
  "detail": "String",  
  "message": "String"
}
```

```json
"inbound_relations": [  
  {  
    "sys_id": "String",  
    "target": {Object},  
    "type": {Object}
  }
]```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inbound_relations.sys_id</td>
<td>Sys_id of the inbound CI relationship. Located in the CI Relationship [cmdb_rel_ci] table. Data type: String</td>
</tr>
<tr>
<td>inbound_relations.target</td>
<td>Information that describes the target inbound relation. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;target&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;display_value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;link&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>inbound_relations.target.display_value</td>
<td>Name of the inbound relation to display in a user interface. Data type: String</td>
</tr>
<tr>
<td>inbound_relations.target.link</td>
<td>CMDB REST endpoint URL to use to retrieve the inbound relation from the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>inbound_relations.target.value</td>
<td>Sys_id of the inbound CI. Located in the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>inbound_relations.type</td>
<td>Information about the type of inbound relationship. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;display_value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;link&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Response parameters (continued)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>inbound_relations.type.display_value</strong></td>
<td>Name of the inbound type to display in a user interface.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>inbound_relations.type.link</strong></td>
<td>Table REST endpoint URL to use to retrieve the inbound relation type from the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>inbound_relations.type.value</strong></td>
<td>Sys_id of the type of inbound relationship. Located in the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>outbound_relations</strong></td>
<td>List of objects that define the outbound relations associated with the CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>outbound_relations.sys_id</strong></td>
<td>Sys_id of the outbound CI relationship. Located in the CI Relationship [cmdb_rel_ci] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>outbound_relations.target</strong></td>
<td>Required. Information about the target outbound relation.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
</tbody>
</table>
### Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;target&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;display_value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;link&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>outdelta_relations.target.display_value</td>
<td>Name of the outbound relation to display in a user interface.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>outdelta_relations.target.link</td>
<td>CMDB REST endpoint URL to use to retrieve the outbound relation from the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>outdelta_relations.target.value</td>
<td>Sys_id of the outbound CI. Located in the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>outdelta_relations.type</td>
<td>Information about the type of outbound relationship.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;display_value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;link&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>outdelta_relations.type.display_value</td>
<td>Name of the outbound type to display in a user interface.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>outdelta_relations.type.link</td>
<td>Table REST endpoint URL to use to retrieve the outbound relation from the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Response parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>outbound_relations.type.value</td>
<td>Sys_id of the type of outbound relationship. Located in the CI Relationship Type [cmdb_rel_type] table. Data type: String</td>
</tr>
</tbody>
</table>
| status                                    | Only appears if an error is encountered. Status of the endpoint processing. Possible values:  
  • failure 
  Data type: String |

Example: Sample cURL request

```bash
curl  
  "https://instance.servicenow.com/api/now/cmdb/instance/cmdb_class_sw/70bc4e11c0a8210d01f074cbe6bd73b4" \  
  --request PATCH \  
  --header "Accept:application/json" \  
  --header "Content-Type:application/json"\  
  --user "username":"password"\  
  --data {  
    "attributes": {  
      "firewall_status": "Active",  
      "name": "Linux299",  
      "used_for": "Production"  
    },  
    "source": "ServiceNow"  
  }

{  
  "result": {  
    "outbound_relations": [  
      {  
        "sys_id": "403ff2641b425010593876a61a4bcb4b",  
        "type": {  
          "display_value": "Depends on::Used by",  
```
"link": "https://instance.servicenow.com/api/now/table/cmdb_rel_type/1a9cb166f1571100a92eb60da2bce5c5",
   "value": "1a9cb166f1571100a92eb60da2bce5c5"
},
"target": {
   "display_value": "PS ORA01",
   "link": "https://instance.servicenow.com/api/now/cmdb/instance/cmdb_ci/3a307c930a0a0bb400353965d0b8861f",
   "value": "3a307c930a0a0bb400353965d0b8861f"
}
},
{
   "sys_id": "443ff2641b425010593876a61a4bcb4c",
   "type": {
      "display_value": "Exchanges data with::Exchanges data with",
      "link": "https://instance.servicenow.com/api/now/table/cmdb_rel_type/607ad1b2c0a8010e01941856b365af90",
      "value": "607ad1b2c0a8010e01941856b365af90"
   },
   "target": {
      "display_value": "PS ORA01",
      "link": "https://instance.servicenow.com/api/now/cmdb/instance/cmdb_ci/3a307c930a0a0bb400353965d0b8861f",
      "value": "3a307c930a0a0bb400353965d0b8861f"
   }
},
"attributes": {
   "firewall_status": "Active",
   "os_address_width": 
   "", 
   "attested_date": 
   
   
   
   
   "operational_status": "1",
   "os_service_pack": 
   
   
   
   
   
   "cpu_core_thread": 
   
   
   
   "cpu_manufacturer": 
   "",
   "sys_updated_on": "2020-07-13 21:24:05",
   "discovery_source": "ServiceNow",
   "first_discovered": "2020-07-13 20:27:28",
   "due_in": 
   "",
   "used_for": "Production"}
"display_value": "Unknown",
"link": "https://instance.servicenow.com/api/now/table/alm_asset/21cfba641b425010593876a61a4bcb52",
"value": "21cfba641b425010593876a61a4bcb52",
"form_factor": "",
"cpu_core_count": "",
"skip_sync": "false",
"attestation_score": "",
"sys_updated_by": "dora.gray",
"sys_created_on": "2020-07-13 20:27:28",
"sys_domain": {
  "display_value": "global",
  "link": "https://instance.servicenow.com/api/now/table/sys_user_group/global",
  "value": "global"
},
"cpu_type": "",
"install_date": "",
"asset_tag": "",
"dr_backup": "",
"hardware_substatus": "",
"fqdn": "",
"change_control": "",
"internet_facing": "true",
"delivery_date": "",
"hardware_status": "installed",
"install_status": "i",
"supported_by": "",
"name": "Linux299",
"subcategory": "Computer",
"default_gateway": "",
"chassis_type": "",
"virtual": "false",
"assignment_group": "",
"managed_by_group": "",
"sys_id": "0c3ff2641b425010593876a61a4bcb39",
"po_number": "",
"checked_in": "",
"sys_class_path": "/!!/!2/!/!!/!/!0",
"mac_address": "",
"company": "",
"justification": "",
"department": ""
CMDB Meta API

Use the CMDB Meta API to obtain meta data on a CMDB class.

This API requires that the user have the ITIL role.

**CMDB Meta - GET /now/cmdb/meta/{classname}**

Returns the meta data for the specified CMDB class.

**URL format**

Versioned URL: /api/now/{api_version}/cmdb/meta/{className}

Default URL: /api/now/cmdb/meta/{className}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| api_version | Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.  
Data type: String |
| className | CMDB class name. This is the name of the table that contains the desired CI records, such as cmdb_ci_linux_server or cmdb_ci_apache_web_server.  
Data type: String |

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml.</td>
</tr>
<tr>
<td></td>
<td>Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>Available fields in the specified class table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>children</td>
<td>List of classes extended from the specified class.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>icon</td>
<td>Class icon sys_id.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>icon_url</td>
<td>Class icon URL.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>identification_rules</td>
<td>Attributes associated with the configuration item identification rules for</td>
</tr>
<tr>
<td></td>
<td>the specified class.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>is_extendable</td>
<td>Flag that indicates whether the class can be extended to create other classes. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Class can be extended.</td>
</tr>
<tr>
<td></td>
<td>• false: Class cannot be extended.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>label</td>
<td>Specified class display name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>name</td>
<td>Table/class name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>parent</td>
<td>Parent class.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>relationship_rules</td>
<td>Relationships between the specified class and other classes in the CMDB.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>
Example: Sample cURL request

curl "https://instance.servicenow/api/now/cmdb/meta/cmdb_ci_computer" \
   --request GET \
   --header "Accept:application/json" \
   --user 'username':'password'

{
   "result": {
   "icon_url": "images/app.ngbsm/computer.svg",
   "is_extendable": true,
   "parent": "cmdb_ci_hardware",
   "children": [
   "cmdb_ci_ucs_blade",
   "cmdb_ci_pc_hardware",
   "cmdb_ci_ucs_rack_unit",
   "cmdb_ci_mainframe_hardware",
   "cmdb_ci_server",
   "cmdb_ci_storage_switch"
   ],
   "name": "cmdb_ci_computer",
   "icon": "c6442dd69fb00200eb3919eb552e7012",
   "attributes": [
   {
   "is_inherited": "false",
   "is_mandatory": "false",
   "is_read_only": "false",
   "default_value": null,
   "label": "OS Address Width (bits)",
   "type": "integer",
   "element": "os_address_width",
   "max_length": "40",
   "is_display": "false"
   },
   {
   "is_inherited": "true",
   "is_mandatory": "false",
   "is_read_only": "true",
   "default_value": "false",
   "label": "Skip sync",
   "type": "boolean",
   "element": "skip_sync",
   "max_length": "40",
   "is_display": "false"
   }
   ]
}
"is_inherited": "true",
"is_mandatory": "false",
"is_read_only": "false",
"default_value": null,
"label": "DNS Domain",
"type": "string",
"element": "dns_domain",
"max_length": "255",
"is_display": "false"
},
{
"is_inherited": "true",
"is_mandatory": "false",
"is_read_only": "false",
"default_value": null,
"label": "Purchased",
"type": "glide_date",
"element": "purchase_date",
"max_length": "40",
"is_display": "false"
},
{
"is_inherited": "true",
"is_mandatory": "false",
"is_read_only": "false",
"default_value": null,
"label": "Lease contract",
"type": "string",
"element": "lease_id",
"max_length": "40",
"is_display": "false"
}
],
"relationship_rules": [
{
"parent": "cmdb_ci_computer",
"relation_type": "cb5592603751200032ff8c00dfbe5d17",
"child": "dscy_route_next_hop"
},
{
"parent": "cmdb_ci_computer",
"relation_type": "cb5592603751200032ff8c00dfbe5d17",
"child": "dscy_route_next_hop"
}
]
"child": "dscy_router_interface"
],
{
"parent": "cmdb_ci_computer",
"relation_type": "cb5592603751200032ff8c00dfbe5d17",
"child": "dscy_route_interface"
],
{
"parent": "cmdb_ci_computer",
"relation_type": "55c95bf6c0a8010e0118ec7056ebc54d",
"child": "cmdb_ci_storage_pool"
],
{
"parent": "cmdb_ci_computer",
"relation_type": "55c95bf6c0a8010e0118ec7056ebc54d",
"child": "cmdb_ci_disk_partition"
],
{
"parent": "cmdb_ci_computer",
"relation_type": "55c95bf6c0a8010e0118ec7056ebc54d",
"child": "cmdb_ci_storage_volume"
],
{
"parent": "cmdb_ci_computer",
"relation_type": "55c95bf6c0a8010e0118ec7056ebc54d",
"child": "cmdb_ci_storage_device"
}
],
"label": "Computer",
"identification_rules": {
"related_rules": [
{
"condition": "",
"exact_count_match": false,
"referenced_field": "installed_on",
"active": true,
"attributes": "name",
"allow_fallback": false,
"table": "cmdb_print_queue_instance",
"order": 100,
"allow_null_attribute": false
}]
},
"applies_to": "cmdb_ci_hardware"
"identifiers": [
{
    "condition": "valid=true^absent=false"EQ",
    "exact_count_match": true,
    "referenced_field": "cmdb_ci",
    "active": true,
    "attributes": "serial_number,serial_number_type",
    "allow_fallback": false,
    "table": "cmdb_serial_number",
    "order": 100,
    "allow_null_attribute": false
},
{
    "condition": null,
    "exact_count_match": false,
    "referenced_field": null,
    "active": true,
    "attributes": "serial_number",
    "allow_fallback": false,
    "table": null,
    "order": 200,
    "allow_null_attribute": false
},
{
    "condition": null,
    "exact_count_match": false,
    "referenced_field": null,
    "active": true,
    "attributes": "name",
    "allow_fallback": false,
    "table": null,
    "order": 300,
    "allow_null_attribute": false
},
{
    "condition": "install_status!=100"EQ",
    "exact_count_match": true,
    "referenced_field": "cmdb_ci",
    "active": true,
    "attributes": "ip_address,mac_address",
    "allow_fallback": false,
    "table": "cmdb_ci_network_adapter",
    "order": 400,
    "allow_null_attribute": false
}]}
Consumer API

The Consumer REST API enables you to retrieve and update Customer Service Management (CSM) consumer records.

In addition, you can generate new social media profile records when creating a consumer.

The Consumer API requires the Customer Service plugin (com.sn_customerservice) and is provided within the now namespace.

Users require the csm_ws_integration role for full API access.

Consumer - GET /now/consumer

Retrieves a specified set of Customer Service Management (CSM) consumer records.

URL format

Versioned URL: /api/now/{api_version}/consumer
Default URL: /api/now/consumer

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
</tbody>
</table>

Data type: String
## Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| sysparm_limit      | Maximum number of records to return. For requests that exceed this number of records, use the `sysparm_offset` parameter to paginate record retrieval.  
Data type: Number  
Default: 10                                                                                     |
| sysparm_offset     | Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks.  
For example, the first time you call this endpoint, `sysparm_offset` is set to "0". To simply page through all available records, use `sysparm_offset=sysparm_offset+sysparm_limit`, until you reach the end of all records.  
Do not pass a negative number in the `sysparm_offset` parameter.  
Data type: Number  
Default: 0                                                                                     |
| sysparm_query      | Encoded query used to filter the result set.  
For example:  
```plaintext
sysparm_query=caller_id=javascript:gs.getUserID()^active=true
```

The encoded query supports `order by`. To sort responses based on certain fields, use the `ORDERBY` and `ORDERBYDESC` clauses in `sysparm_query`. For example,  
```plaintext
sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory
```
filters all active records and orders the results in ascending order by number first, and then in descending order by category.  

If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property `glide.invalid_query.returns_no_rows`. Set this property to true to return no rows on an invalid query. |
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The glide.invalid_query returns_no_rows property controls the behavior of all queries across the instance, such as in lists, scripts (GlideRecord.query()), and web service APIs. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

The endpoint may return the following JSON or XML elements in the response body. In addition to the list of elements defined below (which define the elements in a base system), the endpoint also returns any custom fields added to the Consumer [csm_consumer] table. For additional information on these elements, refer to your specific table definition [System Definition > Tables].

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| active              | Flag that indicates whether the consumer is active. Possible values:  
|                     |   • true: Consumer active  
|                     |   • false: Consumer de-activated  
|                     | Data type: Boolean  
|                     | Default: true |
| business_phone      | Business phone number of the consumer.  
|                     | Data type: String  
|                     | Maximum length: 40 |
| city                | City in which the consumer resides.  
|                     | Data type: String  
|                     | Maximum length: 100 |
| country             | Country in which the consumer resides.  
<p>|                     | Data type: String |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>date_format</td>
<td>Format in which to display dates. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• dd-mm-yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd/mm/yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd.mm.yyyy</td>
</tr>
<tr>
<td></td>
<td>• mm-dd-yyyy</td>
</tr>
<tr>
<td></td>
<td>• yyyy-mm-dd</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system date format)</td>
</tr>
<tr>
<td>email</td>
<td>Email address of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>fax</td>
<td>Fax number of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>first_name</td>
<td>Consumer first name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>gender</td>
<td>Gender of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>home_phone</td>
<td>Home phone number of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>household</td>
<td>Sys_id of the record that describes the household characteristics. Located in the Household [csm_household] table.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>last_name</td>
<td>Consumer last name. Data type: String, Maximum length: 50</td>
</tr>
<tr>
<td>middle_name</td>
<td>Consumer middle name. Data type: String, Maximum length: 50</td>
</tr>
<tr>
<td>mobile_phone</td>
<td>Consumer mobile phone number. Data type: String, Maximum length: 40</td>
</tr>
<tr>
<td>name</td>
<td>Consumer full name; first_name+middle_name+last_name. Data type: String,</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 152</td>
</tr>
<tr>
<td>notes</td>
<td>Notes on consumer. Data type: String, Maximum length: 4,000</td>
</tr>
<tr>
<td>notification</td>
<td>Indicates whether the consumer should receive notifications. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Disabled</td>
</tr>
<tr>
<td></td>
<td>• 2: Enabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Integer, Maximum length: 40, Default: 2</td>
</tr>
<tr>
<td>number</td>
<td>Unique number associated with the consumer. Data type: String, Maximum</td>
</tr>
<tr>
<td></td>
<td>length: 40</td>
</tr>
<tr>
<td>photo</td>
<td>Photo of the consumer.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| preferred_language | Consumer primary language. Data type: String  
Max length: 3 |
| prefix            | Consumer name prefix such as, Dr., Mr., Mrs., or Ms. Data type: String  
Max length: 40 |
| primary           | Flag that indicates whether this is the primary consumer. Possible values:  
• true: Primary consumer  
• false: Not primary consumer |
| state             | State in which the consumer resides. Data type: String  
Max length: 100 |
| street            | Consumer street address. Data type: String  
Max length: 255 |
| suffix            | Consumer name suffix such as Jr., Sr., or II. Data type: String |
| sys_created_by    | User that created the consumer record. Data type: String  
Max length: 40 |
| sys_created_on    | Date and time the consumer record was originally created. Data type: String |
| sys_domain        | ServiceNow domain in which the consumer information resides. |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>Unique identifier for the consumer.</td>
</tr>
<tr>
<td>sys_mod_count</td>
<td>Number of times that the associated consumer information has been modified.</td>
</tr>
<tr>
<td>sys_tags</td>
<td>System tags.</td>
</tr>
<tr>
<td>sys_updated_by</td>
<td>User that last updated the consumer information.</td>
</tr>
<tr>
<td>sys_updated_on</td>
<td>Date and time when the consumer information was last updated.</td>
</tr>
<tr>
<td>time_format</td>
<td>Format in which to display time.</td>
</tr>
<tr>
<td>time_zone</td>
<td>Consumer time zone, such as Canada/Central or US/Eastern.</td>
</tr>
<tr>
<td>title</td>
<td>Consumer business title such as Manager, Software Developer, or Contractor.</td>
</tr>
</tbody>
</table>

- **Data type**: String
- **Maximum length**: 40
- **Default**: blank (system time format)
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>Sys_id of the consumer user. Located in the Consumer User [csm_consumer_user] table. Data type: String</td>
</tr>
<tr>
<td>zip</td>
<td>Consumer zip code. Data type: String Maximum length: 40</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
  "https://instance.servicenow.com/api/now/consumer?sysparm_query=account=86837a386f0331003b3c498f5d3ee4ca&sysparm_limit=2&sysparm_offset=2>;rel="next" \
--request GET \
--header "Accept:application/json" \
--user "username":"password"

```

```json
{
  "result": [
    {
      "country": "USA",
      "notes": "",
      "gender": "Male",
      "city": "Plano",
      "prefix": "",
      "sys_updated_on": "2016-08-12 00:19:12",
      "suffix": "",
      "title": "",
      "number": "CSMR0000004",
      "notification": "2",
      "sys_id": "01d8403fdbl1b1200b6075200cf961941",
      "business_phone": "",
      "sys_updated_by": "prithvi",
      "mobile_phone": "",
      "street": "6900 Dallas Pkwy",
      "sys_created_on": "2016-06-16 19:20:13",
      "sys_domain": "global",
      "state": "TX",
      "fax": "",
      "first_name": "Harding",
      "email": "harding.asher@mailinator.com",
    }
  ]
}
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
"preferred_language": "",
"sys_created_by": "guest",
"zip": "75024",
"home_phone": "",
"time_format": "",
"sys_mod_count": "8",
"last_name": "Asher",
"photo": "",
"active": "true",
"middle_name": "",
"time_zone": "",
"sys_tags": "",
"name": "Harding Asher",
"household": "",
"date_format": "",
"user": "c3d35d82c37122005871d44d81d3ae91",
"primary": "false"
},
{
"country": "USA",
"notes": "",
"gender": "Male",
"city": "San Francisco",
"prefix": "",
"sys_updated_on": "2016-08-12 00:20:27",
"suffix": "",
"title": "",
"number": "CSMR0000002",
"notification": "2",
"sys_id": "a0488cfbdb1b1200b6075200cf9619db",
"business_phone": "",
"sys_updated_by": "prithvi",
"mobile_phone": "",
"street": "144 2nd St",
"sys_created_on": "2016-06-16 19:17:44",
"sys_domain": "global",
"state": "CA",
"fax": "",
"first_name": "Sam",
"email": "sam.collins@mailinator.com",
"preferred_language": "",
"sys_created_by": "guest",
"zip": "94105",
"home_phone": "",

6081

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


"time_format": "",
"sys_mod_count": "13",
"last_name": "Collins",
"photo": "",
"active": "true",
"middle_name": "",
"time_zone": "",
"sys_tags": "",
"name": "Sam Collins",
"household": "",
"date_format": "",
"user": "64488cfdbeb1200b6075200cf9619db",
"primary": "false"
}
}

Example: Python request

#Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/consumer?sysparm_limit=1'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept':"application/xml"}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
<xml version="1.0" encoding="UTF-8">
<response>
  <result>
    <country>USA</country>
    <gender>Female</gender>
    <city>Atlanta</city>
    <number>CSMR0000003</number>
    <street>1201 Peachtree St NE</street>
    <first_name>Dee</first_name>
    <last_name>Sam</last_name>
    <email>dee.sam@mailinator.com</email>
    <active>true</active>
    <primary>false</primary>
  </result>
</response>
Consumer - GET /now/consumer/{id}
Retrieves the specified Customer Service Management (CSM) consumer record.

**URL format**
Versioned URL: /api/now/{api_version}/consumer/{id}
Default URL: /api/now/consumer/{id}

**Supported request parameters**

| Path parameters |
|-----------------|-------------------------------------------------------------|
| **Name**       | **Description**                                            |
| api_version    | Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String |
| id             | Sys_id of the consumer record to return. Located in the Consumer [csm_consumer] table. Data type: String |

| Query parameters |
|------------------|---------------------------------------------------------------|
| **Name**         | **Description**                                              |
| None             |                                                               |

<table>
<thead>
<tr>
<th>Request body parameters (XML or JSON)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

**Headers**
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
</tbody>
</table>
| 404         | Indicates that the request is invalid. Could be due to one of the following reasons:  
|             | • Requested case does not exist.                                           |
|             | • User does not have access to the consumer record.                        |
| 500         | Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error. |

Response body parameters (JSON or XML)

The endpoint may return the following JSON or XML elements in the response body. In addition to the list of elements defined below (which define the elements in a base system), the endpoint also returns any custom fields added to the Consumer [csm_consumer] table. For additional information on these elements, refer to your specific table definition [System Definition > Tables].
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>Flag that indicates whether the consumer is active. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Consumer active</td>
</tr>
<tr>
<td></td>
<td>• false: Consumer de-activated</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td>business_phone</td>
<td>Business phone number of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>city</td>
<td>City in which the consumer resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>country</td>
<td>Country in which the consumer resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>date_format</td>
<td>Format in which to display dates.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• dd-mm-yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd/mm/yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd.mm.yyyy</td>
</tr>
<tr>
<td></td>
<td>• mm-dd-yyyy</td>
</tr>
<tr>
<td></td>
<td>• yyyy-mm-dd</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system date format)</td>
</tr>
<tr>
<td>email</td>
<td>Email address of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>fax</td>
<td>Fax number of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>first_name</td>
<td>Consumer first name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>gender</td>
<td>Gender of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>home_phone</td>
<td>Home phone number of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>household</td>
<td>Sys_id of the record that describes the household characteristics. Located in the Household [csm_household] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>last_name</td>
<td>Consumer last name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>middle_name</td>
<td>Consumer middle name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>mobile_phone</td>
<td>Consumer mobile phone number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>name</td>
<td>Consumer full name; first_name+middle_name +last_name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 152</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>notes</td>
<td>Notes on consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 4,000</td>
</tr>
<tr>
<td>notification</td>
<td>Indicates whether the consumer should receive notifications.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Disabled</td>
</tr>
<tr>
<td></td>
<td>• 2: Enabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Integer</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: 2</td>
</tr>
<tr>
<td>number</td>
<td>Unique number associated with the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>photo</td>
<td>Photo of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: Image</td>
</tr>
<tr>
<td>preferred_language</td>
<td>Consumer primary language.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 3</td>
</tr>
<tr>
<td>prefix</td>
<td>Consumer name prefix such as, Dr., Mr., Mrs., or Ms.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>primary</td>
<td>Flag that indicates whether this is the primary consumer.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Primary consumer</td>
</tr>
<tr>
<td></td>
<td>• false: Not primary consumer</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>State in which the consumer resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>street</td>
<td>Consumer street address.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>suffix</td>
<td>Consumer name suffix such as Jr., Sr., or II.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_created_by</td>
<td>User that created the consumer record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sys_created_on</td>
<td>Date and time the consumer record was originally created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_domain</td>
<td>ServiceNow domain in which the consumer information resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Unique identifier for the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_mod_count</td>
<td>Number of times that the associated consumer information has been modified.</td>
</tr>
<tr>
<td></td>
<td>Data type: Integer</td>
</tr>
<tr>
<td>sys_updated_by</td>
<td>User that last updated the consumer information.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>sys_updated_on</td>
<td>Date and time when the consumer information was last updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>time_format</td>
<td>Format in which to display time.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>• hh.mm.ss a: hh.mm.ss (12 hour)</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss (12 hour)</td>
</tr>
<tr>
<td></td>
<td>• HH.mm.ss: hh.mm.ss (24 hour)</td>
</tr>
<tr>
<td></td>
<td>• HH:mm:ss: hh:mm:ss (24 hour)</td>
</tr>
<tr>
<td>time_zone</td>
<td>Consumer time zone, such as Canada/Central or US/Eastern.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system time format)</td>
</tr>
<tr>
<td>title</td>
<td>Consumer business title such as Manager, Software Developer, or Contractor.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 60</td>
</tr>
<tr>
<td>user</td>
<td>Sys_id of the consumer user. Located in the Consumer User [csm_consumer_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>zip</td>
<td>Consumer zip code.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl "https://instance.servicenow.com/api/now/consumer/01d8403fbb1b1200b6075200cf961941 "
    --request GET \
    --header "Accept:application/json" \
    --user "username":"password"
```

```json
{
    "result": {
        "country": "USA",
        "notes": ",",
        "gender": "Male",
```
<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>city</td>
<td>Plano</td>
</tr>
<tr>
<td>prefix</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>sys_updated_on</td>
<td>&quot;2016-08-12 00:19:12&quot;</td>
</tr>
<tr>
<td>suffix</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>title</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>number</td>
<td>&quot;CSMR0000004&quot;</td>
</tr>
<tr>
<td>notification</td>
<td>&quot;2&quot;</td>
</tr>
<tr>
<td>sys_id</td>
<td>&quot;01d8403fdb1b1200b6075200cf961941&quot;</td>
</tr>
<tr>
<td>business_phone</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>sys_updated_by</td>
<td>&quot;prithvi&quot;</td>
</tr>
<tr>
<td>mobile_phone</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>street</td>
<td>&quot;6900 Dallas Pkwy&quot;</td>
</tr>
<tr>
<td>sys_created_on</td>
<td>&quot;2016-06-16 19:20:13&quot;</td>
</tr>
<tr>
<td>sys_domain</td>
<td>&quot;global&quot;</td>
</tr>
<tr>
<td>state</td>
<td>&quot;TX&quot;</td>
</tr>
<tr>
<td>fax</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>first_name</td>
<td>&quot;Harding&quot;</td>
</tr>
<tr>
<td>email</td>
<td>&quot;<a href="mailto:harding.asher@mailinator.com">harding.asher@mailinator.com</a>&quot;</td>
</tr>
<tr>
<td>preferred_language</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>sys_created_by</td>
<td>&quot;guest&quot;</td>
</tr>
<tr>
<td>zip</td>
<td>&quot;75024&quot;</td>
</tr>
<tr>
<td>home_phone</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>time_format</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>sys_mod_count</td>
<td>&quot;8&quot;</td>
</tr>
<tr>
<td>last_name</td>
<td>&quot;Asher&quot;</td>
</tr>
<tr>
<td>photo</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>active</td>
<td>&quot;true&quot;</td>
</tr>
<tr>
<td>middle_name</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>time_zone</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>sys_tags</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>name</td>
<td>&quot;Harding Asher&quot;</td>
</tr>
<tr>
<td>household</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>date_format</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>user</td>
<td>&quot;c3d35d82c37122005871d44d81d3ae91&quot;</td>
</tr>
<tr>
<td>primary</td>
<td>&quot;false&quot;</td>
</tr>
</tbody>
</table>

**Example: Python request**

```python
import requests

# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/consumer/01d8403fdb1b1200b6075200cf961941'
```
# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept':"application/xml"}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers )

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <country>USA</country>
    <notes/>
    <gender>Male</gender>
    <city>Plano</city>
    <prefix/>
    <sys_updated_on>2016-08-12 00:19:12</sys_updated_on>
    <suffix/>
    <title/>
    <number>CSMR0000004</number>
    <notification>2</notification>
    <sys_id>01d8403fdd1b1200b6075200cf961941</sys_id>
    <business_phone/>
    <sys_updated_by>prithvi</sys_updated_by>
    <mobile_phone/>
    <street>6900 Dallas Pkwy</street>
    <sys_created_on>2016-06-16 19:20:13</sys_created_on>
    <sys_domain>global</sys_domain>
    <state>TX</state>
    <fax/>
    <first_name>Harding</first_name>
    <email>harding.asher@mailinator.com</email>
  </result>
</response>
Consumer - POST /now/consumer

Creates a new Customer Service Management (CSM) consumer.

In addition, you can create a social media profile for the consumer using this endpoint. To create the profile, you must specify the following parameters in the request body:

- social_channel
- social_handle
- social_handle_url

⚠️ Warning: This endpoint does not perform parameter validation as doing so can create excessive overhead. If a request parameter is misspelled, is not valid, or is not supported by the endpoint, it is ignored without warning.

URL format

Versioned URL: /api/now/{api_version}/consumer
Default URL: /api/now/consumer

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, <code>v1</code> or <code>v2</code>. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>Flag that indicates whether the consumer is active. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Consumer active</td>
</tr>
<tr>
<td></td>
<td>• false: Consumer de-activated</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td>business_phone</td>
<td>Business phone number of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>city</td>
<td>City in which the consumer resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>country</td>
<td>Country in which the consumer resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>date_format</td>
<td>Format in which to display dates.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• dd-mm-yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd/mm/yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd.mm.yyyy</td>
</tr>
<tr>
<td></td>
<td>• mm-dd-yyyy</td>
</tr>
<tr>
<td></td>
<td>• yyyy-mm-dd</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system date format)</td>
</tr>
<tr>
<td>email</td>
<td>Email address of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 100</td>
</tr>
<tr>
<td>fax</td>
<td>Fax number of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>first_name</td>
<td>Consumer first name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>gender</td>
<td>Gender of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>home_phone</td>
<td>Home phone number of the consumer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>household</td>
<td>Sys_id of the record that describes the household characteristics.</td>
</tr>
<tr>
<td></td>
<td>Located in the Household [csm_household] table.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>last_name</td>
<td>Consumer last name. Data type: String, Maximum length: 50</td>
</tr>
<tr>
<td>middle_name</td>
<td>Consumer middle name. Data type: String, Maximum length: 50</td>
</tr>
<tr>
<td>mobile_phone</td>
<td>Consumer mobile phone number. Data type: String, Maximum length: 40</td>
</tr>
<tr>
<td>name</td>
<td>Consumer full name; first_name+middle_name+last_name. Data type: String, Maximum length: 152</td>
</tr>
<tr>
<td>notes</td>
<td>Notes on consumer. Data type: String, Maximum length: 4,000</td>
</tr>
<tr>
<td>notification</td>
<td>Indicates whether the consumer should receive notifications. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Disabled</td>
</tr>
<tr>
<td></td>
<td>• 2: Enabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Integer, Maximum length: 40, Default: 2</td>
</tr>
<tr>
<td>photo</td>
<td>Photo of the consumer. Data type: Image</td>
</tr>
<tr>
<td>preferred_language</td>
<td>Consumer primary language.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>prefix</td>
<td>Consumer name prefix such as, Dr., Mr., Mrs., or Ms.</td>
</tr>
<tr>
<td>primary</td>
<td>Flag that indicates whether this is the primary consumer. Possible values:</td>
</tr>
<tr>
<td>social_channel</td>
<td>Social media channel to which the consumer is associated such as Twitter, Facebook, or Instagram.</td>
</tr>
<tr>
<td>social_handle</td>
<td>User handle on the social media channel.</td>
</tr>
<tr>
<td>social_handle_url</td>
<td>URL to the consumer's social channel handle.</td>
</tr>
<tr>
<td>state</td>
<td>State in which the consumer resides.</td>
</tr>
<tr>
<td>street</td>
<td>Consumer street address.</td>
</tr>
<tr>
<td>suffix</td>
<td>Consumer name suffix such as Jr., Sr., or II.</td>
</tr>
<tr>
<td>time_format</td>
<td>Format in which to display time.</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss (12 hour)</td>
</tr>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss (12 hour)</td>
</tr>
<tr>
<td></td>
<td>• HH:mm:ss: hh:mm:ss (24 hour)</td>
</tr>
<tr>
<td></td>
<td>• HH:mm:ss: hh:mm:ss (24 hour)</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system time format)</td>
</tr>
<tr>
<td>time_zone</td>
<td>Consumer time zone, such as Canada/Central or US/Eastern.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>title</td>
<td>Consumer business title such as Manager, Software Developer, or Contractor.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 60</td>
</tr>
<tr>
<td>user</td>
<td>Sys_id of the consumer user. Located in the Consumer User [csm_consumer_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>zip</td>
<td>Consumer zip code.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>New consumer record was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

**Response body parameters (JSON or XML)**

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Sys_id of the newly created consumer record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

curl -X POST "https://instance.servicenow.com/api/now/consumer" \
-H "Accept: application/json" \
-H "Content-Type: application/json" \
-d "{" \n"country": "USA", \n"notes": "Never comes in before 10am", \n"gender": "Female", \n"city": "San Marcos", \n"prefix": "Ms", \n"title": "Director", \n"notification": "1", \n"business_phone": "(555)555-1234", \n"mobile_phone": "(555)555-1235", \n"street": "123 Sesame St", \n"state": "CA", \n"fax": "(555)555-1236", \n"first_name": "Jane", \n"zip": "92001", \n"home_phone": "(555)555-1234", \n"last_name": "Brown", \n"active": "true", \n"middle_name": "Dell", \n"time_zone": "PST", \n"name": "Jane Brown", \n"household": "4", \n"user": "c3d35d82c37122005871d44d81d3ae91", \n"primary": "false" \n}" \
--user "username":"password"

{ 
"result": "0f5c13addb93230057c3fd441d9619b8"
}

Example: Python request

# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/consumer'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {"Content-Type":"application/json","Accept":"application/json"}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, data="{"country":"Norway","notes":"Do not call after 5pm","gender":"Female","city":"San Marcos","prefix":"Ms","suffix":"Jr.","title":"Director","notification":"1","business_phone":"(555)555-1234" }")

# Check for HTTP codes other than 201
if response.status_code != 201:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

"result":"00d033eddbd32300a2a451735e961972"
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, <code>v1</code> or <code>v2</code>. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. For requests that exceed this number of records, use the <code>sysparm_offset</code> parameter to paginate record retrieval. Data type: Number Default: 10</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, <code>sysparm_offset</code> is set to &quot;0&quot;. To simply page through all available records, use <code>sysparm_offset=sysparm_offset+sysparm_limit</code>, until you reach the end of all records. Do not pass a negative number in the <code>sysparm_offset</code> parameter. Data type: Number Default: 0</td>
</tr>
<tr>
<td>sysparm_query</td>
<td>Encoded query used to filter the result set. For example: <code>sysparm_query=caller_id=javascript:gs.getUserID()^active=true</code> The encoded query supports <code>order by</code>. To sort responses based on certain fields, use the ORDERBY and ORDERBYDESC clauses in <code>sysparm_query</code>. For example, <code>sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory</code></td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>filters all active records and orders the results in ascending order by number first, and then in descending order by category.</td>
</tr>
<tr>
<td></td>
<td>If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the property <code>glide.invalid_query.returns_no_rows</code>. Set this property to true to return no rows on an invalid query.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The <code>glide.invalid_query.returns_no_rows</code> property controls the behavior of all queries across the instance, such as in lists, scripts (<code>GlideRecord.query()</code>), and web service APIs.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

The endpoint may return the following JSON or XML elements in the response body. In addition to the list of elements defined below (which define the elements in a base system), the endpoint also returns any custom fields added to the Contact [customer_contact] table. For additional information on these elements, refer to your specific table definition [System Definition > Tables].

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account</td>
<td>Sys_id of the account record to which the contact is associated; Account [customer_account] table. Data type: String</td>
</tr>
</tbody>
</table>
| active  | Flag that indicates whether the contact is active within the system. Possible values:  
  • true: Contact is active  
  • false: Contact is inactive |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| agent_status  | Status of the agent. Possible values:  
  • Off work  
  • On break  
  • On route  
  • On site  
  Data type: String  
  Maximum length: 40 |
| building      | Sys_id of the record that describes the building in which the contact resides; Building [cmn_building] table.  
  Data type: String |
| calendar_integration | Calendar application that the contact uses.  
  • 1: Outlook  
  Data type: Number (Integer)  
  Default: 1 |
| city          | City in which the contact resides.  
  Data type: String  
  Maximum length: 40 |
| company       | Sys_id of the company record to which the contact is associated; Company [core_company] table.  
  Data type: String |
| cost_center   | Sys_id of the cost center associated with the contact; Cost Center [cmn_cost_center] table.  
  Data type: String |
| country       | Country code of the country in which the contact resides.  
  Data type: String |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>date_format</td>
<td>Format in which to display dates to contacts. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• dd/mm/yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd-mm-yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd.mm.yyyy</td>
</tr>
<tr>
<td></td>
<td>• mm-dd-yyyy</td>
</tr>
<tr>
<td></td>
<td>• yyyy-mm-dd</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system date format)</td>
</tr>
<tr>
<td>default_perspective</td>
<td>Sys_id of the default perspective for the contact. Located in the Menu List [sys_perspective] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>department</td>
<td>Sys_id of the department associated with the contact. Located in the Department [cmn_department] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>edu_status</td>
<td>Education status of the associated contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: faculty</td>
</tr>
<tr>
<td>email</td>
<td>Contact email address.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>employee_number</td>
<td>Contact employee number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>enable_multifactor_authn</td>
<td>Flag that indicates whether multifactor authorization is required for the contact to log in to the service portal.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| true: Multifactor authorization enabled  
false: Multifactor authorization disabled |
| Data type: Boolean            |
| Default: false                |
| failed_attempts               | Number of failed log in attempts.                                           |
| Data type: Number (Integer)   |
| first_name                    | Contact first name.                                                         |
| Data type: String             |
| Maximum length: 50            |
| gender                        | Contact gender.                                                             |
| Data type: String             |
| Maximum length: 40            |
| geolocation_tracked           | Flag that indicates whether the contact location is obtained through geotracking. |
| Possible values:              |
| • true: Contact location obtained through geotracking  
• false: Contact location not obtained through geotracking |
| Data type: Boolean            |
| Default value: false          |
| home_phone                    | Contact home phone number.                                                  |
| Data type: String             |
| Maximum length: 40            |
| internal_integration_user     | Flag that indicates whether the contact is an internal integration user.     |
| Possible values:              |
| • true: Internal integration user  
• false: Other type of user |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>introduction</td>
<td>Introduction</td>
</tr>
<tr>
<td>last_login</td>
<td>Date on which the contact last logged into the system.</td>
</tr>
<tr>
<td>last_login_device</td>
<td>Device the consumer used the last time they logged in to the system.</td>
</tr>
<tr>
<td>last_login_time</td>
<td>Date and time the contact logged in to the system.</td>
</tr>
<tr>
<td>last_name</td>
<td>Contact last name.</td>
</tr>
<tr>
<td>last_position_update</td>
<td>Date and time the last position was updated.</td>
</tr>
<tr>
<td>latitude</td>
<td>Latitude coordinate of the contact.</td>
</tr>
<tr>
<td>ldap_server</td>
<td>Sys_id of the LDAP server used by the contact to last log in to the system; LDAP Server [ldap_server_config] table.</td>
</tr>
<tr>
<td>location</td>
<td>Sys_id of the record that describes the location of the contact; Location [cmn_location] table.</td>
</tr>
<tr>
<td>locked_out</td>
<td>Flag that indicates if the contact is locked-out. Possible values:</td>
</tr>
</tbody>
</table>

Data type: Boolean  
Default: false

Data type: String
Maximum length: 40

Data type: String (Date)

Data type: String
Maximum length: 40

Data type: String (Date/time)

Data type: String
Maximum length: 50

Data type: String (Date/time)

Data type: Number (Floating point)
Maximum length: 40

Data type: String

Data type: String
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>Contact locked-out</td>
</tr>
<tr>
<td>false</td>
<td>Contact not locked-out</td>
</tr>
<tr>
<td>Data type:</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default:</td>
<td>false</td>
</tr>
<tr>
<td>longitude</td>
<td>Longitude coordinate of the contact.</td>
</tr>
<tr>
<td>Data type:</td>
<td>Number (Floating point)</td>
</tr>
<tr>
<td>Maximum length:</td>
<td>40</td>
</tr>
<tr>
<td>manager</td>
<td>Sys_id of the record that describes the direct supervisor of the contact; User [sys_user] table.</td>
</tr>
<tr>
<td>Data type:</td>
<td>String</td>
</tr>
<tr>
<td>middle_name</td>
<td>Contact middle name.</td>
</tr>
<tr>
<td>Data type:</td>
<td>Number (Floating point)</td>
</tr>
<tr>
<td>Maximum length:</td>
<td>50</td>
</tr>
<tr>
<td>mobile_phone</td>
<td>Contact mobile phone number.</td>
</tr>
<tr>
<td>Data type:</td>
<td>String</td>
</tr>
<tr>
<td>Maximum length:</td>
<td>40</td>
</tr>
<tr>
<td>name</td>
<td>Contact full name.</td>
</tr>
<tr>
<td>Data type:</td>
<td>String</td>
</tr>
<tr>
<td>Maximum length:</td>
<td>151</td>
</tr>
<tr>
<td>notification</td>
<td>Indicates whether the contact should receive notifications.</td>
</tr>
<tr>
<td>Valid values:</td>
<td>• 1: Disabled</td>
</tr>
<tr>
<td></td>
<td>• 2: Enabled</td>
</tr>
<tr>
<td>Data type:</td>
<td>Number (Integer)</td>
</tr>
<tr>
<td>Default:</td>
<td>2</td>
</tr>
<tr>
<td>on_schedule</td>
<td>Indicates the timeliness of dispatched service personnel.</td>
</tr>
<tr>
<td>Valid values:</td>
<td></td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>• Ahead: Ahead of schedule.</td>
</tr>
<tr>
<td></td>
<td>• behind_less30: Behind schedule, but less than 30 minutes.</td>
</tr>
<tr>
<td></td>
<td>• behind_30to60: Behind schedule between 30 and 60 minutes.</td>
</tr>
<tr>
<td></td>
<td>• behind_more60: Behind schedule more than 60 minutes.</td>
</tr>
<tr>
<td></td>
<td>• on_time: On schedule.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>phone</td>
<td>Contact business phone number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>photo</td>
<td>Photo image of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: Image</td>
</tr>
<tr>
<td>preferred_language</td>
<td>Country code of the contact primary language.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 3</td>
</tr>
<tr>
<td>roles</td>
<td>List of user roles associated with the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>schedule</td>
<td>Sys_id of the record that describes the work schedule for the associated contact; Schedule [cmn_schedule] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>source</td>
<td>Source of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 255</td>
</tr>
<tr>
<td>state</td>
<td>State in which the contact resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| street          | Contact street address.  
Data type: String  
Maximum length: 255 |
| sys_class_name  | Table that contains the contact record.  
Data type: String  
Maximum length: 80 |
| sys_created_by  | User that originally created the associated contact record.  
Data type: String  
Maximum length: 40 |
| sys_created_on  | Data and time the associated contact was originally created.  
Data type: String (Date/time) |
| sys_domain      | ServiceNow instance domain of the associated contact record.  
Data type: String |
| sys_domain_path | Contact record domain path.  
Data type: String  
Maximum length: 255  
Default: / (global) |
| sys_id          | Unique identifier for the associated contact record.  
Data type: String |
| sys_mod_count   | Number of times that the associated contact record has been modified.  
Data type: Number (Integer) |
| sys_tags        | System tags.  
Data type: String |
| sys_updated_by  | User that last updated the associated contact information.  
Data type: String |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_updated_on</td>
<td>Data and time the associated contact information was updated. Data type: String (Date/time)</td>
</tr>
</tbody>
</table>
| time_format         | Format in which to display time. Valid values:  
  • hh:mm:ss a: hh:mm:ss (12 hour)  
  • hh:mm:ss a: hh:mm:ss (12 hour)  
  • HH:mm:ss: hh:mm:ss (24 hour)  
  • HH:mm:ss: hh:mm:ss (24 hour)  
  Data type: String  
  Maximum length: 40  
  Default: Blank (system time format) |
| time_sheet_policy   | Sys_id of the record that contains the time sheet policy for the associated contact; Time Sheet Policy [time_sheet_policy] table. Data type: String |
| time_zone           | Time zone in which the contact resides, such as Canada/Central or US/Eastern. Data type: String  
  Maximum length: 40 |
| title               | Contact business title such as Manager, Software Developer, or Contractor. Data type: String  
  Maximum length: 60 |
| user_name           | Contact user ID. Data type: String  
  Maximum length: 40 |
<p>| vip                 | Flag that indicates whether the associated contact has VIP status. |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: VIP</td>
</tr>
<tr>
<td></td>
<td>• false: Not VIP</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>web_service_access_only</td>
<td>Flag that indicates whether the contact can only access services through the web.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Web access only</td>
</tr>
<tr>
<td></td>
<td>• false: Access through all available methods</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>zip</td>
<td>Contact zip code.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
  "https://instance.servicenow.com/api/now/contact?sysparm_query=account=86837a386f0331003b3c498f5d3ee4ca&sysparm_limit=2&sysparm_offset=2>;rel="next" \
  --request GET \
  --header "Accept:application/json" \
  --user "username":"password"

{
  "result": [
  
  
  "country": ",",
  "calendar_integration": "1",
  "last_position_update": ",",
  "last_login_time": "2018-03-10 21:48:11",
  "last_login_device": ",",
  "source": ",",
  "sys_updated_on": "2019-01-03 05:49:34",
```
"email": "geo.warren@mailinator.com",
"manager": "",
"locked_out": "false",
"sys_mod_count": "3",
"last_name": "Warren",
"photo": "",
"sys_tags": "",
"middle_name": "",
"time_zone": "",
"schedule": "",
"on_schedule": "",
"date_format": "",
"location": "25ab8e460a0a0bb300857304ff811af5",
"account": "86837a386f0331003b3c498f5d3ee4ca"
},
{
"country": "",
"calendar_integration": "1",
"last_position_update": "",
"last_login_time": "2019-01-03 15:08:57",
"last_login_device": "73.71.157.241",
"source": "",
"sys_updated_on": "2019-01-03 23:26:12",
"building": "",
"web_service_access_only": "false",
"notification": "2",
"sys_updated_by": "admin",
"enable_multifactor_authn": "false",
"sys_created_on": "2019-01-03 15:07:25",
"sys_domain": "global",
"agent_status": "",
"state": "",
"vip": "false",
"sys_created_by": "carl.customer",
"longitude": "",
"zip": "",
"home_phone": "",
"time_format": "",
"last_login": "",
"default_perspective": "",
"geolocation_tracked": "false",
"active": "true",
"time_sheet_policy": "",
"sys_domain_path": "/"
Example: Python request

```python
# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/contact?sysparm_limit=1&sysparm_offset=2>;rel="next"'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept':"application/xml"}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <calendar_integration>1</calendar_integration>
    <last_position_update/>
    <last_login_time/>
    <last_login_device/>
    <source/>
    <sys_updated_on>2018-05-14 21:11:33</sys_updated_on>
    <building/>
    <web_service_access_only>false</web_service_access_only>
    <notification>2</notification>
    <sys_updated_by>admin</sys_updated_by>
    <enable_multifactor_authn>false</enable_multifactor_authn>
    <sys_created_on>2018-05-14 20:55:30</sys_created_on>
  </result>
</response>
```
<sys_domain>global</sys_domain>
<agent_status/>
<vip>false</vip>
<sys_created_by>admin</sys_created_by>
<longitude/>
<zip/>
<home_phone/>
<time_format/>
<last_login/>
<default_perspective/>
<geolocation_tracked>false</geolocation_tracked>
<active>true</active>
<time_sheet_policy/>
<sys_domain_path>/</sys_domain_path>
<phone>498-987-9999</phone>
<cost_center/>
<name>Amy Pascal</name>
<employee_number/>
<gender/>
<city/>
<user_name>amy.pascal</user_name>
<failed_attempts/>
<edu_status>faculty</edu_status>
<latitude/>
<roles/>
<title>Manager</title>
<sys_class_name>customer_contact</sys_class_name>
<sys_id>268e22b0d7300200e5982cf65e6103d3</sys_id>
<internal_integration_user>false</internal_integration_user>
<ldap_server/>
<mobile_phone/>
<street/>
<company>051f62b0d7300200e5982cf65e610333</company>
<department/>
<first_name>Amy</first_name>
<preferred_language/>
<introduction/>
$email>amy.pasal@mailinator.com</email>
<manager/>
<locked_out>false</locked_out>
<sys_mod_count>8</sys_mod_count>
<last_name>Pascal</last_name>
<photo/>
Contact - GET /now/contact/{id}
Retrieves the specified Customer Service Management (CSM) contact.

**URL format**

Versioned URL: /api/now/{api_version}/contact/{id}
Default URL: /api/now/contact/{id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>Sys_id of the contact to retrieve. Located in the Contact [customer_contact] table.</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
</tbody>
</table>
| 404         | Indicates that the request is invalid. Could be due to one of the following reasons:  
  - Requested case does not exist.  
  - User does not have access to the case. |
| 500         | Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error. |
Response body parameters (JSON or XML)

The endpoint may return the following JSON or XML elements in the response body. In addition to the list of elements defined below (which define the elements in a base system), the endpoint also returns any custom fields added to the Contact [customer_contact] table. For additional information on these elements, refer to your specific table definition [System Definition > Tables].

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account</td>
<td>Sys_id of the account record to which the contact is associated; Account [customer_account] table.</td>
</tr>
<tr>
<td>active</td>
<td>Flag that indicates whether the contact is active within the system.</td>
</tr>
<tr>
<td>calendar_integration</td>
<td>Calendar application that the contact uses.</td>
</tr>
<tr>
<td>agent_status</td>
<td>Status of the agent.</td>
</tr>
<tr>
<td>building</td>
<td>Sys_id of the record that describes the building in which the contact resides; Building [cmn_building] table.</td>
</tr>
<tr>
<td>building</td>
<td></td>
</tr>
</tbody>
</table>

**Example:**

```
{  
  "account": "87654321",  
  "active": true,           
  "agent_status": "On site",  
  "building": "123 Main St",  
  "calendar_integration": "Outlook"  
}
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 1</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>company</td>
<td>Sys_id of the company record to which the contact is associated; Company [core_company] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>cost_center</td>
<td>Sys_id of the cost center associated with the contact; Cost Center [cmn_cost_center] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>country</td>
<td>Country code of the country in which the contact resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 3</td>
</tr>
<tr>
<td>date_format</td>
<td>Format in which to display dates to contacts.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• dd/mm/yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd-mm-yyyy</td>
</tr>
<tr>
<td></td>
<td>• dd.mm.yyyy</td>
</tr>
<tr>
<td></td>
<td>• mm-dd-yyyy</td>
</tr>
<tr>
<td></td>
<td>• yyyy-mm-dd</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system date format)</td>
</tr>
<tr>
<td>default_perspective</td>
<td>Sys_id of the default perspective for the contact. Located in the Menu List [sys_perspective] table.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>department</td>
<td>Sys_id of the department associated with the contact. Located in the Department [cmn_department] table. Data type: String</td>
</tr>
<tr>
<td>edu_status</td>
<td>Education status of the associated contact. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: faculty</td>
</tr>
<tr>
<td>email</td>
<td>Contact email address. Data type: String</td>
</tr>
<tr>
<td>employee_number</td>
<td>Contact employee number. Data type: String</td>
</tr>
<tr>
<td>enable_multifactor_authn</td>
<td>Flag that indicates whether multifactor authorization is required for the contact to log in to the service portal. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Multifactor authorization enabled</td>
</tr>
<tr>
<td></td>
<td>• false: Multifactor authorization disabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>failed_attempts</td>
<td>Number of failed log in attempts. Data type: Number (Integer)</td>
</tr>
<tr>
<td>first_name</td>
<td>Contact first name. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>gender</td>
<td>Contact gender. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>geolocation_tracked</td>
<td>Flag that indicates whether the contact location is obtained through geotracking.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Contact location obtained through geotracking</td>
</tr>
<tr>
<td></td>
<td>• false: Contact location not obtained through geotracking</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default value: false</td>
</tr>
<tr>
<td>home_phone</td>
<td>Contact home phone number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>internal_integration_user</td>
<td>Flag that indicates whether the contact is an internal integration user.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Internal integration user</td>
</tr>
<tr>
<td></td>
<td>• false: Other type of user</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>introduction</td>
<td>Introduction</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>last_login</td>
<td>Date on which the contact last logged into the system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date)</td>
</tr>
<tr>
<td>last_login_device</td>
<td>Device the consumer used the last time they logged in to the system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>last_login_time</td>
<td>Date and time the contact logged in to the system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date/time)</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>last_name</td>
<td>Contact last name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>last_position_update</td>
<td>Date and time the last position was updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date/time)</td>
</tr>
<tr>
<td>latitude</td>
<td>Latitude coordinate of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Floating point)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>ldap_server</td>
<td>Sys_id of the LDAP server used by the contact to last log in to the system; LDAP Server [ldap_server_config] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>location</td>
<td>Sys_id of the record that describes the location of the contact; Location [cmn_location] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>locked_out</td>
<td>Flag that indicates if the contact is locked-out.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Contact locked-out</td>
</tr>
<tr>
<td></td>
<td>• false: Contact not locked-out</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>longitude</td>
<td>Longitude coordinate of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Floating point)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>manager</td>
<td>Sys_id of the record that describes the direct supervisor of the contact; User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>middle_name</td>
<td>Contact middle name.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Floating point)</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>mobile_phone</td>
<td>Contact mobile phone number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>name</td>
<td>Contact full name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 151</td>
</tr>
<tr>
<td>notification</td>
<td>Indicates whether the contact should receive notifications.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Disabled</td>
</tr>
<tr>
<td></td>
<td>• 2: Enabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 2</td>
</tr>
<tr>
<td>on_schedule</td>
<td>Indicates the timeliness of dispatched service personnel.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• Ahead: Ahead of schedule</td>
</tr>
<tr>
<td></td>
<td>• behind_less30: Behind schedule, but less than 30 minutes</td>
</tr>
<tr>
<td></td>
<td>• behind_30to60: Behind schedule between 30 and 60 minutes</td>
</tr>
<tr>
<td></td>
<td>• behind_more60: Behind schedule more than 60 minutes</td>
</tr>
<tr>
<td></td>
<td>• on_time: On schedule</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>phone</td>
<td>Contact business phone number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>photo</td>
<td>Photo image of the contact. Data type: Image</td>
</tr>
<tr>
<td>preferred_language</td>
<td>Country code of the contact primary language. Data type: String Maximum length: 3</td>
</tr>
<tr>
<td>roles</td>
<td>List of user roles associated with the contact. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>schedule</td>
<td>Sys_id of the record that describes the work schedule for the associated contact; Schedule [cmn_schedule] table. Data type: String</td>
</tr>
<tr>
<td>source</td>
<td>Source of the contact. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>state</td>
<td>State in which the contact resides. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>street</td>
<td>Contact street address. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>sys_class_name</td>
<td>Table that contains the contact record. Data type: String Maximum length: 80</td>
</tr>
<tr>
<td>sys_created_by</td>
<td>User that originally created the associated contact record. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| sys_created_on      | Data and time the associated contact was originally created.  
Data type: String (Date/time)                                                                 |
| sys_domain          | ServiceNow instance domain of the associated contact record.  
Data type: String                                                                 |
| sys_domain_path     | Contact record domain path.  
Data type: String  
Maximum length: 255  
Default: / (global)                                                                 |
| sys_id              | Unique identifier for the associated contact record.  
Data type: String                                                                 |
| sys_mod_count       | Number of times that the associated contact record has been modified.  
Data type: Number (Integer)                                                                 |
| sys_updated_by      | User that last updated the associated contact information.  
Data type: String  
Maximum length: 40                                                                                           |
| sys_updated_on      | Data and time the associated contact information was updated.  
Data type: String (Date/time)                                                                 |
| time_format         | Format in which to display time.  
Valid values:  
• hh.mm.ss a: hh.mm.ss (12 hour)  
• hh:mm:ss a: hh:mm:ss (12 hour)  
• HH.mm.ss: hh.mm.ss (24 hour)  
• HH:mm:ss: hh:mm:ss (24 hour)  
Data type: String  
Maximum length: 40                                                                                           |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default: Blank (system time format)</td>
<td></td>
</tr>
<tr>
<td>time_sheet_policy</td>
<td>Sys_id of the record that contains the time sheet policy for the associated contact; Time Sheet Policy [time_sheet_policy] table. Data type: String</td>
</tr>
<tr>
<td>time_zone</td>
<td>Time zone in which the contact resides, such as Canada/Central or US/Eastern. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>title</td>
<td>Contact business title such as Manager, Software Developer, or Contractor. Data type: String Maximum length: 60</td>
</tr>
<tr>
<td>user_name</td>
<td>Contact user ID. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>vip</td>
<td>Flag that indicates whether the associated contact has VIP status. Possible values: • true: VIP • false: Not VIP Data type: Boolean Default: false</td>
</tr>
<tr>
<td>web_service_access_only</td>
<td>Flag that indicates whether the contact can only access services through the web. Possible values: • true: Web access only • false: Access through all available methods Data type: Boolean Default: false</td>
</tr>
<tr>
<td><strong>Element</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>zip</td>
<td>Contact zip code.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl "https://instance.servicenow.com/api/now/contact/ddce70866f9331003b3c498f5d3ee417 \
--request GET \
--header "Accept:application/json" \
--user "username":"password"
```

```json
{
  "result": {
    "country": "",
    "calendar_integration": "1",
    "last_position_update": "",
    "last_login_time": "2018-03-10 21:48:11",
    "last_login_device": "",
    "source": "",
    "sys_updated_on": "2019-01-03 05:49:34",
    "building": "",
    "web_service_access_only": "false",
    "notification": "2",
    "sys_updated_by": "system",
    "enable_multifactor_authn": "false",
    "sys_created_on": "2018-03-04 20:26:32",
    "sys_domain": "global",
    "agent_status": "",
    "state": "",
    "vip": "false",
    "sys_created_by": "admin",
    "longitude": "",
    "zip": "",
    "home_phone": "",
    "time_format": "",
    "last_login": "",
    "default_perspective": "",
    "geolocation_tracked": "false",
    "active": "true",
    "time_sheet_policy": "",
    "sys_domain_path": "/",
    "phone": "+1 858 287 7834",
```
"cost_center": "",
"name": "George Warren",
"employee_number": "",
"gender": "",
"city": "",
"user_name": "george.warren",
"failed_attempts": "",
"edu_status": "",
"latitude": "",
"roles": "",
"title": "Network Administrator",
"sys_class_name": "customer_contact",
"sys_id": "ddce70866f9331003b3c498f5d3ee417",
"internal_integration_user": "false",
"ldap_server": "",
"mobile_phone": "+1 858 867 7857",
"street": "",
"company": "86837a386f0331003b3c498f5d3ee4ca",
"department": "",
"first_name": "George",
"preferred_language": "",
"introduction": "",
"email": "geo.warren@mailinator.com",
"manager": "",
"locked_out": "false",
"sys_mod_count": "3",
"last_name": "Warren",
"photo": "",
"sys_tags": "",
"middle_name": "",
"time_zone": "",
"schedule": "",
"on_schedule": "",
"date_format": "",
"location": "25ab8e460a0a0bb300857304ff811af5",
"account": "86837a386f0331003b3c498f5d3ee4ca"

Example: Python request

#Need to install requests package for python
import requests
# Set the request parameters
url = 'https://instance.servicenow.com/api/now/contact/268e22b0d7300200e5982cf65e6103d3'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept':"application/xml"}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <calendar_integration>1</calendar_integration>
    <last_position_update/>
    <last_login_time/>
    <last_login_device/>
    <source/>2018-05-14 21:11:33</source>
    <building/>
    <web_service_access_only>false</web_service_access_only>
    <notification>2</notification>
    <sys_updated_on>2018-05-14 20:55:30</sys_updated_on>
    <sys_domain>global</sys_domain>
    <agent_status/>
    <state/>
    <vip>false</vip>
    <sys_created_by>admin</sys_created_by>
    <longitude/>
    <zip/>
  </result>
</response>
Contact - POST /now/contact

Creates a new Customer Service Management (CSM) contact.

In addition, you can create a social media profile for the contact using this endpoint. To create the profile, you must specify the following parameters in the request body:

- `social_channel`
- `social_handle`
- `social_handle_url`

⚠️ **Warning:** This endpoint does not perform parameter validation as doing so can create excessive overhead. If a request parameter is misspelled, is not valid, or is not supported by the endpoint, it is ignored without warning.

**URL format**

Versioned URL: `/api/now/{api_version}/contact`
Default URL: `/api/now/contact`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account</td>
<td>Sys_id of the account record to which the contact is associated; Account [customer_account] table. Data type: String</td>
</tr>
<tr>
<td>active</td>
<td>Flag that indicates whether the contact is active within the system. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Contact is active</td>
</tr>
<tr>
<td></td>
<td>• false: Contact is inactive</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: true</td>
</tr>
<tr>
<td>agent_status</td>
<td>Status of the agent. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Off work</td>
</tr>
<tr>
<td></td>
<td>• On break</td>
</tr>
<tr>
<td></td>
<td>• On route</td>
</tr>
<tr>
<td></td>
<td>• On site</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>building</td>
<td>Sys_id of the record that describes the building in which the contact resides; Building [cmn_building] table. Data type: String</td>
</tr>
<tr>
<td>calendar_integration</td>
<td>Calendar application that the contact uses. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Outlook</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Integer)</td>
</tr>
<tr>
<td></td>
<td>Default: 1</td>
</tr>
<tr>
<td>city</td>
<td>City in which the contact resides.</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>company</td>
<td>Sys_id of the company record to which the contact is associated; Company [core_company] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>cost_center</td>
<td>Sys_id of the cost center associated with the contact; Cost Center [cmn_cost_center] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>country</td>
<td>Country code of the country in which the contact resides.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 3</td>
</tr>
<tr>
<td>date_format</td>
<td>Format in which to display dates to contacts.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>- dd/mm/yyyy</td>
</tr>
<tr>
<td></td>
<td>- dd-mm-yyyy</td>
</tr>
<tr>
<td></td>
<td>- dd.mm.yyyy</td>
</tr>
<tr>
<td></td>
<td>- mm-dd-yyyy</td>
</tr>
<tr>
<td></td>
<td>- yyyy-mm-dd</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: blank (system date format)</td>
</tr>
<tr>
<td>default_perspective</td>
<td>Sys_id of the default perspective for the contact. Located in the Menu List [sys_perspective] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>department</td>
<td>Sys_id of the department associated with the contact. Located in the Department [cmn_department] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>edu_status</td>
<td>Education status of the associated contact.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Maximum length: 40</td>
<td></td>
</tr>
<tr>
<td>Default: faculty</td>
<td></td>
</tr>
<tr>
<td>email</td>
<td>Contact email address.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>employee_number</td>
<td>Contact employee number.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>enable_multifactor_authn</td>
<td>Flag that indicates whether multifactor authorization is required for the</td>
</tr>
<tr>
<td></td>
<td>contact to log in to the service portal.</td>
</tr>
<tr>
<td>Possible values:</td>
<td></td>
</tr>
<tr>
<td>• true: Multifactor</td>
<td>Authorization enabled</td>
</tr>
<tr>
<td>• false: Multifactor</td>
<td>Authorization disabled</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>Default: false</td>
<td></td>
</tr>
<tr>
<td>failed_attempts</td>
<td>Number of failed log in attempts.</td>
</tr>
<tr>
<td>Data type: Number (Integer)</td>
<td></td>
</tr>
<tr>
<td>first_name</td>
<td>Contact first name.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Maximum length: 50</td>
<td></td>
</tr>
<tr>
<td>gender</td>
<td>Contact gender.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Maximum length: 40</td>
<td></td>
</tr>
<tr>
<td>geolocation_tracked</td>
<td>Flag that indicates whether the contact location is obtained through</td>
</tr>
<tr>
<td></td>
<td>geotracking.</td>
</tr>
<tr>
<td>Possible values:</td>
<td></td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>home_phone</td>
<td>Contact home phone number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>internal_integration_user</td>
<td>Flag that indicates whether the contact is an internal integration user.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Internal integration user</td>
</tr>
<tr>
<td></td>
<td>• false: Other type of user</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>introduction</td>
<td>Introduction</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>last_login_device</td>
<td>Device the consumer used the last time they logged in to the system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>last_login_time</td>
<td>Date and time the contact logged in to the system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String (Date/time)</td>
</tr>
<tr>
<td>last_name</td>
<td>Contact last name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>latitude</td>
<td>Latitude coordinate of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Floating point)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>ldap_server</td>
<td>Sys_id of the LDAP server used by the contact to last log in to the system; LDAP Server [ldap_server_config] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>location</td>
<td>Sys_id of the record that describes the location of the contact; Location [cmn_location] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>locked_out</td>
<td>Flag that indicates if the contact is locked-out.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Contact locked-out</td>
</tr>
<tr>
<td></td>
<td>• false: Contact not locked-out</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>longitude</td>
<td>Longitude coordinate of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Floating point)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>manager</td>
<td>Sys_id of the record that describes the direct supervisor of the contact; User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>middle_name</td>
<td>Contact middle name.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number (Floating point)</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 50</td>
</tr>
<tr>
<td>mobile_phone</td>
<td>Contact mobile phone number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| name         | Contact full name.  
Data type: String  
Maximum length: 151                                   |
| notification | Indicates whether the contact should receive notifications.  
Valid values:  
• 1: Disabled  
• 2: Enabled  
Data type: Number (Integer)  
Default: 2                                    |
| on_schedule  | Indicates the timeliness of dispatched service personnel.  
Valid values:  
• Ahead: Ahead of schedule.  
• behind_less30: Behind schedule, but less than 30 minutes.  
• behind_30to60: Behind schedule between 30 and 60 minutes.  
• behind_more60: Behind schedule more than 60 minutes.  
• on_time: On schedule.  
Data type: String  
Maximum length: 40                                    |
| phone        | Contact business phone number.  
Data type: String  
Maximum length: 40                                    |
| photo        | Photo image of the contact.  
Data type: Image                                         |
## Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>preferred_language</td>
<td>Country code of the contact primary language. Data type: String Maximum length: 3</td>
</tr>
<tr>
<td>roles</td>
<td>List of user roles associated with the contact. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>schedule</td>
<td>Sys_id of the record that describes the work schedule for the associated contact; Schedule [cmn_schedule] table. Data type: String</td>
</tr>
<tr>
<td>social_channel</td>
<td>Social media channel to which the contact is associated such as Twitter, Facebook, or Instagram. Data type: String</td>
</tr>
<tr>
<td>social_handle</td>
<td>User handle on the social media channel. Data type: String</td>
</tr>
<tr>
<td>social_handle_url</td>
<td>URL to the contact's social channel profile. Data type: String</td>
</tr>
<tr>
<td>source</td>
<td>Source of the contact. Data type: String Maximum length: 255</td>
</tr>
<tr>
<td>state</td>
<td>State in which the contact resides. Data type: String Maximum length: 40</td>
</tr>
<tr>
<td>street</td>
<td>Contact street address. Data type: String Maximum length: 255</td>
</tr>
</tbody>
</table>
| time_format           | Format in which to display time. Valid values:
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• hh:mm:ss a: hh:mm:ss (12 hour)                                      • hh:mm:ss a: hh:mm:ss (12 hour)                                      • HH:mm:ss: hh:mm:ss (24 hour)                                      • HH:mm:ss: hh:mm:ss (24 hour)</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td></td>
<td>Default: Blank (system time format)</td>
</tr>
<tr>
<td>time_sheet_policy</td>
<td>Sys_id of the record that contains the time sheet policy for the associated contact; Time Sheet Policy [time_sheet_policy] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>time_zone</td>
<td>Time zone in which the contact resides, such as Canada/Central or US/Eastern.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>title</td>
<td>Contact business title such as Manager, Software Developer, or Contractor.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 60</td>
</tr>
<tr>
<td>user_name</td>
<td>Contact user ID.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
<tr>
<td>vip</td>
<td>Flag that indicates whether the associated contact has VIP status.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: VIP</td>
</tr>
<tr>
<td></td>
<td>• false: Not VIP</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>web_service_access_only</td>
<td>Flag that indicates whether the contact can only access services through the web.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>- true: Web access only</td>
</tr>
<tr>
<td></td>
<td>- false: Access through all available methods</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>zip</td>
<td>Contact zip code.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 40</td>
</tr>
</tbody>
</table>

Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>New contact record was successfully created.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Sys_id of the newly created contact record. Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

```
curl -X POST "https://instance.servicenow.com/api/now/contact" \
-H 'Accept: application/json' \
-H 'Content-Type: application/json' \
-d '{ "country": "USA", "calendar_integration": "1", "last_login_time": "2018-03-10 21:48:11", "last_login_device": "tablet", "building": "Cardinal West", "web_service_access_only": "false", "notification": "1", "enable_multifactor_authn": "true", "agent_status": "Travelling", "state": "CA", "vip": "false", "longitude": "123.76", \
}
```
"zip": "92069", 
"home_phone": "(555) 555-1234", 
"time_format": "hh:mm:ss", 
"geolocation_tracked": "false", 
"active": "true", 
"phone": "+1 858 287 7834", 
"cost_center": "1345", 
"name": "Dora Warren", 
"employee_number": "546", 
"gender": "Female", 
"city": "Orlando", 
"user_name": "dora.warren", 
"failed_attempts": "2", 
"edu_status": "current", 
"latitude": "57.6", 
"title": "Network Administrator", 
"internal_integration_user": "false", 
"ldap_server": "10.24.23.123", 
"mobile_phone": "+1 858 867 7857", 
"street": "123 Lagume", 
"company": "86837a386f0331003b3c498f5d3ee4ca", 
"department": "IT", 
"first_name": "Dora", 
"preferred_language": "Spanish", 
"email": "dora.warren@mailinator.com", 
"manager": "ddce70866f9331003b3c498f5d3ee417", 
"locked_out": "false", 
"last_name": "Warren", 
"middle_name": "Dell", 
"time_zone": "PST", 
"schedule": "9-5", 
"date_format": "MM/DD/YY", 
"location": "25ab8e460a0a0bb300857304ff811af5", 
"account": "86837a386f0331003b3c498f5d3ee4ca" 
'}
--user 'username':'password'

```
"result": "62fe1c97db76c3006b7a9646db961999"
```

**Example: Python request**

```python
# Need to install requests package for python
import requests

# Python request
response = requests.get('http://example.com/endpoint', auth=('username', 'password'))
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
# Set the request parameters
url = 'https://instance.servicenow.com/api/now/contact'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {"Content-Type":"application/json","Accept":"application/json"}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, data="{"account":"86837a386f0331003b3c498f5d3ee4ca","gender": "Female", "first_name": "Joan", "last_name": "Jetson", "phone": "(555)555-1234" }")

# Check for HTTP codes other than 201
if response.status_code != 201:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

"result":"380d45b5dbd72300a2a451735e96190a"

Continuous Integration/Continuous Delivery (CI/CD) API

The CI/CD API provides endpoints that enable integration with common DevOps tools for building pipelines. These tools include Jenkins, Azure Pipelines, GitLab, GitHub Actions, Atlassian Bamboo/Pipelines, and more.

It provides integrators with the ability to build CI/CD pipelines for Now Platform applications driven by pull/merge request workflows typical in source-driven development. It leverages the Source Control, Application Repository, and Automated Test Framework features.

The CI/CD API provides the following product and feature areas:

- Source control
  - Apply remote changes for an application linked to a remote Git repository through the Source Control feature in Studio, on a specific instance.
• Application repository: Available for scoped applications, global application, application customizations for scoped applications on the ServiceNow Store, and through plugins.
  ◦ Publish an application from an instance to the connected application repository.
  ◦ Install an application onto an instance from the connected application repository.
  ◦ Rollback an application that was installed through rollback context.
• Batch
  ◦ Define a JSON manifest file to batch install multiple applications together.
  ◦ Rollback the entire batch install through rollback context.
• Plugins
  ◦ Activate a plugin.
  ◦ Rollback a plugin through rollback context.
• Automated Test Framework (ATF)
  ◦ Start an ATF test suite.
  ◦ Obtain the pass/fail result of an ATF test suite.
• Instance scan
  ◦ Start an instance scan (full, point scan on table, point scan on record, scoped application, update set, combination of these items.)
  ◦ Obtain the pass/fail result of an instance scan run.
• Progress
  ◦ Obtain the percentage completion on an instance for processes initiated by other endpoint calls. You can use this feature to poll until progress completion before proceeding with the next step.

This API uses an asynchronous response model. When you call an endpoint, such as to run a test suite /sn_cicd/testsuite/run, the endpoint kicks off the associated function, such as starting the execution of the test suite. The endpoint then returns a response message that contains the execution status of the requested function, typically Pending, along with a unique identifier and link. Use this information to obtain additional information about the progress, source, results, and/or findings of the request. Depending on the function request, you can use one of the following means to obtain additional information about the request:
• Call the /sn_cicd/progress/{progress_id} endpoint using the links.progress.id parameter returned by the initiating endpoint.

• Call the /sn_cicd/testsuite/results/{result_id} endpoint using the links.result.id parameter returned by the initiating endpoint.

• Call the /sn_cicd/instance_scan/result/{progress_id} endpoint using the links.progress.id parameter returned by the initiating endpoint.

• Link out to the links.<progress/result/source/findings>.url parameter returned by the initiating endpoint.

**Note:** To install an application from a repository, the application must first be published to the repository using the /sn_cicd/app_repo/publish endpoint. Once published, any instance connected to that repository can then install and rollback the application. If the application is not first published, the installation request fails.

The sn_cicd.sys_ci_automation or admin role is required to use this API.

For additional information on CI/CD, see:

• Continuous Integration and Continuous Delivery (CICD) Spoke
• https://plugins.jenkins.io/servicenow-cicd/
• https://marketplace.visualstudio.com/items?itemName=ServiceNow.vss-services-servicenow-cicd
• https://github.com/marketplace?type=actions&query=servicenow
• https://hub.docker.com/r/servicenowdevx/sncicd-gitlab-docker

**CI/CD - GET /sn_cicd/app/batch/results/{result_id}**

Returns the results of a batch install based on a specified result identifier.

Call this method after calling the /sn_cicd/app/batch/install endpoint to obtain the results of the batch install. Use the value returned in the links.results.id parameter of the return results of /sn_cicd/app/batch/install endpoint as the result_id for this endpoint. You can also call the /api/sn_cicd/progress/{progress_id} endpoint to obtain information on the progress of the installation if it has not yet completed.

**URL format**

Versioned URL: /api/sn_cicd/{api_version}/app/batch/results/{result_id}

Default URL: /api/sn_cicd/app/batch/results/{result_id}
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>result_id</td>
<td>Sys_id of the batch installation for which to return results information. This value is returned in the <code>links.results.id</code> parameter of the <code>/sn_cicd/app/batch/install</code> endpoint. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json Of application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| error         | Error message.  
Data type: String |
| batch_items   | JSON array, where each object provides details of a package installation.  
Data type: Array  
```
"batch_items": [{
   "customization_version": "String",
   "id": "String",
   "install_date": "String",
   "name": "String",
   "notes": "String",
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>batch_items.customization_version</td>
<td>Only applicable when the <code>batch_items.type</code> parameter is set to <code>application</code> or <code>plugin</code> and the <code>requested_customization_version</code> parameter is passed in the batch install request. Version of the store application or scoped ServiceNow plugin customization package to install, such as 1.0.2 or 2.3. Data type: String</td>
</tr>
<tr>
<td>batch_items.id</td>
<td>Sys_id of the application or identifier of the plugin to install. Same as the <code>packages.id</code> parameter in the initiating <code>/sn_cicd/app/batch/install</code> endpoint call. Data type: String</td>
</tr>
<tr>
<td>batch_items.install_date</td>
<td>Date and time that the package was installed. Empty if the request is still in progress. Data type: String</td>
</tr>
<tr>
<td>batch_items.name</td>
<td>Name of the package. Data type: String</td>
</tr>
<tr>
<td>batch_items.notes</td>
<td>User specified notes about the package. Same as the <code>packages.notes</code> parameter in the initiating <code>/sn_cicd/app/batch/install</code> endpoint call. Data type: String</td>
</tr>
</tbody>
</table>
| batch_items.state | Current state of the associated package installation. Possible values:  
• Failed  
• In progress |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>batch_items.status_message</td>
<td>Describes any errors that occurred during the installation of the package and/or customizations. Data type: String</td>
</tr>
<tr>
<td>batch_items.type</td>
<td>Type of application. Same as the <code>packages.type</code> parameter in the initiating <code>/sn_cicd/app/batch/install</code> endpoint call. Valid values: application, plugin. Data type: String</td>
</tr>
<tr>
<td>batch_items.url</td>
<td>URL of the associated package installation record on your ServiceNow instance. Data type: String</td>
</tr>
<tr>
<td>batch_items.version</td>
<td>Version of the package to install, such as 1.0.2 or 2.3. Same as the <code>packages.requested_version</code> parameter in the initiating <code>/sn_cicd/app/batch/install</code> endpoint call. Data type: String</td>
</tr>
<tr>
<td>batch_plan</td>
<td>Describes the installation batch plan. Data type: Object</td>
</tr>
</tbody>
</table>

```
"batch_plan": {
  "id": "String",
  "name": "String",
  "notes": "String",
  "state": "String",
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>batch_plan.id</td>
<td>Sys_id of the return results information. Same value as what was passed in the <code>result_id</code> request parameter. Data type: String</td>
</tr>
<tr>
<td>batch_plan.name</td>
<td>User specified descriptive name for this batch request. Same as the <code>name</code> parameter in the initiating <code>/sn_cicd/app/batch/install</code> endpoint call. Data type: String</td>
</tr>
<tr>
<td>batch_plan.notes</td>
<td>Notes that were passed in when the batch install was invoked. Same as the <code>notes</code> parameter in the initiating <code>/sn_cicd/app/batch/install</code> endpoint call. Data type: String</td>
</tr>
<tr>
<td>batch_plan.state</td>
<td>Current state of the overall batch installation. Possible values: • Failed • In progress • Installed • Invalid • Partial Install • Ready • Rolled Back Data type: String</td>
</tr>
<tr>
<td>batch_plan.url</td>
<td>URL of the batch installation plan record on your ServiceNow instance. Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

This example checks the status of a batch installation plan called Release 2.0 IT Operations.

curl
'instance.servicenow.com/api/sn_cicd/app/batch/results/df24b1e9db2d0110b5e3f6c5ae97c561 \
--request GET \
--header 'Accept: application/json' \
--user 'username':'password'

{
  "result": {
    "batch_plan": {
      "name": "Release 2.0 IT Operations",
      "id": "df24b1e9db2d0110b5e3f6c5ae97c561",
      "url": "https://instance.service-now.com/sys_batch_install_plan.do?sys_id=df24b1e9db2d0110b5e3f6c5ae97c561",
      "state": "Installed",
      "notes": "User specified notes for batch install plan",
    }
    "batch_items": [
      {
        "name": "com.sn_cicd_spoke",
        "type": "Application",
        "version": "7.0.0",
        "state": "Installed",
        "install_date": "2020-08-31 15:30:01",
        "id": "c159b1e9db1c0010b5e3f6c5ae961903",
        "url": "https://instance.service-now.com/sys_batch_install_item.do?sys_id=c159b1e9db1c0010b5e3f6c5ae961903",
        "notes": ""
      },
      {
        "name": "Customization for CSM App1",
        "type": "Application",
        "version": "1.0.0",
        "state": "Installed",
        "install_date": "2020-08-31 15:32:01",
        "id": "e824b1e9db2d1001b5e3f6c5ae97d628",
        "url": "https://instance.service-now.com/sys_batch_install_item.do?sys_id=e824b1e9db2d1001b5e3f6c5ae97d628",
        "notes": ""
      }
    ]
  }
}
CI/CD - GET /sn_cicd/instance_scan/result/{progress_id}

Returns the current progress and status of the CI/CD instance scan function associated with a passed-in progress ID. Call this method after calling one of the other instance scan CI/CD endpoints such as /sn_cicd/instance_scan/full_scan or /sn_cicd/instance_scan/point_scan.

⚠  Note: This endpoint is part of the CICD Instance Scan Execution Service API and is used under the sn_cicd namespace.

**URL format**

Versioned URL: /api/sn_cicd/{api_version}/instance_scan/result/{progress_id}
Default URL: /api/sn_cicd/instance_scan/result/{progress_id}

**Supported request parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>progress_id</td>
<td>Unique identifier of the function for which to return progress information. This value is returned in the <code>links.progress.id</code> parameter of the endpoint that initiated the action, such as /sn_cicd/instance_scan/full_scan or /sn_cicd/instance_scan/point_scan. Data type: String</td>
</tr>
</tbody>
</table>
## Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

### Request body

The API accepts these JSON or XML elements in the request body.
Elements accepted in the request body

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Response body

The API returns these JSON or XML elements in the response body.

Elements returned in the response body

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message. Data type: String</td>
</tr>
<tr>
<td>links</td>
<td>All links and sys_ids associated with the response. Data type: Object</td>
</tr>
<tr>
<td>links.findings</td>
<td>Object that contains information about the instance scan findings.</td>
</tr>
<tr>
<td>links.findings.label</td>
<td>Additional information about the instance scan findings.</td>
</tr>
<tr>
<td>links.findings.url</td>
<td>URL to use to retrieve a list of records that violated the checks.</td>
</tr>
<tr>
<td>links.progress</td>
<td>Describes the progress link information. Data type: Object</td>
</tr>
<tr>
<td>links.progress.id</td>
<td>Unique identifier of the progress detail.</td>
</tr>
<tr>
<td>links.progress.url</td>
<td>URL to use to retrieve the progress details. Data type: String</td>
</tr>
<tr>
<td>links.results</td>
<td>Results information. Data type: Object</td>
</tr>
</tbody>
</table>
Elements returned in the response body (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;results&quot;: {</td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>links.results.id</td>
<td>Sys_id of the instance scan results record.</td>
</tr>
<tr>
<td>links.results.url</td>
<td>URL to use to obtain the results of the instance scan.</td>
</tr>
<tr>
<td>status</td>
<td>Numeric execution state. Used with status_label. Values:</td>
</tr>
<tr>
<td></td>
<td>• 0 (Pending)</td>
</tr>
<tr>
<td></td>
<td>• 1 (Running)</td>
</tr>
<tr>
<td></td>
<td>• 2 (Successful)</td>
</tr>
<tr>
<td></td>
<td>• 3 (Failed)</td>
</tr>
<tr>
<td></td>
<td>• 4 (Canceled)</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status_detail</td>
<td>Additional information about the current state.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status_label</td>
<td>Execution state description. Used with status. Values:</td>
</tr>
<tr>
<td></td>
<td>• Canceled</td>
</tr>
<tr>
<td></td>
<td>• Failed</td>
</tr>
<tr>
<td></td>
<td>• Pending</td>
</tr>
<tr>
<td></td>
<td>• Running</td>
</tr>
<tr>
<td></td>
<td>• Successful</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status_message</td>
<td>Additional information on why the operation failed. Empty if the operation is successful.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Example: Sample cURL request

curl 'instance.service-now.com/api/sn_cicd/instance_scan/result/a74c70a11b7800103d374087bc4bcb28' \
--request GET \
--header 'Accept: application/json' \
--user 'username':'password'

{
    "result": {
        "links": {
            "findings": {
                "url": "https://instance.service-now.com/api/now/table/scan_finding?sysparm_query%3Dresult%3D6f4cf8651b7800103d374087bc4bcb41%5EORDERBYsys_id",
                "label": "First 10,000 findings"
            },
            "progress": {
                "id": "a74c70a11b7800103d374087bc4bcb28",
                "url": "https://instance.service-now.com/api/sn_cicd/progress/a74c70a11b7800103d374087bc4bcb28"
            },
            "results": {
                "id": "a74c70a11b7800103d374087bc4bcb28",
                "url": "https://instance.service-now.com/api/sn_cicd/instance_scan/result/a74c70a11b7800103d374087bc4bcb28"
            }
        }
    },
    "status": "2",
    "status_label": "Successful",
    "status_message": "",
    "status_detail": "",
    "error": ""
}

Example: Sample Python request

# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.service-now.com/api/sn_cicd/result/a74c70a11b7800103d374087bc4bcb28'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept': 'application/json'}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "links": {
            "findings": {
                "url":
                  "https://instance.service-now.com/api/now/table/scan_finding?sysparm_query%3Dresult%3D6f4cf8651b7800103d374087bc4bcb41%5EORDERBYSysId",
                "label": "First 10,000 findings"
            },
            "progress": {
                "id": "a74c70a11b7800103d374087bc4bcb28",
                "url":
                  "https://instance.service-now.com/api/sn_cicd/progress/a74c70a11b7800103d374087bc4bcb28"
            }
        }
    }
}
CI/CD - GET /sn_cicd/progress/{progress_id}

Returns the current progress and status of the CI/CD function associated with a passed-in progress ID.

Call this method after calling one of the other CI/CD endpoints such as /sn_cicd/app_repo/publish, /sn_cicd/app_repo/install, or /sn_cicd/testsuite/run.

URL format

Versioned URL: /api/sn_cicd/{api_version}/progress/{progress_id}
Default URL: /api/sn_cicd/progress/{progress_id}

Supported request parameters

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>progress_id</td>
<td>Unique identifier of the function for which to return progress information. This value is returned in the links.progress.id parameter of the endpoint that initiated the action, such as /sn_cicd/app_repo/publish, /sn_cicd/app_repo/install, or /sn_cicd/testsuite/run.</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>409</td>
<td>Conflict. The requested item is not unique.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links</td>
<td>All links and sys_ids associated with the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;links&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {Object}</td>
</tr>
<tr>
<td>links.progress</td>
<td>Describes the progress link information.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>links.progress.id</td>
<td>Unique identifier to use to obtain the progress details for the operation.</td>
</tr>
<tr>
<td></td>
<td>Use this value when calling the endpoint / sn_cicd/progress/{progress_id}.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links.progress.url</td>
<td>URL to use to retrieve the progress details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links.results</td>
<td>Results information.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;results&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>links.results.id</td>
<td>Unique identifier of the results information. Use this value when calling the associated results endpoint. Data type: String</td>
</tr>
<tr>
<td>links.results.url</td>
<td>URL to use to obtain the results of the endpoint execution, such as results. Data type: String</td>
</tr>
<tr>
<td>links.source</td>
<td>Source information. Data type: Object</td>
</tr>
<tr>
<td>links.source.id</td>
<td>Unique identifier of the source record in the Execution Tracker [sys_execution_tracker] table. Data type: String</td>
</tr>
<tr>
<td>links.source.url</td>
<td>URL to use to obtain the source information in the Execution Tracker [sys_execution_tracker] table. Data type: String</td>
</tr>
<tr>
<td>percent_complete</td>
<td>Percentage of the request that is complete. Data type: Number</td>
</tr>
<tr>
<td>status</td>
<td>Numeric execution state. Used with status_label. Values:</td>
</tr>
<tr>
<td></td>
<td>• 0 (Pending)</td>
</tr>
<tr>
<td></td>
<td>• 1 (Running)</td>
</tr>
<tr>
<td></td>
<td>• 2 (Successful)</td>
</tr>
<tr>
<td></td>
<td>• 3 (Failed)</td>
</tr>
<tr>
<td></td>
<td>• 4 (Canceled)</td>
</tr>
<tr>
<td>status_detail</td>
<td>Additional information about the current state. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>status_label</td>
<td>Execution state description. Used with status. Values:</td>
</tr>
<tr>
<td></td>
<td>• Canceled</td>
</tr>
<tr>
<td></td>
<td>• Failed</td>
</tr>
<tr>
<td></td>
<td>• Pending</td>
</tr>
<tr>
<td></td>
<td>• Running</td>
</tr>
<tr>
<td></td>
<td>• Successful</td>
</tr>
<tr>
<td>status_message</td>
<td>Description of the current state.</td>
</tr>
</tbody>
</table>

Example: cURL request

```
curl 'instance.servicenow.com/api/sn_cicd/progress/a4fae8911bdc00103d374087bc4bcbbd' \
  --request GET \n  --header 'Accept: application/json' \n  --user 'username':'password'
```

Progress of an apply changes call (/sn_cicd/sc/apply_changes).

```
{
  "result": {
    "links": {
      "progress": {
        "id": "a4fae8911bdc00103d374087bc4bcbbd",
        "url": "https://instance.servicenow.com/api/sn_cicd/progress/a4fae8911bdc00103d374087bc4bcbbd",
      },
      "source": {
        "id": "59c4c4d11b5c00103d374087bc4bcbb6",
        "url": "https://instance.servicenow.com/api/now/table/sys_app/59c4c4d11b5c00103d374087bc4bcbb6"
      }
    },
    "status": "2",
    "status_label": "Successful",
    "status_message": "This operation succeeded",
  }
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
"status_detail": "Successfully applied commit 1f14e11a7dedcbfa194beb5875fcdab15ed8acdb from source control",
"error": "",
"percent_complete": 100
}
}

Example: cURL request

curl 'instance.servicenow.com/api/sn_cicd/progress/a4fae891bdc00103d374087bc4bcbdb' \
   --request GET \
   --header 'Accept: application/json' \
   --user 'username': 'password'

Progress of a test suite run (/sn_cicd/testsuite/run).

{
  "result": {
    "links": {
      "progress": {
        "id": "e891389d1b1040103d374087bc4bcb75",
        "url": "https://instance.servicenow.com/api/sn_cicd/progress/e891389d1b1040103d374087bc4bcb75",
      },
      "results": {
        "id": "2891389d1b1040103d374087bc4bcb09",
        "url": "https://instance.servicenow.com/api/sn_cicd/testsuite/results/2891389d1b1040103d374087bc4bcb09",
      }
    },
    "status": "2",
    "status_label": "Successful",
    "status_message": "Suite passed",
    "status_detail": "Suite passed",
    "error": "",
    "percent_complete": 100
  }
}

Example: Python request

# Install requests package for python
import requests
# Set the request parameters
url =
    'https://instance.servicenow.com/api/sn_cicd/progress/cb3a50a1dbff300caf55268dc9619fe'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Accept":"application/json"}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "links": {
            "progress": {
                "id": "cb3a50a1dbff300caf55268dc9619fe",
                "url":
                    "https://instance.service-now.com/api/sn_cicd/progress/cb3a50a1dbff300caf55268dc9619fe"
            },
            "results": {
                "id": "2891389d1b1040103d374087bc4bcb09",
                "url":
                    "https://instance.servicenow.com/api/sn_cicd/testsuite/results/2891389d1b1040103d374087bc4bcb09"
            }
        },
        "status": "2",
        "status_label": "Successful",
        "status_message": "Suite passed",
        "status_detail": "Suite passed",
        "error": ""
    }
}
CI/CD - GET /sn_cicd/testsuite/results/{result_id}

Returns the results of a test suite run based on a specified results identifier.

Call this endpoint after calling /sn_cicd/progress/{progress_id} to obtain the result of the executed test suite.

**URL format**

Versioned URL: /api/sn_cicd/{api_version}/testsuite/results/{result_id}

Default URL: /api/sn_cicd/testsuite/results/{result_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>result_id</td>
<td>Unique identifier of the test suite results to retrieve. The corresponding /sn_cicd/progress/{progress_id} endpoint call returns this information. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>409</td>
<td>Conflict. The requested item is not unique.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>
### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>child_suite_results</td>
<td>Results of nested test suites. The format of this content is the same as the parent test.</td>
</tr>
<tr>
<td>error</td>
<td>Error message.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links</td>
<td>Object that contains all links and sys_ids associated with the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>links.results</td>
<td>Object that contains the results information.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>links.results.id</td>
<td>Unique identifier of the results information. Use this value when calling the endpoint <code>/sn_cicd/testsuite/results/{result_id}</code>.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links.results.url</td>
<td>URL to use to obtain the results of the endpoint execution, such as results.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>percent_complete</td>
<td>Percentage of the request that is complete.</td>
</tr>
<tr>
<td>rolledup_test_error_count</td>
<td>Number of tests with errors.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>rolledup_test_failure_count</td>
<td>Number of tests that failed.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>rolledup_test_skip_count</td>
<td>Number of tests that were skipped.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>rolledup_test_success_count</td>
<td>Number of tests that ran successfully.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>status</td>
<td>Numeric execution state. Used with <code>status_label</code>, such as 0: Pending.</td>
</tr>
<tr>
<td></td>
<td>Values:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status_label</td>
<td>Execution state description. Used with <code>status</code>, such as 0: Pending. Values:</td>
</tr>
<tr>
<td></td>
<td>• Pending</td>
</tr>
<tr>
<td></td>
<td>• Running</td>
</tr>
<tr>
<td></td>
<td>• Successful</td>
</tr>
<tr>
<td></td>
<td>• Failed</td>
</tr>
<tr>
<td></td>
<td>• Canceled</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status_message</td>
<td>Description of the current state.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>test_suite_duration</td>
<td>Amount of time that it took to execute the test suite.</td>
</tr>
<tr>
<td></td>
<td>Unit: Seconds</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>test_suite_name</td>
<td>Name of the test suite.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>test_suite_status</td>
<td>State of the test suite.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Details:**
- **Name:** [Variable Name]
- **Description:** Additional information about the current state.
- **Data type:** String
- **status_detail**
- **status_label**
- **status_message**
- **test_suite_duration**
- **test_suite_name**
- **test_suite_status**
Example: cURL request

curl
  'instance.servicenow.com/api/sn_cicd/testsuite/results/2891389d1b1040103d374087bc4bcb09' \
  --request GET \
  --header 'Accept: application/json' \
  --user 'username':'password'

{
  "result": {
    "links": {
      "results": {
        "id": "2891389d1b1040103d374087bc4bcb09",
        "url": "https://instance.servicenow.com/sys_atf_test_suite_result.do?sys_id=2891389d1b1040103d374087bc4bcb09"
      },
    },
    "status": "2",
    "status_label": "Successful",
    "status_message": "",
    "status_detail": "",
    "error": "",
    "test_suite_status": "success",
    "test_suite_duration": "1 Second",
    "rolledup_test_success_count": 1,
    "rolledup_test_failure_count": 0,
    "rolledup_test_error_count": 0,
    "rolledup_test_skip_count": 0,
    "test_suite_name": "Quick Test",
    "child_suite_results": []
  }
}

Example: Python request

# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_cicd/testsuite/results/2891389d1b1040103d374087bc4bcb09'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Accept":"application/json"}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
  "result": {
    "links": {
      "results": {
        "id": "2891389d1b1040103d374087bc4bcb09",
        "url": "https://instance.servicenow.com/sys_atf_test_suite_result.do?sys_id=2891389d1b1040103d374087bc4bcb09"
      }
    },
    "status": "2",
    "status_label": "Successful",
    "status_message": "",
    "status_detail": "",
    "error": "",
    "test_suite_status": "success",
    "test_suite_duration": "1 Second",
    "rolledup_test_success_count": 1,
    "rolledup_test_failure_count": 0,
    "rolledup_test_error_count": 0,
    "rolledup_test_skip_count": 0,
    "test_suite_name": "Quick Test",
    "child_suite_results": []
  }
}
CI/CD - POST /sn_cicd/app/batch/install
Installs two or more packages in a single specified batch.

A batch can contain any combination of the following types:

• Application:
  ◦ Scoped applications in the ServiceNow Store that are not owned by your company
  ◦ Scoped or global applications in the application repository that are owned by your company
  ◦ Your application customizations for Store applications in the application repository

• ServiceNow plugin:
  ◦ Application customizations for scoped applications that were packaged in a plugin and were published to the application repository.

⚠️ Note: This endpoint uses an asynchronous response model. When you call an endpoint, it kicks off the associated function, such as starting the execution of the batch install plan. The endpoint then returns a response message that contains the execution status of the requested function, typically Pending, along with a unique identifier and link. Use this information to obtain additional information about the progress or results of the request.

URL format

Versioned URL: /api/sn_cicd/{api_version}/app/batch/install
Default URL: /api/sn_cicd/app/batch/install

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>
### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| name     | Required. User specified descriptive name for this batch request.  
Data type: String |
| notes    | User specified additional notes about the batch install plan.  
Data type: String  
Default: Null |
| packages | Required. JSON array, where each object specifies details of a package to install.  
Data type: Array |

```json
"packages": [{
  "id": "String",
  "load_demo_data": Boolean,
  "notes": "String",
  "requested_customization_version": "String",
  "requested_version": "String",
  "type": "String"
}]
```

| packages.id | Required. Sys_id of the application or identifier of the plugin to install.  
Data type: String |
|-------------|----------------------------------------------------------------------------------------------------------------------------------|
| packages.load_demo_data | Flag that indicates whether demo data is loaded when installing the package.  
Valid values: |
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
|                                           | • true: Demo data should be loaded.  
   • false: Demo data should not be loaded.  
   Data type: Boolean  
   Default: false |
| packages.notes                            | User specified notes about the package.  
   Data type: String  
   Default: Null |
| packages.requested_customization_version  | Version of the application customization to the store application or to the scoped ServiceNow plugin to install, such as 1.0.2 or 2.3.  
   Data type: String  
   Default: None. If this parameter is not set, then the system does not look for customizations for the application. |
| packages.requested_version                | Required if packages.type is set to application; ignored if set to plugin. Version of the package to install, such as 1.0.2 or 2.3.  
   Data type: String |
| packages.type                            | Required. Type of package.  
   Valid values:  
   • application  
   • plugin  
   Data type: String |
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>links</td>
<td>All links and sys_ids associated with the response. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;links&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: (Object),</td>
</tr>
<tr>
<td></td>
<td>&quot;results&quot;: (Object),</td>
</tr>
<tr>
<td></td>
<td>&quot;rollback&quot;: (Object)</td>
</tr>
<tr>
<td>links.progress</td>
<td>Describes the progress link information. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>links.progress.id</td>
<td>Unique identifier to use to obtain the progress details for the operation. Use this value when calling the endpoint / sn_cicd/progress/{progress_id}. Data type: String</td>
</tr>
<tr>
<td>links.progress.url</td>
<td>URL to use to retrieve the progress details. Data type: String</td>
</tr>
<tr>
<td>links.results</td>
<td>Describes the results link information. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;results&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>links.results.id</td>
<td>Sys_id of the results of the batch endpoint call. Use this value when calling the endpoint specified in links.results.url to query the batch install results. Data type: String</td>
</tr>
<tr>
<td>links.results.url</td>
<td>URL to use to retrieve the results of the batch install. Data type: String</td>
</tr>
<tr>
<td>links.rollback</td>
<td>Describes the batch install rollback information. Data type: Object</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;rollback&quot;: {</td>
<td>&quot;id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>links.rollback.id</td>
<td>Sys_id of the rollback details for the installed packages. Use this parameter to rollback the installation of the specified packages to their previous versions using the /sn_cicd/app/batch/rollback/{rollback_id} endpoint.</td>
</tr>
<tr>
<td>percent_complete</td>
<td>Percentage of the request that is complete.</td>
</tr>
<tr>
<td>status</td>
<td>Numeric execution state. Used with status_label. Values: • 0 (Pending) • 1 (Running) • 2 (Successful) • 3 (Failed) • 4 (Canceled)</td>
</tr>
<tr>
<td>status_detail</td>
<td>Additional information about the current state.</td>
</tr>
<tr>
<td>status_label</td>
<td>Execution state description. Used with status. Values: • Canceled • Failed • Pending • Running • Successful</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>status_message</td>
<td>Description of the current state.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

This example initiates a batch installation called Release 2.2 Deployment that contains three packages (one of each type) to install.

```bash
curl 'instance.servicenow.com/api/sn_cicd/app/batch/install \
  --request POST \
  --header 'Accept: application/json' \
  --user 'username': 'password' \
  --data { "name": "Release 2.2 Deployment",  
"packages": [ 
  {  
    "id": "syd_id_abcefghi",  
    "type": "application",  
    "load_demo_data": false,  
    "requested_version": "1.0.2",  
    "notes": "User specific text to describe this application install"  
  },  
  {  
    "id": "syd_id_defabcde",  
    "type": "application",  
    "requested_version": "1.0.0",  
    "requested_customization_version": "2.0.7",  
    "notes": "Customization for CSM App1"  
  },  
  {  
    "id": "com.glide.some.plugin",  
    "type": "plugin",  
    "load_demo_data": true,  
    "notes": "Plugin related notes"  
  }] 
} 

{  
"result": {  
  "links": {  
    "progress": {  
      "id": "c159b1e9db1c0010b5e3f6c5ae961903",  
      "url":  
      "https://instance.servicenow.com/api/sn_cicd/progress/c159b1e9db1c0010b5e3f6c5ae961903"  
    }  
  }  
}  
```
CI/CD - POST /sn_cicd/app/batch/rollback/{rollback_id}

Rolls back all of the packages associated with a specific batch installation to their previous versions.

This endpoint works in conjunction with the /sn_cicd/app/batch/install endpoint. The rollback_id is provided in the return results of this endpoint. When this endpoint is called, it rolls back all packages specified in the associated install; you cannot rollback individual packages. If there is no previous version, the endpoint uninstalls the associated package.

⚠️ Note: This endpoint uses an asynchronous response model. When you call an endpoint, it kicks off the associated function, such as starting the execution of the batch install plan. The endpoint then returns a response message that contains the execution status of the requested function, typically Pending, along with a unique identifier and link. Use this information to obtain additional information about the progress or results of the request.

**URL format**

Versioned URL: /api/sn_cicd/{api_version}/app/batch/rollback/{rollback_id}

Default URL: /api/sn_cicd/app/batch/rollback/{rollback_id}
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>rollback_id</td>
<td>Sys_id of the batch installation plan for which to rollback the installed packages to their previous versions. This value is returned in the links.rollback.id parameter of the /sn_cicd/app/batch/install endpoint. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links</td>
<td>All links and sys_ids associated with the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;links&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: { Object)</td>
</tr>
<tr>
<td>links.progress</td>
<td>Describes the progress link information.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>links.progress.id</td>
<td>Unique identifier to use to obtain the progress details for the operation. Use this value when calling the endpoint <code>/sn_cicd/progress/{progress_id}</code>. Data type: String</td>
</tr>
<tr>
<td>links.progress.url</td>
<td>URL to use to retrieve the progress details. Data type: String</td>
</tr>
<tr>
<td>percent_complete</td>
<td>Percentage of the request that is complete. Data type: Number</td>
</tr>
</tbody>
</table>
| status                | Numeric execution state. Used with `status_label`. Values:  
  • 0 (Pending)  
  • 1 (Running)  
  • 2 (Successful)  
  • 3 (Failed)  
  • 4 (Canceled)  
  Data type: String                                                                                                                             |
| status_detail         | Additional information about the current state. Data type: String                                                                                                                                           |
| status_label          | Execution state description. Used with `status`. Values:  
  • Canceled  
  • Failed  
  • Pending  
  • Running  
  • Successful  
  Data type: String                                                                                                                             |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status_message</td>
<td>Description of the current state.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

This example rolls back all the packages within the specified installation batch plan to their previous version.

```
curl 'instance.servicenow.com/api/sn_cicd/app/batch/rollback/a329f82e871da64c724ba21c82a764f2 \
--request POST \
--header 'Accept: application/json' \
--user 'username': 'password'

{
"result": {
  "links": {
    "progress": {
      "id": "c159b1e9db1c0010b5e3f6c5ae961903",
      "url": "https://instance.servicenow.com/api/sn_cicd/progress/c159b1e9db1c0010b5e3f6c5ae961903"
    },
    "status": "0",
    "status_label": "Pending",
    "status_message": "",
    "status_detail": "",
    "error": "",
    "percent_complete": 0
  }
}
```

**CI/CD - POST /sn_cicd/app_repo/install**

Installs the specified application from the application repository onto the instance making the endpoint call.

Using this endpoint you are able to install both unaltered and customized ServiceNow base system and ServiceNow Store applications, customized ServiceNow plugins, and your own ServiceNow applications.

The following outlines the parameters that you need to pass to implement specific use cases. All parameters that are not specified are not passed in and use the associated defaults.
• To only install the base application and not the corresponding application customizations.
  ◦ sys_id: Sys_id of the base application that has been customized.
  ◦ base_app_version: Version of the base application to install.

• To install a specified version of application customizations and automatically install the recommended base application version. The recommended base application version is the version of the base application that was installed on the instance at the time that the customizations were last published. Every version of application customizations has a corresponding recommended base application version.
  ◦ sys_id: Sys_id of the application that has been customized.
  ◦ auto_upgrade_base_app: true
  ◦ version: Version of the application customizations to install.

• To install the latest version of the application customizations and its corresponding recommended base application version.
  ◦ sys_id: Sys_id of the application that has been customized.
  ◦ auto_upgrade_base_app: true

• To install the latest version of the application customizations, but not upgrade the base application to the application customizations recommended base application version.
  ◦ sys_id: Sys_id of the application that has been customized.
  ◦ auto_upgrade_base_app: false

• To install application customizations along with the scoped ServiceNow plugin of the current release.
  ◦ scope: Scope of the plugin that has been customized OR
  ◦ sys_id: Sys_id of the application that has been customized. This is not the ID of the plugin.
  ◦ version: Version of the application customizations to install.

• To install application customizations for an already installed and activated scoped plugin.
  ◦ sys_id: Sys_id of the application that has been customized. This is not the ID of the plugin.
  ◦ auto_upgrade_base_app: false
  ◦ version: Version of the application customizations to install.
Since base plugins can have only one version per family release, the `auto_upgrade_base_app` and `base_app_version` parameters do not apply to application customization installs for plugins. To activate a plugin for the first time on an instance, use the `/sn_cicd/plugin/{plugin_id}/activate` endpoint instead.

ℹ️ **Note:** This endpoint uses an asynchronous response model. When you call an endpoint, it kicks off the associated function, such as starting the execution of the batch install plan. The endpoint then returns a response message that contains the execution status of the requested function, typically Pending, along with a unique identifier and link. Use this information to obtain additional information about the progress or results of the request.

**URL format**

**Versioned URL:** `/api/sn_cicd/{api_version}/app_repo/install`

**Default URL:** `/api/sn_cicd/app_repo/install`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, <code>v1</code> or <code>v2</code>. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>auto_upgrade_base_app</code></td>
<td>Only applicable if you intend to install application customizations and the base application was built on a version that is later than the currently installed version. Flag that indicates whether the associated base application should be automatically upgraded to a later version. Valid values:</td>
</tr>
</tbody>
</table>
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| auto_upgrade_base_app | • true: Automatically upgrade the associated base application.  
• false: Leave the base application as the version currently installed on the instance.  
Data type: Boolean  
Default: false |
| base_app_version     | Version of the base application to install. A base application is any third-party application that is available in the ServiceNow Store.  
If the application has no application customizations, or if the application has application customizations and you are only updating the base application, then enter the updated version of the application in this parameter. If you are updating application customizations, don’t pass this parameter, but rather pass the version parameter.  
Data type: String  
Default: Null |
| scope               | Required if sys_id is not specified. Scope name of the application, such as x_aah_custom_app. You can locate this value in the scope field in the Custom Application [sys_app] table or Store Application [sys_store_app] table.  
Data type: String |
| sys_id              | Required if scope is not specified. Sys_id of the application to install. You can locate this value in the Sys ID field in the Custom Application [sys_app] table or Store Application [sys_store_app] table.  
Data type: String |
| version             | Version of the application to install.  
Data type: String  
Default: If the base_app_version parameter is empty and the auto_upgrade_base_app parameter is set to... |
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>auto_upgrade_base_app</td>
<td>true, installs the latest application customizations and its recommended base application version (as long as it doesn't downgrade the base application.)</td>
</tr>
<tr>
<td></td>
<td>If the auto_upgrade_base_app parameter is false, installs the application customizations, but the base application is not upgraded.</td>
</tr>
<tr>
<td></td>
<td>If the base_app_version parameter is defined, only installs the base application.</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>409</td>
<td>Conflict. The requested item is not unique.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links</td>
<td>All links and sys_ids associated with the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;links&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>links.progress</td>
<td>Describes the progress link information.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>links.progress.id</td>
<td>Unique identifier to use to obtain the progress details for the operation. Use this value when calling the endpoint / sn_cicd/progress/{progress_id}.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>links.progress.url</td>
<td>URL to use to retrieve the progress details.</td>
</tr>
<tr>
<td>percent_complete</td>
<td>Percentage of the request that is complete.</td>
</tr>
<tr>
<td>rollback_version</td>
<td>If available, the previously installed version. If not available, null.</td>
</tr>
<tr>
<td>status</td>
<td>Numeric execution state. Used with <code>status_label</code>. Values:</td>
</tr>
<tr>
<td></td>
<td>• 0 (Pending)</td>
</tr>
<tr>
<td></td>
<td>• 1 (Running)</td>
</tr>
<tr>
<td></td>
<td>• 2 (Successful)</td>
</tr>
<tr>
<td></td>
<td>• 3 (Failed)</td>
</tr>
<tr>
<td></td>
<td>• 4 (Canceled)</td>
</tr>
<tr>
<td>status_detail</td>
<td>Additional information about the current state.</td>
</tr>
<tr>
<td>status_label</td>
<td>Execution state description. Used with <code>status</code>. Values:</td>
</tr>
<tr>
<td></td>
<td>• Canceled</td>
</tr>
<tr>
<td></td>
<td>• Failed</td>
</tr>
<tr>
<td></td>
<td>• Pending</td>
</tr>
<tr>
<td></td>
<td>• Running</td>
</tr>
<tr>
<td></td>
<td>• Successful</td>
</tr>
<tr>
<td>status_message</td>
<td>Description of the current state.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: cURL request

curl
  'instance.servicenow.com/api/sn_cicd/app_repo/install?sys_id=2d146921dbd80010caf55268dc9619d8' \ 
--request POST \ 
--header 'Accept: application/json' \ 
--user 'username': 'password'

{
  "result": {
    "links": {
      "progress": {
        "id": "c159b1e9db1c0010b5e3f6c5ae961903",
        "url": "https://instance.servicenow.com/api/sn_cicd/progress/c159b1e9db1c0010b5e3f6c5ae961903"
      }
    },
    "status": "0",
    "status_label": "Pending",
    "status_message": "",
    "status_detail": "",
    "error": "",
    "percent_complete": 0,
    "rollback_version": "1.1.0"
  }
}

Example: Python request

# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_cicd/app_repo/install?sys_id=2d146921dbd80010caf55268dc9619d8'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/json'}
CI/CD - POST /sn_cicd/app_repo/publish

Publishes the specified application and all of its artifacts to the application repository.

**URL format**

Versioned URL: /api/sn_cicd/{api_version}/app_repo/publish

Default URL: /api/sn_cicd/app_repo/publish
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dev_notes</td>
<td>Developer notes to store with the application.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>scope</td>
<td>Required if sys_id is not specified. The scope of the application. You can locate this value in the scope field in the Custom Application [sys_app] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Required if scope is not specified. The sys_id of the application to store in the repository. You can locate this value in the Sys ID field in the Custom Application [sys_app] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>version</td>
<td>Version under which to store the application.</td>
</tr>
<tr>
<td></td>
<td>If the version number is passed, the publish process uses that version and updates the local application version if different. If the version number is not passed, the publish process uses the current version of the local application and will fail if that version already exists in application repository.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>409</td>
<td>Conflict. The requested item is not unique.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>error</strong></td>
<td>Error message. Data type: String</td>
</tr>
<tr>
<td><strong>links</strong></td>
<td>All links and sys_ids associated with the response. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;links&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {Object}</td>
</tr>
<tr>
<td><strong>links.progress</strong></td>
<td>Describes the progress link information. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td><strong>links.progress.id</strong></td>
<td>Unique identifier to use to obtain the progress details for the operation. Use this value when calling the endpoint / sn_cicd/progress/{progress_id}. Data type: String</td>
</tr>
<tr>
<td><strong>links.progress.url</strong></td>
<td>URL to use to retrieve the progress details. Data type: String</td>
</tr>
<tr>
<td><strong>percent_complete</strong></td>
<td>Percentage of the request that is complete. Data type: Number</td>
</tr>
<tr>
<td><strong>status</strong></td>
<td>Additional information about the current state. Data type: String</td>
</tr>
<tr>
<td><strong>status_detail</strong></td>
<td>Additional information about the current state. Data type: String</td>
</tr>
<tr>
<td><strong>status_label</strong></td>
<td>Execution state description. Used with <strong>status</strong>. Values:</td>
</tr>
<tr>
<td></td>
<td>• Cancelled</td>
</tr>
<tr>
<td></td>
<td>• Failed</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>• Pending</td>
</tr>
<tr>
<td></td>
<td>• Running</td>
</tr>
<tr>
<td></td>
<td>• Successful</td>
</tr>
<tr>
<td>status_message</td>
<td>Description of the current state.</td>
</tr>
</tbody>
</table>

Data type: String

Example: cURL request

curl
'instance.servicenow.com/api/sn_cicd/app_repo/publish?sys_id=4dd9686d1b9800103d374087bc4bc8b3d' \  
--request POST \  
--header 'Accept: application/json' \  
--user 'username':'password'

{  
"result": {  
"links": {  
   "progress": {  
      "id": "d174f8e11bd800103d374087bc4bcb9",  
      "url":  
      "https://instance.service-now.com/api/sn_cicd/progress/d174f8e11bd800103d374087bc4bcb9"  
   },  
   "status": "0",  
   "status_label": "Pending",  
   "status_message": "",  
   "status_detail": "",  
   "error": "",  
   "percent_complete": 0  
  }  
}

Example: Python request

# Install requests package for python
import requests

# Set the request parameters
```python
url = 'https://instance.servicenow.com/api/sn_cicd/app_repo/publish?sys_id=4dd9686d1b9800103d374087bc4bcb3d'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/json'}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
"result": {
"links": {
"progress": {
"id": "d174f8e11bd800103d374087bc4bcb9",
"url": "https://instance.service-now.com/api/sn_cicd/progress/d174f8e11bd800103d374087bc4bcbd9"
}
},
"status": "0",
"status_label": "Pending",
"status_message": "",
"status_detail": "",
"error": "",
"percent_complete": 0
}
}
```

**CI/CD - POST /sn_cicd/app_repo/rollback**

Initiates a rollback of a specified application to a specified version.
Note: The rollback functionality is only available after the installation is complete.

URL format

Versioned URL: /api/sn_cicd/{api_version}/app_repo/rollback

Default URL: /api/sn_cicd/app_repo/rollback

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>api_version</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>scope</td>
</tr>
<tr>
<td>sys_id</td>
</tr>
<tr>
<td>version</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the <code>sn_cicd.sys_ci_automation</code> role.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>
# Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message. Data type: String</td>
</tr>
<tr>
<td>links</td>
<td>All links and sys_ids associated with the response. Data type: Object</td>
</tr>
<tr>
<td>links.progress</td>
<td>Describes the progress link information. Data type: Object</td>
</tr>
<tr>
<td>links.progress.id</td>
<td>Unique identifier to use to obtain the progress details for the operation. Use this value when calling the endpoint /sn_cicd/progress/{progress_id}. Data type: String</td>
</tr>
<tr>
<td>links.progress.url</td>
<td>URL to use to retrieve the progress details. Data type: String</td>
</tr>
<tr>
<td>percent_complete</td>
<td>Percentage of the request that is complete. Data type: Number</td>
</tr>
<tr>
<td>status</td>
<td>Numeric execution state. Used with status_label. Values:</td>
</tr>
<tr>
<td></td>
<td>• 0 (Pending)</td>
</tr>
<tr>
<td></td>
<td>• 1 (Running)</td>
</tr>
<tr>
<td></td>
<td>• 2 (Successful)</td>
</tr>
<tr>
<td></td>
<td>• 3 (Failed)</td>
</tr>
<tr>
<td></td>
<td>• 4 (Canceled)</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>status_detail</td>
<td>Additional information about the current state.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status_label</td>
<td>Execution state description. Used with status.</td>
</tr>
<tr>
<td></td>
<td>Values:</td>
</tr>
<tr>
<td></td>
<td>• Canceled</td>
</tr>
<tr>
<td></td>
<td>• Failed</td>
</tr>
<tr>
<td></td>
<td>• Pending</td>
</tr>
<tr>
<td></td>
<td>• Running</td>
</tr>
<tr>
<td></td>
<td>• Successful</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status_message</td>
<td>Description of the current state.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

curl
```
'instance.servicenow.com/api/sn_cicd/app_repo/rollback?sys_id=2d146921dbd80010caf55268dc9619d8&version=1.1.0' \ 
--request POST \ 
--header 'Accept: application/json' \ 
--user 'username':'password'
```

```
|
"result": {
  "links": {
    "progress": {
      "id": "f7db1eddb1c0010b5e3f6c5ae9619c1",
      "url": "https://instance.service-now.com/api/sn_cicd/progress/f7db1eddb1c0010b5e3f6c5ae9619c1"
    }
  },
  "status": "0",
  "status_label": "Pending",
  "status_message": "",
  "status_detail": "",
  "error": "",
  "percent_complete": 0
```
}
}

Example: Python request
# Install requests package for python
import requests

# Set the request parameters
url =
'https://instance.servicenow.com/api/sn_cicd/app_repo/rollback?sys_id=2d146921dbd80010caf5
5268dc9619d8&version=1.1.0'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Accept":"application/json"}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
response.json())
exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)
{
"result": {
"links": {
"progress": {
"id": "f7ddb1eddb1c0010b5e3f6c5ae9619c1",
"url":
"https://instance.service-now.com/api/sn_cicd/progress/f7ddb1eddb1c0010b5e3f6c5ae9619c1"
}
},
"status": "0",
"status_label": "Pending",

6203

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


CI/CD - POST /sn_cicd/instance_scan/full_scan

Runs all active checks present in your instance.

You can run checks that are provided with Instance Scan or you can create your own checks. For more information, see Instance Scan.

ℹ️ Note: This endpoint is part of the CICD Instance Scan Execution Service API and is used under the sn_cicd namespace.

### URL format

**Versioned URL:** /api/sn_cicd/{api_version}/instance_scan/full_scan

**Default URL:** /api/sn_cicd/instance_scan/full_scan

### Supported request parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml.</td>
</tr>
</tbody>
</table>
Request headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default:</td>
<td>application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

Request body

The API accepts these JSON or XML elements in the request body.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Response body

The API returns these JSON or XML elements in the response body.
### Elements returned in the response body

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links</td>
<td>All links and sys_ids associated with the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;links&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {Object}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>links.progress</td>
<td>Describes the progress link information.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>links.progress.id</td>
<td>Unique identifier of the progress detail. Use this value when calling the</td>
</tr>
<tr>
<td></td>
<td>endpoint /sn_cicd/instance_scan/result/{progress_id}.</td>
</tr>
<tr>
<td>links.progress.url</td>
<td>URL to use to retrieve the progress details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>Numeric execution state. Used with status_label.</td>
</tr>
<tr>
<td></td>
<td>Values:</td>
</tr>
<tr>
<td></td>
<td>• 0 (Pending)</td>
</tr>
<tr>
<td></td>
<td>• 1 (Running)</td>
</tr>
<tr>
<td></td>
<td>• 2 (Successful)</td>
</tr>
<tr>
<td></td>
<td>• 3 (Failed)</td>
</tr>
<tr>
<td></td>
<td>• 4 (Canceled)</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status_detail</td>
<td>Additional information about the current state.</td>
</tr>
<tr>
<td>status_label</td>
<td>Execution state description. Used with status.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Elements returned in the response body (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Values:</td>
<td></td>
</tr>
<tr>
<td>• Canceled</td>
<td></td>
</tr>
<tr>
<td>• Failed</td>
<td></td>
</tr>
<tr>
<td>• Pending</td>
<td></td>
</tr>
<tr>
<td>• Running</td>
<td></td>
</tr>
<tr>
<td>• Successful</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

status_message  Additional information on why the operation failed. Empty if the operation is successful.
Data type: String

Example: Sample cURL request

```
curl 'instance.service-now.com/api/sn_cicd/instance_scan/full_scan' \   
--request POST \ 
--header 'Accept: application/json' \ 
--user 'username':'password'

{
"result": {
  "links": {
    "progress": {
      "id": "a4fae8911bdc00103d374087bc4bcbbd",
      "url": "https://instance.service-now.com/api/sn_cicd/progress/a4fae8911bdc00103d374087bc4bcbbd"
    },
    "status": "0",
    "status_label": "Pending",
    "status_message": ",",
    "status_detail": ",",
    "error": ""
  }
}
```
Example: Sample Python request

```python
# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.service-now.com/api/sn_cicd/instance_scan/full_scan'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept':"application/json"}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)
```

```json
{
    "result": {
        "links": {
            "progress": {
                "id": "a4fae8911bdc00103d374087bc4bcbbd",
                "url": "https://instance.service-now.com/api/sn_cicd/progress/a4fae8911bdc00103d374087bc4bcbbd"
            }
        },
        "status": "0",
        "status_label": "Pending",
        "status_message": "",
        "status_detail": "",
        "error": ""
    }
}
```
CI/CD - POST /sn_cicd/instance_scan/point_scan

Executes all applicable checks against a specified record.

For example, if you execute a point scan against a Catalog Item, only the checks applicable to the Catalog Item table run, and only the specified record is scanned.

You can run checks that are provided with Instance Scan or you can create your own checks. For more information, see Instance Scan.

⚠️ **Note:** This endpoint is part of the CICD Instance Scan Execution Service API and is used under the sn_cicd namespace.

**URL format**

Versioned URL: /api/sn_cicd/{api_version}/instance_scan/point_scan

Default URL: /api/sn_cicd/instance_scan/point_scan

**Supported request parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>target_table</td>
<td>Required. Name of the table in which the record to scan resides, such as &quot;Incident.&quot; Data type: String</td>
</tr>
<tr>
<td>target_sys_id</td>
<td>Required. The sys_id of the record within the specified table to scan. Data type: String</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. Invalid or empty query parameters.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

Request body

The API accepts these JSON or XML elements in the request body.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
**Response body**

The API returns these JSON or XML elements in the response body.

### Elements returned in the response body

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message. Data type: String</td>
</tr>
<tr>
<td>links</td>
<td>All links and sys_ids associated with the response. Data type: Object</td>
</tr>
<tr>
<td>links.progress</td>
<td>Describes the progress link information. Data type: Object</td>
</tr>
<tr>
<td>links.progress.id</td>
<td>Unique identifier of the progress detail. Use this value when calling the endpoint /sn_cicd/instance_scan/result/{progress_id}.</td>
</tr>
<tr>
<td>links.progress.url</td>
<td>URL to use to retrieve the progress details. Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>Numeric execution state. Used with status_label. Values:</td>
</tr>
<tr>
<td>status_detail</td>
<td>Additional information about the current state. Data type: String</td>
</tr>
</tbody>
</table>

Values:
- 0 (Pending)
- 1 (Running)
- 2 (Successful)
- 3 (Failed)
- 4 (Canceled)
Elements returned in the response body (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| status_label    | Execution state description. Used with `status`. Values:  
• Canceled  
• Failed  
• Pending  
• Running  
• Successful  
Data type: String |
| status_message  | Additional information on why the operation failed. Empty if the operation is successful.  
Data type: String |

Example: Sample cURL request

```bash
curl  
'https://instance.service-now.com/api/sn_cicd/instance_scan/point_scan?target_table=incident&target_sys_id=2d146921db80010caf55268dc9619d8' \  
--request POST \  
--header 'Accept: application/json' \  
--user 'username': 'password'

{
  "result": {
    "links": {
      "progress": {
        "id": "a4fae8911bdc00103d374087bc4bcbbd",
        "url": "https://instance.service-now.com/api/sn_cicd/progress/a4fae8911bdc00103d374087bc4bcbbd"
      }
    },
    "status": "0",
    "status_label": "Pending",
    "status_message": "",
    "status_detail": "",
    "error": ""
  }
}
```
Example: Sample Python request

```python
# Need to install requests package for python
import requests

# Set the request parameters
url = 
  'https://instance.service-now.com/api/sn_cicd/instance_scan/point_scan?target_table=incident&target_sys_id=2d146921dbd80010caf55268dc9619d8'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {"Accept":"application/json"}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)
```

```json
{
  "result": {
    "links": {
      "progress": {
        "id": "a4fae8911bd00103d374087bc4bcbbd",
        "url": "https://instance.service-now.com/api/sn_cicd/progress/a4fae8911bd00103d374087bc4bcbbd"
      },
      "status": "0",
      "status_label": "Pending",
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
CI/CD - POST /sn_cicd/instance_scan/suite_scan/combo/{combo_sys_id}

Runs a scan using a suite and target (scoped apps or update sets) that have already run previously.

Before calling this endpoint, you must call the /sn_cicd/instance_scan/suite_scan/{suite_sys_id}/scoped_apps or /sn_cicd/instance_scan/suite_scan/{suite_sys_id}/update_sets endpoints, or execute a suite scan in the UI.

ℹ️ Note: This endpoint is part of the CICD Instance Scan Execution Service API and is used under the sn_cicd namespace.

### URL format

Versioned URL: /api/sn_cicd/{api_version}/instance_scan/suite_scan/combo/{combo_sys_id}

Default URL: /api/sn_cicd/instance_scan/suite_scan/combo/{combo_sys_id}

### Supported request parameters

#### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>combo_sys_id</td>
<td>The sys_id of the scan to run. The scan must be an existing scan that you want to re-run. Located in the Combo [scan_combo] table.</td>
</tr>
</tbody>
</table>

#### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>
# Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message. Data type: String</td>
</tr>
<tr>
<td>links</td>
<td>All links and sys_ids associated with the response. Data type: Object</td>
</tr>
<tr>
<td>links.progress</td>
<td>Describes the progress link information. Data type: Object</td>
</tr>
<tr>
<td>links.progress.id</td>
<td>Unique identifier of the progress detail. Use this value when calling the endpoint <code>/sn_cicd/instance_scan/result/{progress_id}</code>.</td>
</tr>
<tr>
<td>links.progress.url</td>
<td>URL to use to retrieve the progress details. Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>Numeric execution state. Used with <code>status_label</code>. Values:</td>
</tr>
<tr>
<td></td>
<td>• 0 (Pending)</td>
</tr>
<tr>
<td></td>
<td>• 1 (Running)</td>
</tr>
<tr>
<td></td>
<td>• 2 (Successful)</td>
</tr>
<tr>
<td></td>
<td>• 3 (Failed)</td>
</tr>
<tr>
<td></td>
<td>• 4 (Canceled)</td>
</tr>
<tr>
<td>status_detail</td>
<td>Additional information about the current state. Data type: String</td>
</tr>
<tr>
<td>status_label</td>
<td>Execution state description. Used with <code>status</code>.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Values:</td>
<td></td>
</tr>
<tr>
<td>•Canceled</td>
<td></td>
</tr>
<tr>
<td>•Failed</td>
<td></td>
</tr>
<tr>
<td>•Pending</td>
<td></td>
</tr>
<tr>
<td>•Running</td>
<td></td>
</tr>
<tr>
<td>•Successful</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

status_message | Additional information on why the operation failed. Empty if the operation is successful.
Data type: String

Example: cURL request

```
curl  
  "https://instance.servicenow.com/api/sn_cicd/instance_scan/suite_scan/combo/245d29cc1bd32010178aed72604bcb4b" \  
--request POST \  
--header "Accept:application/json" \  
--user 'username':'password'
```

Response body - success.

```
{
  "result": {
    "links": {
      "progress": {
        "id": "c08be5c71b1b2010178aed72604bcb6e",
        "url": "https://instance.servicenow.com/api/sn_cicd/progress/c08be5c71b1b2010178aed72604bcb6e"
      }
    },
    "status": "0",
    "status_label": "Pending",
    "status_message": "",
    "status_detail": "",
    "error": ""
  }
}
```
CI/CD - POST /sn_cicd/instance_scan/suite_scan/{suite_sys_id}/scoped_apps

Runs all active checks inside a suite against a list of scoped apps.

⚠️ Note: This endpoint is part of the CICD Instance Scan Execution Service API and is used under the sn_cicd namespace.

**URL format**

Versioned URL: /api/sn_cicd/{api_version}/instance_scan/suite_scan/{suite_sys_id}/scoped_apps

Default URL: /api/sn_cicd/instance_scan/suite_scan/{suite_sys_id}/scoped_apps

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>suite_sys_id</td>
<td>The sys_id of the suite to run. Located in the Suite [scan_check_suite] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app_scope_sys_ids</td>
<td>Required. A list of sys_ids of the scoped applications to scan. Located in the Application [sys_scope] table. The array can't be an empty array. Data type: Array</td>
</tr>
</tbody>
</table>

app_scope_sys_ids: ["String", "String", "String"]

© 2021 ServiceNow, Inc. All rights reserved.
Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes
The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>
### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message. Data type: String</td>
</tr>
<tr>
<td>links</td>
<td>All links and sys_ids associated with the response. Data type: Object</td>
</tr>
</tbody>
</table>
|                  | ```json
| links.progress  | 
|                  | ```json
| status           | Numeric execution state. Used with status_label. Values:                   |
|                  | • 0 (Pending)                                                               |
|                  | • 1 (Running)                                                              |
|                  | • 2 (Successful)                                                           |
|                  | • 3 (Failed)                                                               |
|                  | • 4 (Canceled)                                                             |
| status_detail    | Additional information about the current state.                            |
| status_label     | Execution state description. Used with status.                             |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Values:</td>
</tr>
<tr>
<td></td>
<td>• Canceled</td>
</tr>
<tr>
<td></td>
<td>• Failed</td>
</tr>
<tr>
<td></td>
<td>• Pending</td>
</tr>
<tr>
<td></td>
<td>• Running</td>
</tr>
<tr>
<td></td>
<td>• Successful</td>
</tr>
<tr>
<td>status_message</td>
<td>Additional information on why the operation failed. Empty if the operation is successful.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

This example runs all active checks inside a suite against one scoped application.

```
curl
  "https://instance.servicenow.com/api/sn_cicd/instance_scan/suite_scan/833655cc1b94101046e87733cd4bcb4e/scoped_apps" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --data '{app_scope_sys_ids: ["078e8020950f12a215a92e8ad9b346e0"]}' \
  --user 'username':'password'
```

**Response body - success.**

```json
{
  "result": {
    "links": {
      "progress": {
        "id": "147fc14b1b572010178aed72604bcbca",
        "url": "https://instance.servicenow.com/api/sn_cicd/progress/147fc14b1b572010178aed72604bcbca"
      }
    },
    "status": "O",
    "status_label": "Pending",
    "status_message": "",
    "status_detail": "",
  }
}
```
CI/CD - POST /sn_cicd/instance_scan/suite_scan/{suite_sys_id}/update_sets

Runs all active checks inside a suite against a list of update sets.

⚠️ **Note:** This endpoint is part of the CICD Instance Scan Execution Service API and is used under the sn_cicd namespace.

**URL format**

Versioned URL: /api/sn_cicd/{api_version}/instance_scan/suite_scan/
(suite_sys_id)/update_sets

Default URL: /api/sn_cicd/instance_scan/suite_scan/{suite_sys_id}/update_sets

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>suite_sys_id</td>
<td>The sys_id of the suite to run. Located in the Suite [scan_check_suite] table. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>update_set_sys_ids</td>
<td>Required. A list of sys_ids of the update sets to scan. Located in the Update Set [sys_update_set] table. The array can't be an empty array.</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>update_set_sys_ids: [&quot;String&quot;, &quot;String&quot;, &quot;String&quot;]</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| error      | Error message.  
Data type: String |
| links      | All links and sys_ids associated with the response.  
Data type: Object |
|            | "links": {  
|            |   "progress": {Object}  
|            | } |
| links.progress | Describes the progress link information.  
Data type: Object |
|            | "progress": {  
|            |   "id": "String",  
|            |   "url": "String"  
|            | } |
| links.progress.id | Unique identifier of the progress detail. Use this value when calling the endpoint `/sn_cicd/instance_scan/result/{progress_id}`. |
| links.progress.url | URL to use to retrieve the progress details.  
Data type: String |
| status     | Numeric execution state. Used with `status_label`.  
Values:  
- 0 (Pending)  
- 1 (Running)  
- 2 (Successful) |
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• 3 (Failed)</td>
</tr>
<tr>
<td></td>
<td>• 4 (Canceled)</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status_detail</td>
<td>Additional information about the current state.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status_label</td>
<td>Execution state description. Used with \textit{status}.</td>
</tr>
<tr>
<td></td>
<td>Values:</td>
</tr>
<tr>
<td></td>
<td>• Canceled</td>
</tr>
<tr>
<td></td>
<td>• Failed</td>
</tr>
<tr>
<td></td>
<td>• Pending</td>
</tr>
<tr>
<td></td>
<td>• Running</td>
</tr>
<tr>
<td></td>
<td>• Successful</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status_message</td>
<td>Additional information on why the operation failed. Empty if</td>
</tr>
<tr>
<td></td>
<td>the operation is successful.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

\textbf{Example: cURL request}

This example runs all active checks inside a suite against one update set.

```
curl
  "https://instance.servicenow.com/api/sn_cicd/instance_scan/suite_scan/833655cc1b94101046e87733cd4bc4e/update_sets" \ 
  --request POST \ 
  --header "Accept:application/json" \ 
  --header "Content-Type:application/json" \ 
  --data "{update_set_sys_ids: ["7abef63c1b572010178aed72604bcbfd"])" \ 
  --user 'username':'password'
```

Response body - success.

```
{
   "result": {
      "links": {
         "progress": {
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
CI/CD - POST /sn_cicd/plugin/{plugin_id}/activate
Activates the specified plugin.

URL format
Versioned URL: /api/sn_cicd/{api_version}/plugin/{plugin_id}/activate
Default URL: /api/sn_cicd/plugin/{plugin_id}/activate

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
<td>String</td>
</tr>
<tr>
<td>plugin_id</td>
<td>Unique identifier of the plugin. You can locate this identifier on the Plugins page within the card of the desired plugin; identified with the name &quot;ID&quot;.</td>
<td>String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>409</td>
<td>Conflict. The requested item is not unique.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links</td>
<td>All links and sys_ids associated with the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;links&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {Object}</td>
</tr>
<tr>
<td>links.progress</td>
<td>Describes the progress link information.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>links.progress.id</td>
<td>Unique identifier to use to obtain the progress details for the operation. Use this value when calling the endpoint /sn_cicd/progress/{progress_id}.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links.progress.url</td>
<td>URL to use to retrieve the progress details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links.results</td>
<td>Results information.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;results&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>&quot;url&quot;: &quot;String&quot;</strong></td>
</tr>
<tr>
<td>links.results.id</td>
<td>Unique identifier of the results information. Use this value when calling the associated results endpoint.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links.results.url</td>
<td>URL to use to obtain the results of the endpoint execution, such as results.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>percent_complete</td>
<td>Percentage of the request that is complete.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>status</td>
<td>Numeric execution state. Used with <strong>status_label</strong>.</td>
</tr>
<tr>
<td></td>
<td>Values:</td>
</tr>
<tr>
<td></td>
<td>• 0 (Pending)</td>
</tr>
<tr>
<td></td>
<td>• 1 (Running)</td>
</tr>
<tr>
<td></td>
<td>• 2 (Successful)</td>
</tr>
<tr>
<td></td>
<td>• 3 (Failed)</td>
</tr>
<tr>
<td></td>
<td>• 4 (Canceled)</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status_detail</td>
<td>Additional information about the current state.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status_label</td>
<td>Execution state description. Used with <strong>status</strong>.</td>
</tr>
<tr>
<td></td>
<td>Values:</td>
</tr>
<tr>
<td></td>
<td>• Canceled</td>
</tr>
<tr>
<td></td>
<td>• Failed</td>
</tr>
<tr>
<td></td>
<td>• Pending</td>
</tr>
<tr>
<td></td>
<td>• Running</td>
</tr>
<tr>
<td></td>
<td>• Successful</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status_message</td>
<td>Description of the current state.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

curl 'instance.servicenow.com/api/sn_cicd/plugin/com.glide.web_service_aggregate/activate'
   --request POST \
   --header 'Accept: application/json' \
   --user 'username':'password'

{
   "result": {
      "links": {
         "progress": {
            "id": "5b3f052db580010caf55268dc961963",
            "url": "https://instance.service-now.com/api/sn_cicd/progress/5b3f052db580010caf55268dc961963"
         },
         "status": "0",
         "status_label": "Pending",
         "status_message": "",
         "status_detail": "",
         "error": "",
         "percent_complete": 0
      }
   }
}

Example: Python request

# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_cicd/plugin/com.glide.web_service_aggregate/activate'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept':'application/json'}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)
# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "links": {
            "progress": {
                "id": "5b3f052ddb580010caf55268dc961963",
                "url": "https://instance.service-now.com/api/sn_cicd/progress/5b3f052ddb580010caf55268dc961963"
            }
        },
        "status": "0",
        "status_label": "Pending",
        "status_message": "",
        "status_detail": "",
        "error": "",
        "percent_complete": 0
    }
}

CI/CD - POST /sn_cicd/plugin/{plugin_id}/rollback

Rolls back the specified plugin to the previous installed version. If no prior version has been installed, returns an error.

URL format

Versioned URL: /api/sn_cicd/{api_version}plugin/{plugin_id}/rollback
Default URL: /api/sn_cicd/plugin/{plugin_id}/rollback
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>plugin_id</td>
<td>Unique identifier of the plugin. You can locate this identifier on the Plugins page within the card of the desired plugin; identified with the name &quot;ID&quot;. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>409</td>
<td>Conflict. The requested item is not unique.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message. Data type: String</td>
</tr>
<tr>
<td>links</td>
<td>All links and sys_ids associated with the response. Data type: Object</td>
</tr>
<tr>
<td>links.progress</td>
<td>Describes the progress link information.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>links.progress.id</td>
<td>Unique identifier to use to obtain the progress details for the operation. Use this value when calling the endpoint /sn_cicd/progress/{progress_id}. Data type: String</td>
</tr>
<tr>
<td>links.progress.url</td>
<td>URL to use to retrieve the progress details. Data type: String</td>
</tr>
<tr>
<td>percent_complete</td>
<td>Percentage of the request that is complete. Data type: Number</td>
</tr>
<tr>
<td>status</td>
<td>Numeric execution state. Used with status_label. Values:</td>
</tr>
<tr>
<td></td>
<td>• 0 (Pending)</td>
</tr>
<tr>
<td></td>
<td>• 1 (Running)</td>
</tr>
<tr>
<td></td>
<td>• 2 (Successful)</td>
</tr>
<tr>
<td></td>
<td>• 3 (Failed)</td>
</tr>
<tr>
<td></td>
<td>• 4 (Canceled)</td>
</tr>
<tr>
<td>status_detail</td>
<td>Additional information about the current state. Data type: String</td>
</tr>
<tr>
<td>status_label</td>
<td>Execution state description. Used with status. Values:</td>
</tr>
<tr>
<td></td>
<td>• Canceled</td>
</tr>
<tr>
<td></td>
<td>• Failed</td>
</tr>
<tr>
<td></td>
<td>• Pending</td>
</tr>
<tr>
<td></td>
<td>• Running</td>
</tr>
<tr>
<td></td>
<td>• Successful</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>status_message</td>
<td>Description of the current state. Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

```
curl 'instance.servicenow.com/api/sn_cicd/plugin/com.glide.web_service_aggregate/rollback'
  
  --request POST
  
  --header 'Accept: application/json'
  
  --user 'username':'password'

{
  "result": {
    "links": {
      "progress": {
        "id": "6185dd61db980010caf55268dc961916",
        "url":
            "https://instance.service-now.com/api/sn_cicd/progress/6185dd61db980010caf55268dc961916"
      },
      "status": "0",
      "status_label": "Pending",
      "status_message": "",
      "status_detail": "",
      "error": "",
      "percent_complete": 0
    }
  }
}
```

Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url =
    'https://instance.servicenow.com/api/sn_cicd/plugin/com.glide.web_service_aggregate/rollback'

# Set the user credentials
user = 'username'
```
pwd = 'password'

# Set the proper headers
headers = {"Accept":"application/json"}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
  "result": {
    "links": {
      "progress": {
        "id": "6185dd61db980010caf55268dc961916",
        "url": "https://instance.service-now.com/api/sn_cicd/progress/6185dd61db980010caf55268dc961916"
      }
    },
    "status": "0",
    "status_label": "Pending",
    "status_message": "",
    "status_detail": "",
    "error": "",
    "percent_complete": 0
  }
}

**CI/CD - POST /sn_cicd/sc/apply_changes**

Starts applying changes from a remote source control to a specified local application or application-customization.

**URL format**

Versioned URL: /api/sn_cicd/{api_version}/sc/apply_changes
**Default URL:** /api/sn_cicd/sc/apply_changes

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>auto_upgrade_base_app</td>
<td>Only applicable when changes are applied for app-customization and the latest commit on the Git repository is built on a version that is later than that of the base application that is currently installed on the local instance. Flag that indicates whether the system should auto upgrade the base application to a later version. Data type: Boolean Default: true</td>
</tr>
<tr>
<td>branch_name</td>
<td>Name of the branch in the source control system from which to acquire the application. Data type: String Default: Default branch specified on the source control system.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Required if scope is not specified. The sys_id of the application for which to apply the changes. Data type: String</td>
</tr>
<tr>
<td>scope</td>
<td>Required if sys_id is not specified. The scope name of the application for which to apply the changes, such as x_aah_custom_app. You can locate this value in the scope field in the Custom Application [sys_app] table or Store Application [sys_store_app] table for the app-customization. Data type: String</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>You can locate this value in the <strong>Sys ID</strong> field in the Custom Application [sys_app] table or Store Application [sys_store_app] table for the app-customization. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>409</td>
<td>Conflict. The requested item is not unique.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| error       | Error message.  
Data type: String |
| links       | All links and sys_ids associated with the response.  
Data type: Object |
|           | "links": {  
"progress": {Object}  
} |
| links.progress | Describes the progress link information.  
Data type: Object |
|           | "progress": {  
"id": "String",  
"url": "String"  
} |
<p>| links.progress.id | Unique identifier to use to obtain the progress details for the operation. Use this value when calling the endpoint /sn_cicd/progress/{progress_id}. |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>links.progress.url</td>
<td>URL to use to retrieve the progress details.</td>
</tr>
<tr>
<td>percent_complete</td>
<td>Percentage of the request that is complete.</td>
</tr>
<tr>
<td>status</td>
<td>Numeric execution state. Used with <code>status_label</code>. Values:</td>
</tr>
<tr>
<td></td>
<td>• 0 (Pending)</td>
</tr>
<tr>
<td></td>
<td>• 1 (Running)</td>
</tr>
<tr>
<td></td>
<td>• 2 (Successful)</td>
</tr>
<tr>
<td></td>
<td>• 3 (Failed)</td>
</tr>
<tr>
<td></td>
<td>• 4 (Canceled)</td>
</tr>
<tr>
<td>status_detail</td>
<td>Additional information about the current state.</td>
</tr>
<tr>
<td>status_label</td>
<td>Execution state description. Used with <code>status</code>. Values:</td>
</tr>
<tr>
<td></td>
<td>• Canceled</td>
</tr>
<tr>
<td></td>
<td>• Failed</td>
</tr>
<tr>
<td></td>
<td>• Pending</td>
</tr>
<tr>
<td></td>
<td>• Running</td>
</tr>
<tr>
<td></td>
<td>• Successful</td>
</tr>
<tr>
<td>status_message</td>
<td>Additional information on why the operation failed. Empty if the operation is successful.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: cURL request

curl
  'instance.servicenow.com/api/sn_cicd/sc/apply_changes?app_sys_id=043db024db737300a9a754e4dc961915' \n  --request POST \n  --header 'Accept: application/json' \n  --user 'username':'password'

{
  "result": {
    "links": {
      "progress": {
        "id": "a4fae8911bdc00103d374087bc4bcbbd",
        "url": "https://instance.service-now/api/sn_cicd/progress/a4fae8911bdc00103d374087bc4bcbbd"
      }
    },
    "status": "0",
    "status_label": "Pending",
    "status_message": "",
    "status_detail": "",
    "error": "",
    "percent_complete": 0
  }
}

Example: Python request

# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_cicd/sc/apply_changes?app_sys_id=043db024db737300a9a754e4dc961915'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept':"application/json"}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
    response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "links": {
            "progress": {
                "id": "a4fae8911bdc00103d374087bc4bcbbd",
                "url": "https://instance.service-now.com/api/sn_cicd/progress/a4fae8911bdc00103d374087bc4bcbbd"
            }
        },
        "status": "0",
        "status_label": "Pending",
        "status_message": "",
        "status_detail": "",
        "error": "",
        "percent_complete": 0
    }
}

CI/CD - POST /sn_cicd/testsuite/run

Starts a specified automated test suite. The test suite runs on the instance from which the endpoint was called.

⚠️ Note: If the suite you are running contains UI tests, you must have either a scheduled Client Test Runner open, or be able to configure headless Test Runner execution. To learn more see Headless browser for Automated Test Framework.

URL format

Versioned URL: /api/sn_cicd/{api_version}/testsuite/run
Default URL: /api/sn_cicd/testsuite/run
### Supported request parameters

#### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

#### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| browser_name | Name of the browser to use to run the client test. This value must match what is specified in the scheduled client test runner. For additional information on scheduled client test runners, see [Scheduled Client Test Runners](#). Data type: String Valid values: (must be all lower-case)  
  - any  
  - chrome  
  - firefox  
  - edge  
  - ie  
  - safari  
  Default: any |
| browser_version | Starting value of the version of the browser specified in browser_name to use to run the test. For example, if you enter "9", that would enable all 9.x.x.x versions. This value must match what is specified in the scheduled client test runner. Data type: String |
| os_name       | Name of the operating system under which to run the test suite. This value must match what is specified in the scheduled client test runner. Data type: String |
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>os_version</td>
<td>Starting value of the version of the operating system under which to run the test suite. For example, if you enter &quot;8&quot;, that would enable all 8.x.x.x versions. This value must match what is specified in the scheduled client test runner.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>test_suite_sys_id</td>
<td>Required if <code>test_suite_name</code> is not specified. The sys_id of the test suite to run. This value is located in the Test [sys_atf_test_suite] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>test_suite_name</td>
<td>Required if <code>test_suite_sys_id</code> is not specified. The name of the test suite to run. This value is located in the Test [sys_atf_test_suite] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>The user credentials are incorrect.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. The user is not an admin or does not have the sn_cicd.sys_ci_automation role.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>409</td>
<td>Conflict. The requested item is not unique.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Error message. Data type: String</td>
</tr>
<tr>
<td>links</td>
<td>All links and sys_ids associated with the response. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;links&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {Object}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>links.progress</td>
<td>Describes the progress link information.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;progress&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>links.progress.id</td>
<td>Unique identifier to use to obtain the progress details for the operation. Use this value when calling the endpoint /sn_cicd/progress/{progress_id}.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links.progress.url</td>
<td>URL to use to retrieve the progress details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links.results</td>
<td>Results information.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;results&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>links.results.id</td>
<td>Unique identifier of the results information. Use this value when calling the associated results endpoint.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>links.results.url</td>
<td>URL to use to obtain the results of the endpoint execution, such as results.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>percent_complete</td>
<td>Percentage of the request that is complete.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>status</td>
<td>Numeric execution state. Used with status_label.</td>
</tr>
<tr>
<td></td>
<td>Values:</td>
</tr>
<tr>
<td></td>
<td>• 0 (Pending)</td>
</tr>
<tr>
<td></td>
<td>• 1 (Running)</td>
</tr>
<tr>
<td></td>
<td>• 2 (Successful)</td>
</tr>
<tr>
<td></td>
<td>• 3 (Failed)</td>
</tr>
<tr>
<td></td>
<td>• 4 (Canceled)</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>status_detail</td>
<td>Additional information about the current state. Data type: String</td>
</tr>
</tbody>
</table>
| status_label     | Execution state description. Used with `status`. Values:  
|                  | • Canceled  
|                  | • Failed  
|                  | • Pending  
|                  | • Running  
|                  | • Successful  
| Data type: String|
| status_message   | Description of the current state. Data type: String                        |

**Example: cURL request**

```bash
curl 'https://instance.servicenow.com/api/sn_cicd/testsuite/run?test_suite_sys_id=632e43900b202050192f15d6673a7e' 
--request POST 
--header 'Accept: application/json' 
--user 'username':'password'
```

```json
{
  "result": {
    "links": {
      "progress": {
        "id": "0578a8d91bdc00103d374087bc4bcbcc",
        "url": "https://instance.service-now.com/api/sn_cicd/progress/0578a8d91bdc00103d374087bc4bcbcc"
      }
    },
    "status": "0",
    "status_label": "Pending",
    "status_message": "",
    "status_detail": "",
    "error": ""
  }
}
```
Failure response

{
  "result": {
    "status": "3",
    "status_label": "Failed",
    "status_message": "",
    "status_detail": "",
    "error": "Scheduled test/suite execution is disabled. Change the value of property 'sn_atf.schedule.enabled' to true to enable it"
  }
}

Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url =
  'https://instance.servicenow.com/api/sn_cicd/testsuite/run?test_suite_sys_id=632e43900b202
20050192f15d6673a7e'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept':"application/json"}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
```
data = response.json()
print(data)

{
  "result": {
    "links": {
      "progress": {
        "id": "0578a8d91bdc00103d374087bc4bcbbcc",
        "url": "https://instance.service-now.com/api/sn_cicd/progress/0578a8d91bdc00103d374087bc4bcbbcc"
      }
    },
    "status": "0",
    "status_label": "Pending",
    "status_message": "",
    "status_detail": "",
    "error": "",
    "percent_complete": 0
  }
}

CSM Attachment API

The CSM Attachment API provides endpoints that allow both internal and external users to upload, download, and remove attachments associated with a table and to retrieve attachment metadata.

Only attachments associated with tables specified in the glide.rest.attachment_csm_api.allowed_tables system property are valid in any of the associated endpoint calls. Any logged in user, both snc_internal and snc_external, can access this API. Guests and public users do not have access rights.

This API respects any system limitations on uploaded files, such as maximum file size and allowed attachment types. You can control these settings using the com.glide.attachment.max_size and glide.attachment.extensions properties.

By default, this API has a rate limit of 500 per hour for unauthenticated and snc_external users. For more information about rate limiting, see Inbound REST API rate limiting.

**CSM Attachment - DELETE /now/attachment_csm/{sys_id}**

Deletes the specified attachment.
URL format

Versioned URL: /api/now/{api_version}/attachment_csm/{sys_id}
Default URL: /api/now/v1/attachment_csm/{sys_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the attachment to delete. This attachment must belong to one of the tables listed within the glide.rest.attachment_csm_api.allowed_tables property in the System Properties [sys_properties] table.</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Successful. Indicates the request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. Indicates that the attachment from this table might not be accessible. Verify that the associated table is included in the glide.rest.attachment_csm_api.allowed_tables system property.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Example: Sample cURL request

```bash
curl
  "https://instance.servicenow.com/api/now/v1/attachment_csm/615ea769c0a80166001cf5f2367302f5b"
  --request DELETE
  --user 'username':'password'
```

None
Example: Sample Python request

```python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/v1/attachment_csm/615ea769c0a80166001cf5f23673025f5'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Do the HTTP request
response = requests.delete(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 204
if response.status_code != 204:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

print('Successfully Deleted')
```

CSM Attachment - GET /now/attachment_csm

Returns the metadata for multiple attachments.

ℹ️ **Note:** You can reference all sysparm query parameters using either their full name or their shortened name (without the `sysparm_` prefix). For example, for `sysparm_limit` you can also use `limit`.

**URL format**

Versioned URL: /api/now/{api_version}/attachment_csm

Default URL: /api/now/v1/attachment_csm
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. For requests that exceed this number of records, use the sysparm_offset parameter to paginate record retrieval. This limit is applied before ACL evaluation. If no records return, including records you have access to, rearrange the record order so records you have access to return first. Note: Unusually large sysparm_limit values can impact system performance. Data type: Number Default: 1000</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, sysparm_offset is set to &quot;0&quot;. simply page through all available records, use sysparm_offset=sysparm_offset+sysparm_limit, until you reach the end of all records. Do not pass a negative number in the sysparm_offset parameter. Data type: Number Default: 0</td>
</tr>
<tr>
<td>sysparm_query</td>
<td>Required. Encoded query to use to search for attachments. This parameter must at least contain sysparm_query=table_name=&lt;table&gt; where &lt;table&gt; can only be those tables specified in the glide.rest.attachment_csm_api.allowed_tables property located in the System Properties [sys_properties] table. For example: sysparm_query=file_name=kb_knowledge.</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax: <code>sysparm_query=&lt;col_name&gt;&lt;operator&gt;&lt;value&gt;</code></td>
<td></td>
</tr>
</tbody>
</table>
- `<col_name>`: Name of the table column to filter against.  
- `<operator>`: Supports the following values:  
  - `=`: Exactly matches `<value>`.  
  - `!`: Does not match `<value>`.  
  - `^`: Logically AND multiple query statements.  
  - `^OR`: Logically OR multiple query statements.  
  - LIKE: `<col_name>` contains the specified string. Only works for `<col_name>` fields whose data type is string.  
  - STARTSWITH: `<col_name>` starts with the specified string. Only works for `<col_name>` fields whose data type is string.  
  - ENDSWITH: `<col_name>` ends with the specified string. Only works for `<col_name>` fields whose data type is string.  
- `<value>`: Value to match against.  
All parameters are case-sensitive. Queries can contain more than one entry, such as `sysparm_query=<col_name><operator><value>[<operator><col_name><operator><value>`.  
For example:  
```
sysparm_query=caller_id=javascript:gs.getUserID()^active=true
```
Encoded queries also supports order by functionality. To sort responses based on certain fields, use the `ORDERBY` and `ORDERBYDESC` clauses in `sysparm_query`.  
Syntax:  
- `ORDERBY<col_name>`  
- `ORDERBYDESC<col_name>`  
For example: `sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory`  
This query filters all active records and orders the results in ascending order by number, and then in descending order by category.  
Data type: String
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Metadata of the requested attachment. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;average_image_color&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;compressed&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;content_type&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;created_by_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;download_link&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;file_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;image_height&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;image_width&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;size_bytes&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;size_compressed&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_created_by&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_created_on&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_mod_count&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_tags&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_updated_by&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_updated_on&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;table_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;table_sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;updated_by_name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

| result.average_image_color     | If the attachment is an image, the sum of all colors. Data type: String     |
|                                | Unit: RGB or number of pixels.                                              |

<p>| result.compressed              | Flag that indicates whether the attachment file has been compressed. Valid values: |
|                                | • true: File has been compressed.                                           |
|                                | • false: File has not been compressed.                                      |
|                                | Data type: String                                                          |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.content_type</td>
<td>Content-type of the associated attachment file, such as image or jpeg or application/x-shockwave-flash. Data type: String</td>
</tr>
<tr>
<td>result.created_by_name</td>
<td>Full name of entity that originally created the attachment file. Data type: String</td>
</tr>
<tr>
<td>result.download_link</td>
<td>Download URL of the attachment on the ServiceNow instance. Data type: String</td>
</tr>
<tr>
<td>result.file_name</td>
<td>File name of the attachment. Data type: String</td>
</tr>
<tr>
<td>result.image_height</td>
<td>If an image file, the height of the image. Data type: String Unit: Pixels</td>
</tr>
<tr>
<td>result.image_width</td>
<td>If an image file, the width of the image. Data type: String Unit: Pixels</td>
</tr>
<tr>
<td>result.size_bytes</td>
<td>Size of the attachment. Data type: String Unit: Bytes</td>
</tr>
<tr>
<td>result.size_compressed</td>
<td>Size of the compressed attachment file. If the file is not compressed, empty. Data type: String Unit: Bytes</td>
</tr>
<tr>
<td>result.sys_created_by</td>
<td>Entity that originally created the attachment file. Data type: String</td>
</tr>
<tr>
<td>result.sys_created_on</td>
<td>Date and time that the attachment file was initially saved to the instance. Data type: String</td>
</tr>
<tr>
<td>result.sys_id</td>
<td>Sys_id of the attachment file.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.sys_mod_count</td>
<td>Number of times the attachment file has been modified (uploaded to the instance). Data type: String</td>
</tr>
<tr>
<td>result.sys_tags</td>
<td>Any system tags associated with the attachment file. Data type: String</td>
</tr>
<tr>
<td>result.sys_updated_by</td>
<td>Entity that last updated the attachment file. Data type: String</td>
</tr>
<tr>
<td>result.sys_updated_on</td>
<td>Date and time that the attachment file was last updated. Data type: String</td>
</tr>
<tr>
<td>result.table_name</td>
<td>Name of the table to which the attachment is associated. Data type: String</td>
</tr>
<tr>
<td>result.table_sys_id</td>
<td>Sys_id of the table associated with the attachment. Data type: String</td>
</tr>
<tr>
<td>result.updated_by_name</td>
<td>Full name of entity that last updated the attachment file. Data type: String</td>
</tr>
</tbody>
</table>

Example: Sample cURL request

```
curl "https://instance.service-now.com/api/now/v1/attachment_csm?sysparm_limit=1" \
--request GET \
--header "Accept:application/json" \
--user "username":"password"
```

```json
{
   "result": [
      {
         "table_sys_id": "5054b6f8c0a800060056addcf551ecf8",
         "size_bytes": "462",
      }
   ]
}
```
Example: Sample Python request

```python
import requests

# Set the request parameters
url = 'https://instance.service-now.com/api/now/v1/attachment_csm?sysparm_limit=1'

# Eg. User name="admin", Password="admin" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {"Accept":"application/xml"}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Error:', response.status_code)
```
print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
  <table_sys_id>5054b6f8c0a800060056addcf551ecf8</table_sys_id>
  <size_bytes>462</size_bytes>
  <download_link>https://instance.service-now.com/api/now/attachment/615ea769c0a80166001cf5f2367302f5/file</download_link>
  <sys_updated_on>2009-05-21 04:12:21</sys_updated_on>
  <sys_id>615ea769c0a80166001cf5f2367302f5</sys_id>
  <sys_created_on>2009-05-21 04:12:21</sys_created_on>
  <file_name>blocks.swf</file_name>
  <sys_created_by>glide.maint</sys_created_by>
  <compressed>true</compressed>
  <sys_updated_by>glide.maint</sys_updated_by>
  <sys_tags />
  <sys_mod_count>0</sys_mod_count>
  <table_name>content_block_programmatic</table_name>
  <content_type>application/x-shockwave-flash</content_type>
  <size_compressed>485</size_compressed>
  <created_by_name>John Smith</created_by_name>
  <updated_by_name>John Smith</updated_by_name>
  </result>
</response>

**CSM Attachment - GET /now/attachment_csm/{sys_id}/file**

Returns the binary file attachment for the specified attachment sys_id.

**URL format**

Versioned URL: /api/now/{api_version}/attachment_csm/{sys_id}/file

Default URL: /api/now/v1/attachment_csm/{sys_id}/file
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the attachment record to obtain. This attachment must belong to one of the tables listed within the glide.rest.attachment_csm_api.allowed_tables property in the System Properties [sys_properties] table.</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. For example, use image/jpeg or image/png to accept JPEG or PNG image files exclusively. To allow all image types, specify image/*; to allow any file type, specify <em>/</em>. Default: <em>/</em></td>
</tr>
</tbody>
</table>
## Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-Attachment-Metadata</td>
<td>Metadata about the returned file, such as size, name, and file type.</td>
</tr>
</tbody>
</table>

## Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. Indicates that the attachment from this table might not be accessible. Verify that the associated table is included in the <code>glide_rest_attachment_csm_api_allowed_tables</code> system property.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

## Response body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary file</td>
<td></td>
</tr>
</tbody>
</table>

## Example: Sample cURL request

```
curl *https://instance.servicenow.com/api/now/v1/attachment_csm/615ea769c0a80166001cf5f2367302f5/file" \
  --request GET \
  --header "Accept:*/**" \
  --user 'username':'password'
```

<Binary file>
Example: Sample Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url =
'https://instance.servicenow.com/api/now/v1/attachment_csm/615ea769c0a80166001cf5f2367302f5/file'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept': '*/*'}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.json())
    exit()

# Binary file
```

CSM Attachment - GET /now/attachment_csm/{sys_id}

Returns the metadata for the attachment file with the specific attachment sys_id.

URL format

Versioned URL: /api/now/{api_version}/attachment_csm/{sys_id}
Default URL: /api/now/v1/attachment_csm/{sys_id}
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the attachment record for which to return metadata. This attachment must belong to one of the tables listed within the glide.rest.attachment_csm_api.allowed_tables property in the System Properties [sys_properties] table. Data type: String</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. Indicates that the attachment from this table might not be accessible. Verify that the associated table is included in the <code>glide.rest.attachment_csm_api.allowed_tables</code> system property.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. Indicates the specified attachment does not exist, or the current user does not have the rights to access it.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Metadata of the requested attachment. Data type: Object</td>
</tr>
</tbody>
</table>

```json
"result": {
  "average_image_color": "String",
  "compressed": "String",
  "content_type": "String",
  "created_by_name": "String",
  "download_link": "String",
  "file_name": "String",
  "image_height": "String",
  "image_width": "String"
}
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;size_bytes&quot;: &quot;String&quot;</td>
<td>&quot;size_bytes&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;size_compressed&quot;: &quot;String&quot;</td>
<td>&quot;size_compressed&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;sys_created_by&quot;: &quot;String&quot;</td>
<td>&quot;sys_created_by&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;sys_created_on&quot;: &quot;String&quot;</td>
<td>&quot;sys_created_on&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;sys_id&quot;: &quot;String&quot;</td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;sys_mod_count&quot;: &quot;String&quot;</td>
<td>&quot;sys_mod_count&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;sys_tags&quot;: &quot;String&quot;</td>
<td>&quot;sys_tags&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;sys_updated_by&quot;: &quot;String&quot;</td>
<td>&quot;sys_updated_by&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;sys_updated_on&quot;: &quot;String&quot;</td>
<td>&quot;sys_updated_on&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;table_name&quot;: &quot;String&quot;</td>
<td>&quot;table_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;table_sys_id&quot;: &quot;String&quot;</td>
<td>&quot;table_sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;updated_by_name&quot;: &quot;String&quot;</td>
<td>&quot;updated_by_name&quot;: &quot;String&quot;</td>
</tr>
</tbody>
</table>

result.average_image_color     If the attachment is an image, the sum of all colors.  
Data type: String  
Unit: RGB or number of pixels.

result.compressed     Flag that indicates whether the attachment file has been compressed.  
Valid values:  
• true: File has been compressed.  
• false: File has not been compressed.  
Data type: String

result.content_type     Content-type of the associated attachment file, such as image or jpeg or application/x-shockwave-flash.  
Data type: String

result.created_by_name     Full name of entity that originally created the attachment file.  
Data type: String

result.download_link     Download URL of the attachment on the ServiceNow instance.  
Data type: String

result.file_name     File name of the attachment.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.image_height</td>
<td>If an image file, the height of the image. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Unit: Pixels</td>
</tr>
<tr>
<td>result.image_width</td>
<td>If an image file, the width of the image. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Unit: Pixels</td>
</tr>
<tr>
<td>result.size_bytes</td>
<td>Size of the attachment. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Unit: Bytes</td>
</tr>
<tr>
<td>result.size_compressed</td>
<td>Size of the compressed attachment file. If the file is not compressed, empty. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Unit: Bytes</td>
</tr>
<tr>
<td>result.sys_created_by</td>
<td>Entity that originally created the attachment file. Data type: String</td>
</tr>
<tr>
<td>result.sys_created_on</td>
<td>Date and time that the attachment file was initially saved to the instance. Data type: String</td>
</tr>
<tr>
<td>result.sys_id</td>
<td>Sys_id of the attachment file. Data type: String</td>
</tr>
<tr>
<td>result.sys_mod_count</td>
<td>Number of times the attachment file has been modified (uploaded to the instance). Data type: String</td>
</tr>
<tr>
<td>result.sys_tags</td>
<td>Any system tags associated with the attachment file. Data type: String</td>
</tr>
<tr>
<td>result.sys_updated_by</td>
<td>Entity that last updated the attachment file. Data type: String</td>
</tr>
<tr>
<td>result.sys_updated_on</td>
<td>Date and time that the attachment file was last updated.</td>
</tr>
</tbody>
</table>
### Element Description

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.table_name</td>
<td>Name of the table to which the attachment is associated.</td>
</tr>
<tr>
<td>result.table_sys_id</td>
<td>Sys_id of the table associated with the attachment.</td>
</tr>
<tr>
<td>result.updated_by_name</td>
<td>Full name of entity that last updated the attachment file.</td>
</tr>
</tbody>
</table>

### Example: Sample cURL request

```bash
curl
  "https://instance.servicenow.com/api/now/v1/attachment_csm/615ea769c0a80166001cf5f2367302f5" \ 
  --request GET \ 
  --header "Accept:application/json" \ 
  --user "username":"password"
```

```json
|
| "result": |
| "table_sys_id": "5054b6f8c0a80006056addcf551ecf8",
| "size_bytes": "462",
| "download_link": |
| "https://instance.service-now.com/api/now/v1/attachment_csm/615ea769c0a80166001cf5f2367302f5/file",
| "sys_updated_on": "2019-05-21 04:12:21",
| "sys_id": "615ea769c0a80166001cf5f2367302f5",
| "image_height": "",
| "sys_created_on": "2019-05-21 04:12:21",
| "file_name": "blocks.swf",
| "sys_created_by": "glide.maint",
| "compressed": "true",
| "average_image_color": "",
| "sys_updated_by": "glide.maint",
| "sys_tags": "",
| "table_name": "content_block_programmatic",
| "image_width": "",
| "sys_mod_count": "0",
```
Example: Sample Python request

```python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/v1/attachment_csm/615ea769c0a80166001cf5f2367302f5'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept': 'application/xml'}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <table_sys_id>5054b6f8c0a800060056adDCF551ecf8</table_sys_id>
    <size_bytes>462</size_bytes>
    <download_link>https://instance.service-now.com/api/now/v1/attachment_csm/615ea769c0a80166001cf5f2367302f5/file</download_link>
    <sys_updated_on>2019-05-21 04:12:21</sys_updated_on>
  </result>
</response>
```
CSM Attachment - POST /now/attachment_csm/file

Uploads a specified binary file as an attachment to a specified record.

The endpoint also returns the metadata for the saved attachment.

**URL format**

Versioned URL: /api/now/{api_version}/attachment_csm/file

Default URL: /api/now/v1/attachment_csm/file

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>encryption_context</td>
<td>Sys_id of an encryption context record. Specify this parameter to allow only users with the specified encryption context to access the attachment. For additional information on encryption context records, see Encryption Support. Data type: String Default: Attached file is not encrypted with any encryption context.</td>
</tr>
<tr>
<td>file_name</td>
<td>Required. Name to give the attachment. Note: The file to attach must be specified after the last parameter in the passed-in query parameter list. Data type: String</td>
</tr>
<tr>
<td>table_name</td>
<td>Required. Name of the table to which you want to attach the file. This table must be listed within the glide.rest.attachment_csm_api.allowed_tables system property in the System Properties [sys_properties] table. Data type: String</td>
</tr>
<tr>
<td>table_sys_id</td>
<td>Required. Sys_id of the record on the specified table to which you want to attach the file. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String&gt;</td>
<td>Path to the binary file to attach to the specified record. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Required. Data format of the file to post. For example, use image/jpeg or image/png to post JPEG or PNG image files exclusively. To allow all image types, specify image/*; to allow any file type, specify <em>/</em>.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. Indicates that the attachment from this table might not be accessible. Verify that the associated table is included in the glide.rest.attachment_csm_api.allowed_tables system property.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. Indicates the record specified by the table_name and table_sys_id parameters does not exist or is not accessible by the current user.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
# Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Metadata of the requested attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;average_image_color&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;compressed&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;content_type&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;created_by_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;download_link&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;file_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;image_height&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;image_width&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;size_bytes&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;size_compressed&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_created_by&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_created_on&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_mod_count&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_tags&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_updated_by&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_updated_on&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;table_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;table_sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;updated_by_name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>result.average_image_color</th>
<th>If the attachment is an image, the sum of all colors.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td>Unit: RGB or number of pixels.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>result.compressed</th>
<th>Flag that indicates whether the attachment file has been compressed.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valid values:</td>
<td>• true: File has been compressed.</td>
</tr>
<tr>
<td></td>
<td>• false: File has not been compressed.</td>
</tr>
<tr>
<td>Data type: String</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.content_type</td>
<td>Content-type of the associated attachment file, such as image or jpeg or application/x-shockwave-flash. Data type: String</td>
</tr>
<tr>
<td>result.created_by_name</td>
<td>Full name of entity that originally created the attachment file. Data type: String</td>
</tr>
<tr>
<td>result.download_link</td>
<td>Download URL of the attachment on the ServiceNow instance. Data type: String</td>
</tr>
<tr>
<td>result.file_name</td>
<td>File name of the attachment. Data type: String</td>
</tr>
<tr>
<td>result.image_height</td>
<td>If an image file, the height of the image. Data type: String Unit: Pixels</td>
</tr>
<tr>
<td>result.image_width</td>
<td>If an image file, the width of the image. Data type: String Unit: Pixels</td>
</tr>
<tr>
<td>result.size_bytes</td>
<td>Size of the attachment. Data type: String Unit: Bytes</td>
</tr>
<tr>
<td>result.size_compressed</td>
<td>Size of the compressed attachment file. If the file is not compressed, empty. Data type: String Unit: Bytes</td>
</tr>
<tr>
<td>result.sys_created_by</td>
<td>Entity that originally created the attachment file. Data type: String</td>
</tr>
<tr>
<td>result.sys_created_on</td>
<td>Date and time that the attachment file was initially saved to the instance. Data type: String</td>
</tr>
<tr>
<td>result.sys_id</td>
<td>Sys_id of the attachment file.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.sys_mod_count</td>
<td>Number of times the attachment file has been modified (uploaded to the instance). Data type: String</td>
</tr>
<tr>
<td>result.sys_tags</td>
<td>Any system tags associated with the attachment file. Data type: String</td>
</tr>
<tr>
<td>result.sys_updated_by</td>
<td>Entity that last updated the attachment file. Data type: String</td>
</tr>
<tr>
<td>result.sys_updated_on</td>
<td>Date and time that the attachment file was last updated. Data type: String</td>
</tr>
<tr>
<td>result.table_name</td>
<td>Name of the table to which the attachment is associated. Data type: String</td>
</tr>
<tr>
<td>result.table_sys_id</td>
<td>Sys_id of the table associated with the attachment. Data type: String</td>
</tr>
<tr>
<td>result.updated_by_name</td>
<td>Full name of entity that last updated the attachment file. Data type: String</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

Shows how to upload the .jpg file `Issue_screenshot.jpg` in the `/images` directory.

```bash
curl "https://instance.servicenow.com/api/now/v1/attachment_csm/file?table_name=incident&table_sys_id=d71f7935c0a8016700802b64c67c11c6&file_name=Issue_screenshot" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type: image/jpeg" \
  --user "username":"password" \
  --data-binary "@/images/Issue_screenshot.jpg"
```

{ "result": { } }

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: Sample Python request

```python
import requests

# Set the request parameters
url =
    "https://instance.servicenow.com/api/now/v1/attachment_csm/file?table_name=incident&table_sys_id=d71f7935c0a8016700802b64c67c11c6&file_name=Issue_screenshot.jpg"

# Specify the file To send. When specifying files to send make sure you specify the path to the file, in this example the file was located in the same directory as the python script being executed.
data = open('Issue_screenshot.jpg', 'rb').read()

# Eq. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'
```
# Set proper headers
headers = {"Content-Type":"image/jpeg","Accept":"application/json"}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, data=data)

# Check for HTTP codes other than 201
if response.status_code != 201:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "table_sys_id": "d71f7935c0a8016700802b64c67c11c6",
        "size_bytes": "36597",
        "download_link": "https://instance.service-now.com/api/now/v1/attachment_csm/6ea10fe64f411200adf9f8e18110c739/file",
        "sys_updated_on": "2019-01-22 15:14:07",
        "sys_id": "6ea10fe64f411200adf9f8e18110c739",
        "image_height": "",
        "sys_created_on": "2019-01-22 15:14:07",
        "file_name": "Issue_screenshot.jpg",
        "sys_created_by": "username",
        "compressed": "true",
        "average_image_color": "",
        "sys_updated_by": "username",
        "sys_tags": "",
        "table_name": "incident",
        "image_width": "",
        "sys_mod_count": "0",
        "content_type": "image/jpeg",
        "size_compressed": "25130",
        "created_by_name": "John Smith",
        "updated_by_name": "John Smith"
    }
}
CSM Attachment - POST /now/attachment_csm/upload

Uploads a multipart file attachment.

The multipart POST method does not accept any parameters. You must specify the table name and record sys_id values within the form body. See the cURL example below for a sample of a multipart/form-data request.

⚠️ **Note:** When using multipart POST, ensure the file content is contained in the final part of the message only. Earlier parts should contain only metadata such as table name and record sys_id.

When sending a multipart/form-data POST request to upload a file attachment, include attachment data in the form body, not in the URL parameters or request body.

**Mandatory values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Content-Type of the file, included in the message body for multipart uploads.</td>
</tr>
<tr>
<td></td>
<td>⚠️ <strong>Note:</strong> You must define the Content-Type within the file portion of the POST message, not within the form data.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>table_name</td>
<td>Name of the table to which you want to attach the file. This table must be specified in the glide.rest.attachment_csm_api.allowed_tables property in the System Properties [sys_properties] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>table_sys_id</td>
<td>Sys_id of the record on the specified table to which you want to attach the file.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**URL format**

Versioned URL: /api/now/{api_version}/attachment_csm/upload

Default URL: /api/now/v1/attachment_csm/upload
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Set this value to <code>multipart/form-data</code> when using the multipart POST method.</td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>URL of the new attachment in the ServiceNow platform instance.</td>
</tr>
</tbody>
</table>
### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Successful. Indicates the request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. Indicates that the attachment from this table might not be accessible. Verify that the associated table is included in the <code>glide.rest.attachment_csm_api.allowed_tables</code> system property.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Metadata of the requested attachment. Data type: Object</td>
</tr>
</tbody>
</table>

```
"result": {
  "average_image_color": "String",
  "compressed": "String",
  "content_type": "String",
  "created_by_name": "String",
  "download_link": "String",
  "file_name": "String",
  "image_height": "String",
  "image_width": "String",
  "size_bytes": "String",
  "size_compressed": "String",
  "sys_created_by": "String",
  "sys_created_on": "String",
  "sys_id": "String",
  "sys_mod_count": "String",
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_tags: String, sys_updated_by: String, sys_updated_on: String, table_name: String, table_sys_id: String, updated_by_name: String</td>
<td></td>
</tr>
<tr>
<td>result.average_image_color</td>
<td>If the attachment is an image, the sum of all colors. Data type: String Unit: RGB or number of pixels.</td>
</tr>
<tr>
<td>result.compressed</td>
<td>Flag that indicates whether the attachment file has been compressed. Valid values: • true: File has been compressed. • false: File has not been compressed. Data type: String</td>
</tr>
<tr>
<td>result.content_type</td>
<td>Content-type of the associated attachment file, such as image or jpeg or application/x-shockwave-flash. Data type: String</td>
</tr>
<tr>
<td>result.created_by_name</td>
<td>Full name of entity that originally created the attachment file. Data type: String</td>
</tr>
<tr>
<td>result.download_link</td>
<td>Download URL of the attachment on the ServiceNow instance. Data type: String</td>
</tr>
<tr>
<td>result.file_name</td>
<td>File name of the attachment. Data type: String</td>
</tr>
<tr>
<td>result.image_height</td>
<td>If an image file, the height of the image. Data type: String Unit: Pixels</td>
</tr>
<tr>
<td>result.image_width</td>
<td>If an image file, the width of the image.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.size_bytes</td>
<td>Size of the attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Unit: Bytes</td>
</tr>
<tr>
<td>result.size_compressed</td>
<td>Size of the compressed attachment file. If the file is not compressed, empty.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Unit: Bytes</td>
</tr>
<tr>
<td>result.sys_created_by</td>
<td>Entity that originally created the attachment file.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.sys_created_on</td>
<td>Date and time that the attachment file was initially saved to the instance.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.sys_id</td>
<td>Sys_id of the attachment file.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.sys_mod_count</td>
<td>Number of times the attachment file has been modified (uploaded to the instance).</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.sys_tags</td>
<td>Any system tags associated with the attachment file.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.sys_updated_by</td>
<td>Entity that last updated the attachment file.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.sys_updated_on</td>
<td>Date and time that the attachment file was last updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.table_name</td>
<td>Name of the table to which the attachment is associated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
### Element Description

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.table_sys_id</td>
<td>Sys_id of the table associated with the attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.updated_by_name</td>
<td>Full name of entity that last updated the attachment file.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Example: Sample cURL request

```bash
curl "https://instance.service-now.com/api/now/v1/attachment_csm/upload" \ 
   --request POST \ 
   --header "Accept:application/json" \ 
   --header "Content-Type:multipart/form-data" \ 
   --user "username":"password" \ 
   -F "table_name=incident" \ 
   -F "table_sys_id=d71f7935c0a8016700802b64c67c11c6" \ 
   -F "uploadFile=@/image/banner-CS0001345_v1_1.jpeg"
```

```json
{
    "result": {
        "table_sys_id": "d71f7935c0a8016700802b64c67c11c6",
        "size_bytes": "36597",
        "download_link": "https://instance.service-now.com/api/now/v1/attachment_csm/994adbc64f511200adf9f8e18110c796/file",
        "sys_updated_on": "2019-02-02 14:00:21",
        "sys_id": "994adbc64f511200adf9f8e18110c796",
        "image_height": "",
        "sys_created_on": "2019-02-02 14:00:21",
        "file_name": "banner-CS0001345_v1_1.jpeg",
        "sys_created_by": "username",
        "compressed": "true",
        "average_image_color": "",
        "sys_updated_by": "username",
        "sys_tags": "",
        "table_name": "incident",
        "image_width": "",
        "sys_mod_count": "0",
        "content_type": "image/jpeg",
        "size_compressed": "25130",
        "created_by_name": "John Smith",
    }
}
```
Example: Sample Python request

```python
# This example uses the Python Requests Library and you will need to install requests package for python
# Documentation can be found at http://docs.python-requests.org/en/master/user/quickstart/
import requests
import print
import json

# Specify the Endpoint URL replacing instance with your ServiceNow Instance Name
url = 'https://instance.service-now.com/api/now/v1/attachment_csm/upload'

# Specify Parameters for File Being Uploaded, the table_name and table_sys_id should be replaced with values that make sense for your use case
payload = {'table_name': 'incident', 'table_sys_id': '81f8915bdb6ba20028927416bf961971'}

# Specify Files To Send and Content Type. When specifying files to send make sure you specify the path to the file, in this example the file was located in the same directory as the python script being executed. It is important to specify the correct file type
files = {'file': ('issue_screenshot.JPG', open('issue_screenshot.JPG', 'rb'), 'image/jpg', {'Expires': '0'})}

# Eg. User name="username", Password="password" for this code sample. This will be sent across as basic authentication
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept': '*/*'}

# Send the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, files=files, data=payload)

# Check for HTTP codes other than 201
if response.status_code != 201:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
```
exit()

# Print Response Details

print 'Response Status Code:', response.status_code

print ''

print('Response Payload:)

print json.dumps(response.json(), indent=4)

Response Status Code: 201

Response Payload:

{
    "result": {
        "sys_tags": "",
        "sys_updated_by": "username",
        "content_type": "image/jpg",
        "sys_created_by": "username",
        "file_name": "issue_screenshot.JPG",
        "sys_updated_on": "2019-01-05 10:47:09",
        "sys_created_on": "2019-01-05 10:47:09",
        "image_width": "",
        "image_height": "",
        "sys_mod_count": "0",
        "table_name": "incident",
        "sys_id": "96679f724f84320025e874828110c7bd",
        "download_link": "https://instance.service-now.com/api/now/v1/attachment_csm/96679f724f84320025e874828110c7bd/file",
        "average_image_color": "",
        "size_bytes": "197484",
        "table_sys_id": "81f8915bcb6ba20028927416bf961971",
        "size_compressed": "197005",
        "compressed": "true",
        "created_by_name": "John Smith",
        "updated_by_name": "John Smith"
    }
}

CTI API

The CTI API provides REST resources that enable Computer Telephony Integration (CTI) providers to interact with the Cloud Call Center framework.

Using this API, integrators are able to:
• Invoke CTI operations using custom operation handlers within the Cloud Call Center framework.
• Pass events from a CCP provider to a ServiceNow instance.
• Render transcripts and recordings for a specified agent call.

This API runs in the `sn_cti_core` namespace. Before you are able to access this API, you must install the Cloud Call Center Core (sn_cti_core) application. For information on this installation, see Install Cloud Call Center applications.

For additional information on the Cloud Call Center, see Cloud Call Center.

**CTI_API - GET /sn_cti_core/cti_api/call_analysis/{interaction_sys_id}**

Renders transcripts and recordings for a specified agent call.

Now components use this endpoint in Agent and Manager workspaces to render associated agent call transcripts and recordings. To access this endpoint, the user credentials that you pass in the call must have the workspace_user, interaction_agent, or admin role.

This endpoint has a corresponding extension point, `sn_cti_core.CTICallAnalysisExtractor`, that is responsible for rendering the call data. The base system provides an implementation for Amazon Connect. If your implementation is different, you should customize this extension point before utilizing this endpoint. Also, the response parameters returned by this endpoint correspond to the implementation of this extension point. For additional information on implementing extension points, see Using extension points to extend application functionality and Extension points in Cloud Call Center.

**URL format**

Default URL: `/api/sn_cti_core/cti_api/call_analysis/{interaction_sys_id}`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interaction_sys_id</td>
<td>Sys_id of the agent call for which to render transcript and recording. Located in the Interaction [interaction] table. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
**Response body parameters (JSON or XML)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Depends on the implementation in the sn_cti_core.CTICallAnalysisExtractor extension point.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

The following example requests the rendering of a specific agent call.

```bash
curl -X POST \\nhttps://instance.servicenow.com/api/sn_cti_core/cti_api/call_analysis/88f2d9b7dbd7101058f05214dc96192a \
-H 'Accept: application/json' \
-H 'Content-Type: application/json' \
-u 'username':'password'
```

Rendered call data based on the processing specified in the sn_cti_core.CTICallAnalysisExtractor extension point.

```json
{
    "result":{
        "status":"success",
        "recordingData":{
            "recordingUrls":[
                "https://instance.awsapps.com/connect/get-recording?format=wav&callLegId=7626da7f-e083-4085-b77a-85bfea18fe24",
            ],
            "status":"success"
        },
        "transcriptData":{
```

bye.

"transcript": [
{
"content":"Okay. Hello? Mhm. Okay, okay.",
"beginOffsetMillis":3940,
"participantId":"CUSTOMER",
"additionalInfo": [
{
  "value":"Neutral",
  "highlightColor":"info"
}
]
},
{
"content":"bye.",
"beginOffsetMillis":26040,
"participantId":"AGENT",
"additionalInfo": [
{
  "value":"Neutral",
  "highlightColor":"info"
}
]
}
],
"sentimentTrend": [
{
  "x":0,
  "y":0
},
{
  "x":3939,
  "y":0
},
{
  "x":3940,
  "y":0
},
{
  "x":26450,
  "y":0
}
],
"sentimentPercents": {
  "positive":0,
CTI_API - POST /sn_cti_core/cti_api/providers/{provider}/components/{component}/versions/{version}

Invokes Computer Telephony Integration (CTI) operations using a specified operation handlers within the Cloud Call Center framework.

Before calling this endpoint, you must create a provider configuration record and associated message transformers. This is typically done when the Cloud Call Center framework is initially implemented in your instance. For additional information, see Provider configuration in Cloud Call Center.

A message transformer is responsible for parsing the payload specified in the request body of this endpoint and setting the payload data on an associated CTIOperationRequest object. An operation handler is then responsible for getting the information from the CTIOperationRequest object and using the data to process the requested operation.
After the message transformer finishes parsing the passed in payload, the Cloud Call Center framework instantiates the specified operation handler. The operating handler uses the `CTIOperationResponse - Scoped, Global API` GET methods to obtain the information that it needs from the associated `CTIOperationRequest` object to process the requested operation.

You define the operation handler to invoke, and its associated parameters, in the request body of this endpoint call. The specified operation handler must be defined in your instance within the Operation Handler `[sn_cti_operation_handler]` table. For additional information on creating operation handlers, see Configure a contact flow for an automated caller interaction.

The ServiceNow base system provides working operation handlers and message transformers that enable connection to Amazon Connect. When building contact flows within Amazon Connect, there are two integration points between Amazon services and a ServiceNow instance:

- Amazon Web Services (AWS) Lambda Proxy (Invoke AWS Lambda function)
- AWS Lex Bot (Get Customer Input)

You can find the available operation handlers and message transformers for these integration points in the Operation Handlers `[sn_cti_operation_handler]` and Provider Message Transformer `[sn_cti_provider_msg_transformer]` tables.

In addition, the user credentials that you pass in the call must have the `sn_cti_core.service` or `admin` role to access this endpoint.

For additional information on the Cloud Call Center, see Cloud Call Center.

**URL format**

Default URL: `/api/sn_cti_core/cti_api/providers/{provider}/components/{component}/versions/{version}`

**Supported request parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>component</td>
<td>Label of your provider component, such as <code>sn_aws_connect_lambda_proxy_component</code> or <code>sn_aws_connect_lex_intent_processor_component</code>. Located in the Component field of the Provider Component <code>[sn_cti_provider_component]</code> table. Data type: String</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>provider</td>
<td>Sys_id of your CTI provider record. Located in the Provider Component [sn_cti_provider_component] table. Data type: String</td>
</tr>
<tr>
<td>version</td>
<td>Version of the specified provider record. Located in the Version field of the Provider Component [sn_cti_provider_component] table. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Depends on the operation handler being called and its specific implementation.</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>Various</td>
<td>Depends on the operation being called and its specific implementation.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Depends on the operation being called and its specific implementation.</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example invokes the interactionEvent operation handler through the sn aws connect lambda proxy component component.

```bash
curl -X POST \
https://instance.servicenow.com/api/sn_cti_core/cti_api/providers/c4b12aa6e700001034b36584c2f6a9bc/components/sn aws connect lambda proxy component/versions/v1.0 \ 
-H 'Accept: application/json' \ 
-H 'Content-Type: application/json' \ 
-u 'username': 'password' \ 
-d '\
  "context": {"callbackWaitsForEmptyEventLoop": true, "functionVersion": "$LATEST", "functionName": "snRALJNLambda", "
```
"memoryLimitInMB": "128",
"logGroupName": "/aws/lambda/snRALJNLambda",
"logStreamName": "2020/09/24/[LATEST]b604a9579ef548a1b3f6b6739d19e1d9",
"awsRequestId": "b528a5a8-bb0b-4ec8-8e76-aa568f68f0c8"
},
"event": {
  "Details": {
    "ContactData": {
      "Attributes": {
        "$accepted_terms": "true"
      },
      "Channel": "VOICE",
      "ContactId": "8fe7c107-78df-4ec8-85b3-91c6dcb8e912",
      "CustomerEndpoint": {
        "Address": "+16693504374",
        "Type": "TELEPHONE_NUMBER"
      },
      "Description": null,
      "InitialContactId": "8fe7c107-78df-4ec8-85b3-91c6dcb8e912",
      "InitiationMethod": "INBOUND",
      "LanguageCode": "en-US",
      "MediaStreams": {
        "Customer": {
          "Audio": null
        }
      },
      "Name": null,
      "PreviousContactId": "8fe7c107-78df-4ec8-85b3-91c6dcb8e912",
      "Queue": {
        "ARN": "arn:aws:connect:us-west-2:028738656045:instance/d276bd83-6305-4f63-abee-e66fd5bdf503/queue/0f17b2d-d493-4399-e8c6-42d0d0e652f",
        "Name": "BasicQueue",
        "OutboundCallerId": {
          "Address": "+18335441399",
          "Type": "TELEPHONE_NUMBER"
        }
      }
    },
    "References": {},
    "SystemEndpoint": {
      "Address": "+18335441399"
    }
  },
"Type": "TELEPHONE_NUMBER"

```
{
  "Parameters": {
    "sn_component": "sn_aws_connect_lambda_proxy_component",
    "sn_operation": "interactionEvent"
  },
  "Name": "ContactFlowEvent"
}
```  

Example response to the interactionEvent operation handler call.

```
{
  "result": {
    "snc_user_first_name": "Jane",
    "snc_user_last_name": "Doe",
    "snc_user_sys_id": "bbe591f9b5710100231dbdb5e9619e3",
    "snc_user_vip": "false",
    "interactionTable": "interaction",
    "interactionId": "973b07c8dfe710100231dbdb5e961908",
    "statusCode": 200,
    "message": "success"
  }
}
```  

CTI_API - POST /sn_cti_core/cti_api/softphone/sources/{source}/actor/{actor}/

Passes events from a Contact Control Panel (CCP) provider to a ServiceNow instance.

Typically you call this endpoint from your CCP UI page to perform various tasks (events), such as changing the presence state of an agent or changing the state of a record on call completion. To access this endpoint, the user credentials that you pass in the call must have the sn_openframe_user or admin role.

This endpoint has a corresponding extension point, sn_cti_core.SoftPhoneEventSink, that is responsible for determining the action that is actually performed by this endpoint call for each event. The base system provides an implementation for Amazon Connect events. If your implementation is different, you should customize this extension point before utilizing this endpoint. The request body and response formats of this endpoint correspond to the implementation of this extension point. For additional information on
implementing extension points, see Using extension points to extend application functionality and Extension points in Cloud Call Center.

**URL format**

Default URL: `/api/sn_cti_core/cti_api/softphone/sources/{source}/actor/{actor}/events/{event}`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| actor | Emitter of the event, such as agent or external_system. The specific values are determined by the implementer.  
Data type: String |
| event | Name of the event in the Computer Telephony Integration (CTI) vendor’s system, such as Available or Offline. The specific values are determined by the implementer but must be a single word/phrase without spaces or special characters.  
Data type: String |
| sources | Identifier of the vendor’s Softphone UI page, such as aws_ccp.  
Data type: String |

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implementation specific. Determined by the implementation of the sn_cti_core.SoftPhoneEventSink extension point.</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example passes AgentStatusOne event information to the ServiceNow instance. The request body parameters are determined by the implementation of the `sn_cti_core.SoftPhoneEventSink` extension point.

```bash
curl -X POST https://instance.servicenow.com/api/sn_cti_core/cti_api/softphone/sources/aws_ccp/actor/agent/events/AgentStatusOne
   -H 'Accept: application/json'
   -H 'Content-Type: application/json'
   -u 'username':'password'
   -d '{
```
No response parameters are returned.

```
{
  "result": {}
}
```

**Customer Central API**

The Customer Central API provides access to customer information and configuration details.
Use this API to obtain information records (cards) that were previously generated through the Customer Central interface.

You must have the admin or sn_esm_agent role to access this API. In addition, before this API is available in your instance you must activate the com.sn_csm_customer_central plugin.

For additional information, see Customer Central.

**Customer Central - GET /cust_central/custinfo/{context}/{context_instance}/{target}**

Returns customer information and configuration details for a specific customer context.

Customer context is determined by the customer content table that you specify, either the Consumer [csm_consumer] or Contact [customer_contact] table, and its associated target table. This information is located in the Customer Information Context [sn_customercentral_cust_info_config] table.

This endpoint returns all active records (cards) within the following Customer Central configuration tables for the specified customer context:

- List Configuration [sn_customercentral_list_config]
- Record Configuration [sn_customercentral_record_config]
- Report Configuration [sn_customercentral_report_config]
- Report Group Configuration [sn_customercentral_report_group_config]

You must define any additional configuration records that you need through the Customer Central interface. For details, see Configure customer central.

**URL format**

**Versioned URL:** /api/sn_customercentral/{api_version}/cust_central/custinfo/{context}/{context_instance}/{target}

**Default URL:** /api/sn_customercentral/cust_central/custinfo/{context}/{context_instance}/{target}
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>context</td>
<td>Name of the customer context table to use. Valid values: • customer_contact • csm_consumer Data type: String</td>
</tr>
<tr>
<td>context_instance</td>
<td>Sys_id of the record in the table specified in the context parameter to return. Data type: String</td>
</tr>
<tr>
<td>target</td>
<td>Target table associated with the customer context table specified in the context parameter. The combination of the context parameter and target parameter define a unique customer information context. This information is located in the Customer Information Context [sn_customercentral_cust_info_config] table. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>before</td>
<td>Return all configuration records created before the specified date and time. Date type: String Format: YYYY-MM-DD or the format defined in the glide.sys.date_format system property Default: Return all configuration records created before current session time.</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>end</td>
<td>Last record to return. Use this value to paginate record retrieval or use with the <strong>start</strong> parameter to select a specific set of records to return. Data type: Number Default: Last record in file</td>
</tr>
<tr>
<td>start</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks. Do not pass a negative number. Data type: Number Default: 0</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected. Ensure that the path parameters contain valid values.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

**Response body parameters (JSON or XML)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cards_info</td>
<td>List of contact cards associated with the specified customer. Data type: Array (content depends on selected customer)</td>
</tr>
<tr>
<td>cards_info.data_configurations</td>
<td>Configuration information. Data type: Object</td>
</tr>
<tr>
<td>*data_configurations</td>
<td></td>
</tr>
<tr>
<td>*aggregate</td>
<td></td>
</tr>
<tr>
<td>*aggregate_function</td>
<td></td>
</tr>
<tr>
<td>*chart_settings</td>
<td></td>
</tr>
<tr>
<td>*decimal</td>
<td></td>
</tr>
<tr>
<td>*filter</td>
<td></td>
</tr>
<tr>
<td>*group_by</td>
<td></td>
</tr>
<tr>
<td>*number</td>
<td></td>
</tr>
<tr>
<td>*show_other</td>
<td></td>
</tr>
<tr>
<td>*source</td>
<td></td>
</tr>
<tr>
<td>*source_type</td>
<td></td>
</tr>
<tr>
<td>*stack_by_field</td>
<td></td>
</tr>
<tr>
<td>*sys_id</td>
<td></td>
</tr>
<tr>
<td>*trend_field</td>
<td></td>
</tr>
<tr>
<td>*trend_interval</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>cards_info.data_configurations.aggregate_field</td>
<td>Column name to aggregate. Data type: String</td>
</tr>
<tr>
<td>cards_info.data_configurations.aggregate_function</td>
<td>Function to use to aggregate the specified table column. Possible values: AVG, COUNT, COUNT(DISTINCT), SUM. Data type: String</td>
</tr>
<tr>
<td>cards_info.data_configurations.decimal_precision</td>
<td>Number of digits to the right of the decimal point to return. Data type: Number</td>
</tr>
<tr>
<td>cards_info.data_configurations.filter_query</td>
<td>Query string to use to refine the data used in the report. Data type: String</td>
</tr>
<tr>
<td>cards_info.data_configurations.group_by</td>
<td>Name of the column to use to group the report data. Data type: String</td>
</tr>
<tr>
<td>cards_info.data_configurations.number_of_groups</td>
<td>Maximum number of groups to display in the report. Possible values: all: Show all groups., 0: Uses the value in the glide.ui.chart.generate_other system property., Greater than 0: Number of groups to display. The groups that appear are the ones with the highest values for whatever is being tracked. Data type: String</td>
</tr>
<tr>
<td>cards_info.data_configurations.show_other</td>
<td>Flag that indicates whether the group &quot;Other&quot; appears within the report. If not all groups are to display, as defined by the glide.ui.chart.generate_other system property, it is not recommended to change the value of the glide.ui.chart.generate_other system property, as it applies to all reports in the instance. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| **number_of_groups** | Parameter, the values of the non-display groups are combined and displayed under a single heading of “Other”. Possible values:  
- true: Include the “Other” group.  
- false: Do not include the “Other” group.  
Data type: Boolean |
| cards_info.data_configurations.source_id | Table name or sys_id of the report source information. This correlates to the information in the cards_info.reportGroupConfigurations.data_configurations.source_type parameter.  
Data type: String |
| cards_info.data_configurations.source_type | Type of information in the cards_info.reportGroupConfigurations.data_configurations.source_id parameter used to identify the source table. Possible values:  
- source: Sys_id of the source table.  
- table: Table name of the source table.  
Data type: String |
| cards_info.data_configurations.stack_by_field | Name of table column to use when implementing “stack by” in the report.  
Data type: String |
| cards_info.data_configurations.sys_id | Sys_id of the report used in the report configuration.  
Data type: String |
| cards_info.data_configurations.trend_field | Name of table column to use in trend analysis.  
Data type: String |
| cards_info.data_configurations.trend_interval | Time interval for the trend information. For example, week, month, year, etc.  
Data type: String |
| cards_info.data_configurations.type | Type of report, such as single score, donut, vertical, line, stacked bar, etc.  
Data type: String |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cards_info.encodedQuery</td>
<td>Encoded query string to use to refine the data that appears in the report. Data type: String</td>
</tr>
<tr>
<td>cards_info.fields</td>
<td>List of name-value pairs that define the fields within the associated record. Data type: Array</td>
</tr>
<tr>
<td>cards_info.fields.label</td>
<td>Field label. Data type: String</td>
</tr>
<tr>
<td>cards_info.fields.value</td>
<td>Field value. Data type: String</td>
</tr>
<tr>
<td>cards_info.listLimit</td>
<td>Pagination limit. Number of records to show on a list card. Data type: String</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations</td>
<td>List of objects that define the report groups. Data type: Array</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations</td>
<td>Configuration information. Data type: Object</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.aggregate_field</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.aggregate_field</td>
<td>Column name to aggregate.</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.aggregate_function</td>
<td>Function to use to aggregate the specified table column. Possible values: • AVG • COUNT • COUNT(DISTINCT) • SUM</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.decimal_precision</td>
<td>Number of digits to the right of the decimal point to return. Data type: Number</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.filter_query</td>
<td>Query string to use to refine the data used in the report. Data type: String</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.group_by</td>
<td>Name of the column to use to group the report data. Data type: String</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.number_of_groups</td>
<td>Maximum number of groups to display in the report. Possible values: • all: Show all groups • 0: Uses the value in the glide.ui.chart.generate_other system property.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> It is not recommended to change the value of the <code>glide.ui.chart.generate_other</code> system property, as it applies to all reports in the instance.</td>
</tr>
<tr>
<td></td>
<td>Ø Greater than 0: Number of groups to display. The groups that appear are the ones with the highest values for whatever is being tracked.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.show_other</td>
<td>Flag that indicates whether the group &quot;Other&quot; appears within the report. If not all groups are to display, as defined by the <code>number_of_groups</code> parameter, the values of the non-display groups are combined and displayed under a single heading of &quot;Other&quot;. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Include the &quot;Other&quot; group.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not include the &quot;Other&quot; group.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.source_id</td>
<td>Table name or sys_id of the report source information. This correlates to the information in the <code>cards_info.reportGroupConfigurations.data_configurations.source_type</code> parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.source_type</td>
<td>Type of information in the <code>cards_info.reportGroupConfigurations.data_configurations.source_id</code> parameter used to identify the source table. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• source: Sys_id of the source table.</td>
</tr>
<tr>
<td></td>
<td>• table: Table name of the source table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.stack_by_field</td>
<td>Name of the table column to use when implementing &quot;stack by&quot; in the report.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.sys_id</td>
<td>Sys_id of the report used in the report configuration.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.trend_field</td>
<td>Name of table column to use in trend analysis. Data type: String</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.trend_interval</td>
<td>Time interval for the trend information. For example, week, month, year, etc. Data type: String</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.data_configurations.type</td>
<td>Type of report associated with the report group, such as donut or single_core. Data type: String</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.style_configurations</td>
<td>Styles to use within the associated report group. Data type: Object</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.style_configurations.colorPalette</td>
<td>Colors used in the report. Data type: String</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.style_configurations.scoreColor</td>
<td>Color of the score value within the report. Data type: String</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.style_configurations.show_zero</td>
<td>Flag that indicates whether data points equal to zero are shown in the report. Possible values: • true: Zero values are shown. • false: Zero values are not shown. Data type: Boolean</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.title</td>
<td>Title of the report group. Data type: String</td>
</tr>
<tr>
<td>cards_info.reportGroupConfigurations.type</td>
<td>File type for which the information is being returned. Always sn_customercentral_report_config. Data type: String</td>
</tr>
<tr>
<td>cards_info.sourceTableName</td>
<td>Name of the table used for the associated list card.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td>cards_info.style_configurations</td>
<td>Styles to use within the associated report.</td>
</tr>
<tr>
<td>cards_info.style_configurations.colorPalette</td>
<td>Colors used in the report.</td>
</tr>
<tr>
<td>cards_info.style_configurations.dataLabelConfig</td>
<td>Configuration parameters that control how data labels appear in the report.</td>
</tr>
</tbody>
</table>
| cards_info.style_configurations.dataLabelConfig.showTotal   | Flag that indicates whether to display the total in the report. Possible values:  
  - true: Show totals.  
  - false: Do not show totals. |
| cards_info.style_configurations.height                      | Height of the associated report.      |
|                                                            | Data type: String. Unit: Pixels or "auto" |

© 2021 ServiceNow, Inc. All rights reserved.  
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.  
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cards_info.style_configurations.hideRegister</td>
<td>Flag that indicates whether to display the register in the card. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Show the register.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not show the register.</td>
</tr>
<tr>
<td>Data type</td>
<td>Boolean</td>
</tr>
<tr>
<td>cards_info.style_configurations.registerConfig</td>
<td>Configuration values that control how the register appears in the report.</td>
</tr>
<tr>
<td>Data type</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>*registerConfig:</td>
</tr>
<tr>
<td></td>
<td>*hideBorder: Boolean</td>
</tr>
<tr>
<td></td>
<td>*showPercentages: Boolean</td>
</tr>
<tr>
<td>cards_info.style_configurations.registerConfig.hideBorder</td>
<td>Flag that indicates whether to hide the border around the register. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Hide the register border.</td>
</tr>
<tr>
<td></td>
<td>• false: Show the register border.</td>
</tr>
<tr>
<td>Data type</td>
<td>Boolean</td>
</tr>
<tr>
<td>cards_info.style_configurations.registerConfig.showPercentages</td>
<td>Flag that indicates whether to show register percentages. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Show register percentages.</td>
</tr>
<tr>
<td></td>
<td>• false: Hide register percentages.</td>
</tr>
<tr>
<td>Data type</td>
<td>Boolean</td>
</tr>
<tr>
<td>cards_info.style_configurations.registerPosition</td>
<td>Position of the register within a card.</td>
</tr>
<tr>
<td>Data type</td>
<td>String</td>
</tr>
<tr>
<td>cards_info.style_configurations.scoreColor</td>
<td>Color of the score value within the report.</td>
</tr>
<tr>
<td>Data type</td>
<td>String</td>
</tr>
<tr>
<td>cards_info.style_configurations.showDataLabels</td>
<td>Flag that indicates whether to show the data labels within the report. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Show the data labels.</td>
</tr>
<tr>
<td></td>
<td>• false: Hide the data labels.</td>
</tr>
<tr>
<td>Data type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| cards_info.style_configurations.showTooltip | Flag that indicates whether to show tool tips within the report. Possible values:  
- true: Show tool tips.  
- false: Hide tool tips. |
| Data type: Boolean |
| cards_info.style_configurations.show_zero | Flag that indicates whether data points equal to zero are shown in the report. Possible values:  
- true: Zero values are shown.  
- false: Zero values are not shown. |
| Data type: Boolean |
| cards_info.style_configurations.width | Width of the associated report.  
Data type: String  
Unit: Pixels or "auto" |
| cards_info.subheader_fields | Describes the details to show on the sub-header of the associated card.  
Data type: Object  
```
"subheader_fields": {
  "title": "String",
  "subheader_tags": [Array],
  "avatar": "String",
  "sys_id": "String"
}
```
<p>| cards_info.subheader_fields.avatar | File name of the avatar associated with the specified context_instance, such as the contact’s avatar. |
| © 2021 ServiceNow, Inc. All rights reserved. |
| ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated. |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cards_info.subheader_fields.subheader_tags</td>
<td>List of tags associated with the record sub-header, such as escalation tags.</td>
</tr>
<tr>
<td>cards_info.subheader_fields.title</td>
<td>Title to display on the sub-header.</td>
</tr>
<tr>
<td>cards_info.title</td>
<td>Title on the card (record).</td>
</tr>
</tbody>
</table>
| cards_info.type | File type for which the information is being returned. Possible values:  
  - sn_customercentral_list_config: List Configuration  
  - sn_customercentral_record_config: Record Configuration  
  - sn_customercentral_report_config: Report Configuration  
  - sn_customercentral_report_group_config: Report Group Configuration |
<p>| cards_info.viewName | Name of the view configured for the table specified in cards_info.sourceTableName. |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>has_more_records</td>
<td>Flag that indicates whether there are more records to retrieve. Possible values: • true: More records to retrieve. • false: No more records.</td>
</tr>
<tr>
<td>layout</td>
<td>Defines the number of columns per row. Each comma separated value defines the width of a column on the report display. For example &quot;6,6&quot; denotes two columns, each being six units wide, for a total of 12 units per row (maximum). A value of &quot;3,6,3&quot; denotes a row with three columns.</td>
</tr>
<tr>
<td>status</td>
<td>HTTP status of the request.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
  "https://instance.servicenow/api/sn_customercentral/cust_central/custinfo/csm_consumer/a04
88cfbdb1b1200b6075200cf9619db/interaction" \
--request GET \
--header "Accept:application/json" \
--user "username":"password"

{
  "result": {
    "layout": "6,6",
    "cards_info": [ 
      {
        "title": "Consumer",
        "type": "sn_customercentral_record_config",
        "subheader_fields": {
          "title": "Sam Collins",
          "subheader_tags": [],
          "avatar": ",",
          "sys_id": "64488cfbdb1b1200b6075200cf9619db"
        },
        "fields": [ 
```


```json
{
    "label": "Mobile phone",
    "value": ""
},
{
    "label": "Business phone",
    "value": ""
},
{
    "label": "Email",
    "value": "sam.collins@mailinator.com"
},
{
    "label": "Street",
    "value": "144 2nd St"
},
{
    "label": "City",
    "value": "San Francisco"
},
{
    "label": "State / Province",
    "value": "CA"
}
}

{  
    "title": "High priority cases",
    "type": "sn_customercentral_report_group_config",
    "reportGroupConfigurations": [
        {
            "title": "Escalated",
            "type": "sn_customercentral_report_config",
            "data_configurations": {
                "type": "single_score",
                "source_type": "table",
                "source_id": "sn_customerservice_case",
                "group_by": "",
                "trend_interval": "year",
                "trend_field": "",
                "stack_by_field": "",
                "aggregate_function": "COUNT",
                "aggregate_field": ""
            }
        }
    ]
}
```
"filter_query":
"active=true^active_escalationISNOTEMPTY^EQ^consumer=a0488cfbdb1b1200b6075200cf9619db^ORDERBYDESCsys_created_on^sys_created_on>=2020-04-07 13:40:48",

"decimal_precision": 2,
"number_of_groups": 0,
"show_other": true,
"sys_id": "c169a0f50f3b0010e6d4fd820b767e2e"
},
"style_configurations": {
  "scoreColor": "#000000",
  "show_zero": true
}
},
{
  "title": "P1",
  "type": "sn_customercentral_report_config",
  "data_configurations": {
    "type": "single_score",
    "source_type": "table",
    "source_id": "sn_customerservice_case",
    "group_by": "",
    "trend_interval": "year",
    "trend_field": "",
    "stack_by_field": "",
    "aggregate_function": "COUNT",
    "aggregate_field": "",
    "filter_query":
"active=true^priority=1^EQ^consumer=a0488cfbdb1b1200b6075200cf9619db^ORDERBYDESCsys_created_on^sys_created_on>=2020-04-07 13:40:48",

"decimal_precision": 2,
"number_of_groups": 0,
"show_other": true,
"sys_id": "cce86cb50f3b0010e6d4fd820b767eba"
},
"style_configurations": {
  "scoreColor": "#000000",
  "show_zero": true
}
},
{
  "title": "SLA breached",
  "type": "sn_customercentral_report_config",
  "data_configurations": {
    "type": "single_score",
    "source_type": "table",
    "source_id": "sn_customerservice_case",
    "group_by": "",
    "trend_interval": "year",
    "trend_field": "",
    "stack_by_field": "",
    "aggregate_function": "COUNT",
    "aggregate_field": "",
    "filter_query":
"active=true^priority=1^EQ^consumer=a0488cfbdb1b1200b6075200cf9619db^ORDERBYDESCsys_created_on^sys_created_on>=2020-04-07 13:40:48",

"decimal_precision": 2,
"number_of_groups": 0,
"show_other": true,
"sys_id": "cce86cb50f3b0010e6d4fd820b767eba"
}
}
"style_configurations": {
  "width": "auto",
  "height": "auto",
  "colorPalette": [
    "#278ecf",
    "#4bd762",
    "#ffca1f",
    "#ff9416",
    "#d42ae8"
  ],
  "hideRegister": false,
  "registerConfig": {
    "hideBorder": true,
    "showPercentages": false
  },
  "registerPosition": "right",
  "showDataLabels": false,
  "dataLabelConfig": {
    "showTotal": false
  },
  "showTooltip": true
},

{ "title": "Recent cases",
  "type": "sn_customercentral_list_config",
  "listLimit": "3",
  "viewName": "customer_central",
  "encodedQuery": "active=true^consumer=a0488cfd6b1200b6075200cf9619db^ORDERBYDESCsys_created_on^sys_created_on>=2020-04-07 13:40:48",
  "sourceTableName": "sn_customerservice_case"
},

{ "title": "Recent chat interactions",
  "type": "sn_customercentral_list_config",
  "listLimit": "3",
  "viewName": "customer_central",
  "encodedQuery": "active=false^type=chat^consumer=a0488cfd6b1200b6075200cf9619db^ORDERBYDESCsys_created_on^sys_created_on>=2020-04-07 13:40:48",
  "sourceTableName": "interaction"
}
{  
  "title": "Recent phone calls",
  "type": "sn_customercentral_list_config",
  "listLimit": "3",
  "viewName": "customer_central",
  "encodedQuery": "active=false^type=phone^consumer=a0488cfbdb1b1200b6075200cf9619db^ORDERBYDESCsys_created_on^sys_created_on>=2020-04-07 13:40:48",
  "sourceTableName": "interaction"
},
{  
  "title": "Recent viewed articles",
  "type": "sn_customercentral_list_config",
  "listLimit": "3",
  "viewName": "customer_central",
  "encodedQuery": "user=64488cfbdb1b1200b6075200cf9619db^viewed=true^ORDERBYDESCsys_created_on^sys_created_on>=2020-04-07 13:40:48",
  "sourceTableName": "kb_use"
},
{  
  "title": "Contracts",
  "type": "sn_customercentral_list_config",
  "listLimit": "3",
  "viewName": "customer_central",
  "encodedQuery": "active=true^state=Active^consumer=a0488cfbdb1b1200b6075200cf9619db^ORDERBYDESCsys_created_on^sys_created_on>=2020-04-07 13:40:48",
  "sourceTableName": "ast_contract"
},
{  
  "title": "Entitlements",
  "type": "sn_customercentral_list_config",
  "listLimit": "3",
  "viewName": "customer_central",
  "encodedQuery": "active=true^end_date=NULL^ORend_date>=2020-05-07 07:00:00^consumer=a0488cfbdb1b1200b6075200cf9619db^ORDERBYDESCsys_created_on^sys_created_on>=2020-04-07 13:40:48",
  "sourceTableName": "service_entitlement"
}
},
"has_more_records": false,
"status": "200"}
Data Classification API

The Data Classification API enables you to group data by type.

With the Data Classification API, you can:

- Assign data classifications to existing dictionary entries.
- Look up the data classifications for specific dictionary entries.
- Remove all data classifications associated with specific dictionary entries.
- Retrieve a list of all data classifications available in the current domain.

This API requires the Data Classification [com.glide.data_classification] plugin.
For more information, see Data Classification.

Data Classification - GET /data_classification/getAllDataClasses

Returns a list of all data classifications available in the current domain.

Requires the admin, data_classification_admin, or data_classification_auditor role.

URL format

Default URL: /api/now/data_classification/getAllDataClasses

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>

**Response body parameters (JSON or XML)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Result of the request. Returns the sys_id and name for each available data classification. If there are no data classifications, it returns an empty array.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Data classifications can be organized into parent-child relationships. If there are parent data classifications, they are identified in the result. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;parent&quot;: (Object),</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>result.parent</td>
<td>Entry for a parent data classification.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;parent&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>result.parent.sys_id</td>
<td>Sys_id of the parent data classification from the Data Classification [data_classification] table. Data type: String</td>
</tr>
<tr>
<td>result.parent.name</td>
<td>Name of the parent data classification.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.sys_id</td>
<td>Sys_id of the data classification from the Data Classification [data_classification] table. Data type: String</td>
</tr>
<tr>
<td>result.name</td>
<td>Name of the data classification.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

Retrieves a list of all the available data classifications.

```bash
curl "https://instance.servicenow.com/api/now/data_classification/getAllDataClasses" \
  --request GET \
  --header "Accept:application/json" \
  --user 'username': 'password'
```
Data Classification - POST /data_classification/classify

Assigns pre-defined or user-defined data classifications to existing dictionary entries.

Requires the admin or data_classification_admin role.

**URL format**

Default URL: /api/now/data_classification/classify
# Supported request parameters

## Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

## Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

## Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dictionary_entries</td>
<td>Required. The sys_ids of the records you want to classify. The sys_ids are from the Dictionary [sys_dictionary] table. Entered as a comma-separated list enclosed in a string. Data type: String</td>
</tr>
<tr>
<td>data_classes</td>
<td>Required. The sys_ids of the data classifications you want to assign. The sys_ids are from the Data Classification [data_classification] table. Entered as a comma-separated list enclosed in a string. Data type: String</td>
</tr>
</tbody>
</table>

## Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml.</td>
</tr>
</tbody>
</table>
Request headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Message describing the result of the operation.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

Assigns a data classification to a dictionary entry.

```bash
curl "https://instance.servicenow.com/api/now/data_classification/classify" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --data "{"dictionary_entries": "445de0a6dba30300efc57416bf9619b0"","data_classes": "40edb1f51bbce50b92a10a61a4bc8a"}"```
Data Classification - POST /data_classification/clear
Removes all data classifications for the specified dictionary entries.
Requires the admin or data_classification_admin role.

**URL format**
Default URL: /api/now/data_classification/clear

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dictionary_entries</td>
<td>Required. The sys_ids of the records you want to remove classifications from. The sys_ids are from the Dictionary [sys_dictionary] table. Entered as a comma-separated list enclosed in a string. Data type: String</td>
</tr>
</tbody>
</table>

**Headers**
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Message describing the result of the operation. Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

Clears the data classifications for a dictionary entry.
curl "https://instance.servicenow.com/api/now/data_classification/clear" \  
--request POST  
--header "Accept:application/json" \  
--header "Content-Type:application/json" \  
--data "{  
"dictionary_entries": "445de0a6dba30300efc57416bf9619b0"  
}"  
--user 'username': 'password'  

```json  
{  
  "result": "Classifications removed for the specified dictionary entries"  
}  
```

**Data Classification - POST /data_classification/getClassification**

Retrieves all data classifications for the specified dictionary entries.

Requires the admin, data_classification_admin, or data_classification_auditor role.

**URL format**

Default URL: /api/now/data_classification/getClassification

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dictionary_entries</td>
<td>Required. The sys_ids of the records you want to retrieve classifications for. The sys_ids are from the Dictionary [sys_dictionary] table. Entered as a comma-separated list enclosed in a string.</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Result of the request. Returns a JSON object containing each dictionary entry's sys_id with an array of its associated data classifications. If there are no associated data classifications, it returns a message describing the result of the operation. Data classifications can be organized into parent-child relationships. If there are parent data classifications, they are identified in the result. Data type: Object</td>
</tr>
<tr>
<td>result.name</td>
<td>Name of the data classification. Data type: String</td>
</tr>
<tr>
<td>result.parent</td>
<td>Entry for a parent data classification. Data type: Object</td>
</tr>
<tr>
<td>result.parent.name</td>
<td>Name of the parent data classification. Data type: String</td>
</tr>
<tr>
<td>result.parent.sys_id</td>
<td>Sys_id of the parent data classification from the Data Classification [data_classification] table. Data type: String</td>
</tr>
<tr>
<td>result.sys_id</td>
<td>Sys_id of the data classification from the Data Classification [data_classification] table.</td>
</tr>
</tbody>
</table>
### Example: cURL request

Get the data classifications associated with a specific dictionary entry.

```bash
curl "https://instance.servicenow.com/api/now/data_classification/getClassification" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --data "{" \
    "dictionary_entries": "445de0a6dba30300efc57416bf9619b0" \
  }" \
  --user 'username':'password'
```

```json
{
  "result": {
    "445de0a6dba30300efc57416bf9619b0": [
      {
        "parent": {
          "sys_id": "a9670fc773fc1010ae8dd21efaf6a735",
          "name": "Confidential"
        },
        "sys_id": "348107b951d71010f877f3f178e7dd0d",
        "name": "Personally identifiable information"
      }
    ]
  }
}
```

### DevOps API

The DevOps API provides endpoints that enable the interaction with external DevOps tools.

This REST API enables integrators to:

- Retrieve the available DevOps tools associated with an instance.
- Retrieve the schema object for a code resource, plan, or orchestration resource.
- Check if an orchestration task is under change control and check its change control status.
• Create callbacks to associate with task executions under change control.
• Send various event payloads, which are transformed into normalized objects and stored as code, orchestration, and plan objects.
• Search a commit, branch, or repository using encoded queries and specific search/sort criteria.
• Register artifact versions and packages along with their associated pipeline execution and commits.
• Create and manage onboarding application and tool events.

For additional information, see DevOps.

**DevOps - GET /devops/code/schema**

Returns the schema object for a specified code resource: commit, repository, or branch.

Once you have the required schema object, use it to create the request body for your corresponding POST call: POST /code/commit, POST /code/repository, or POST /code/branch.

⚠️ **Note:** This endpoint only returns the request body elements that are required.

**URL format**

Versioned URL: /api/sn_devops/{api_version}/devops/code/schema

Default URL: /api/sn_devops/devops/code/schema

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resource</td>
<td>Required. Type of resource schema to return. Valid values (not case-sensitive): • commit • repository • branch</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
## Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Invalid or missing authentication.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

## Response body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>branch</td>
<td>Description of the branch in which the code commit was made.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;branch&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;path&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>branch.name</td>
<td>Name of the branch in which the code commit was made.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>branch.path</td>
<td>Relative path of the branch.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>committedDate</td>
<td>Date/time on which the code was committed in source DevOps tool.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Format: ISO 8601 with time zone offset</td>
</tr>
<tr>
<td></td>
<td>For example: 1970-01-01T08:15:30-05:00</td>
</tr>
<tr>
<td>committer</td>
<td>Description of the person/entity that requested the commit.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;committer&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;email&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>committer.email</td>
<td>Email address of the person that requested the commit. Data type: String</td>
</tr>
<tr>
<td>details</td>
<td>Details of the commit. Data type: Object</td>
</tr>
<tr>
<td>&quot;details&quot;:</td>
<td></td>
</tr>
<tr>
<td>&quot;action&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;additions&quot;: Number,</td>
<td></td>
</tr>
<tr>
<td>&quot;changes&quot;: Number,</td>
<td></td>
</tr>
<tr>
<td>&quot;deletions&quot;: Number,</td>
<td></td>
</tr>
<tr>
<td>&quot;file&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;totalChanges&quot;: Number</td>
<td></td>
</tr>
<tr>
<td>details.action</td>
<td>Action that was performed on the file. Data type: String</td>
</tr>
<tr>
<td>details.additions</td>
<td>Total number of additions in the file. Data type: Number</td>
</tr>
<tr>
<td>details.changes</td>
<td>Total number of changes that were made. Data type: Number</td>
</tr>
<tr>
<td>details.deletions</td>
<td>Total number of deletions in the file. Data type: Number</td>
</tr>
<tr>
<td>details.file</td>
<td>Path of the file that was modified relative to the branch. Data type: String</td>
</tr>
<tr>
<td>details.totalChanges</td>
<td>Total number of additions and deletions. Data type: Number</td>
</tr>
<tr>
<td>id</td>
<td>Source platform identifier or the commit SHA. Data type: String</td>
</tr>
<tr>
<td>name</td>
<td>Name of the branch, such as &quot;Master&quot;. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>path</td>
<td>Relative path of the branch.</td>
</tr>
<tr>
<td>repository</td>
<td>Description of the repository in which the commit was made.</td>
</tr>
<tr>
<td>repository.name</td>
<td>Name of the repository.</td>
</tr>
<tr>
<td>repository.url</td>
<td>URL of the repository in the source DevOps tool.</td>
</tr>
<tr>
<td>url</td>
<td>URL of the specified resource.</td>
</tr>
</tbody>
</table>

For example for resource=commit: "https://github.com/mycompany/mobileplatform/commit/3fa85f6457174562b3fc2c963f66afa6"
For resource=repository: "https://github.com/mycompany/mobileplatform"

**Example: Sample cURL request for resource=branch**

```bash
curl "https://instance.servicenow.com/api/sn_devops/v1/devops/code/schema?resource=branch" \
  --request GET \
  --header "Accept:application/json" \
  --user "username":"password"
```

```json
{
  "result": {
    "name": "Master",
    "path": "refs/heads/master",
    "repository": {
      "name": "Platform-Mobile",
      "url": "https://github.com/mycompany/mobileplatform"
    }
  }
}
```
Example: Sample cURL request for resource=commit

```
  --request GET
  --header "Accept: application/json"
  --user "username":"password"
```

```json
{
  "result": {
    "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
    "url": "https://github.com/mycompany/mobileplatform/commit/3fa85f6457174562b3fc2c963f66afa6",
    "committedDate": "1970-01-01T08:15:30-05:00",
    "repository": {
      "name": "Platform-Mobile",
      "url": "https://github.com/mycompany/mobileplatform"
    },
    "branch": {
      "name": "master",
      "path": "refs/heads/master"
    },
    "committer": {
      "email": "lenn@smithworksinc.com"
    },
    "details": [
      {
        "additions": 0,
        "deletions": 0,
        "totalChanges": 0,
        "file": "src/test/java/com/mycompany/app/App.java",
        "action": "modified",
        "changes": "A
         public void testAppConstructor()
         {
           App.main(null);
         }

         public void testDatabase()
         {
           app.main();
         }

         @Test
         public void testHelloWorld()
         {
           assertEquals("Hello World!", outContent.toString());
         }
      }
    ]
  }
}
```
Example: Sample cURL request for resource=repository

curl
"https://instance.servicenow.com/api/sn_devops/v1/devops/code/schema?resource=repository"

--request GET
--header "Accept:application/json"
--user "username":"password"

{
  "result": {
    "name": "Platform-Mobile",
    "url": "https://github.com/mycompany/mobileplatform"
  }
}

DevOps - GET /devops/onboarding/status

Returns the current status of the specified onboarding event which is processed or being processed asynchronously by the DevOps service.

URL format

Versioned URL: /api/sn_devops/{api_version}/devops/onboarding/status
Default URL: /api/sn_devops/devops/onboarding/status

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Unique identifier of the onboarding event whose status to return. This value is provided in the return results of the endpoint that invoked the event request, such as /devops/onboarding/app or /devops/onboarding/tool. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
## Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>202</td>
<td>Pending - The onboarding request has been created and the DevOps service is processing the request.</td>
</tr>
<tr>
<td>207</td>
<td>Partial Success - The onboarding request has been created and the DevOps service will process the request.</td>
</tr>
<tr>
<td>400</td>
<td>Failed - The onboarding event failed.</td>
</tr>
<tr>
<td>404</td>
<td>Failed - The endpoint failed to find any request matching the specified event ID.</td>
</tr>
</tbody>
</table>

## Response body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| result                  | Results of the onboarding request.  
Data type: Object |
|                         |{ "result": {  
| result.code             | HTTP status code.  
Data type: String |
| result.importRequestsSysIds | List of import requests created as a part of the onboarding request.  
Data type: Array of Strings |
| result.messageDetails   | Details about the onboarding event progress.  
Data type: Object |
|                         |{ "messageDetails": {  
|                         |"apps": [Array],  
|                         |"pipelines": [Array],  
|                         |"plans": [Array],  
<p>|                         |&quot;repositories&quot;: [Array]}}} |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>result.messageDetails.apps</code></td>
<td>Array of objects that define the apps to create within the associated ServiceNow instance. Data type: Array</td>
</tr>
<tr>
<td><code>result.messageDetails.apps.appId</code></td>
<td>Sys_id of the application that was generated when the application was onboarded. Located in the Apps [sn_devops_app] table. Data type: String</td>
</tr>
<tr>
<td><code>result.messageDetails.apps.message</code></td>
<td>Message that describes the status of the application onboarding. Data type: String</td>
</tr>
<tr>
<td><code>result.messageDetails.apps.name</code></td>
<td>Name of the application being onboarded. Data type: String</td>
</tr>
<tr>
<td><code>result.messageDetails.apps.status</code></td>
<td>Current status of the application onboarding. Possible values: Failed, Partial Success, Success. Data type: String</td>
</tr>
<tr>
<td><code>result.messageDetails.pipelines</code></td>
<td>List of pipelines associated with the onboarding event and their corresponding details. Data type: Array</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.messageDetails.pipelines.changeStepAssociation</td>
<td>Array of objects that describe any change steps that are associated with the onboarding. Data type: Array</td>
</tr>
<tr>
<td>result.messageDetails.pipelines.changeStepAssociation.changeStepName</td>
<td>Name of the associated change step. Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.pipelines.changeStepAssociation.status</td>
<td>Status of change step association to the pipeline. Possible values: Failed, Success. Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.pipelines.changeStepAssociation.summary</td>
<td>Summary of the status of the change step association. Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.pipelines.errors</td>
<td>List of errors detected when associating the pipeline. Data type: Array</td>
</tr>
<tr>
<td>result.messageDetails.pipelines.message</td>
<td>Message that describes the status of the pipeline association. Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.pipelines.name</td>
<td>Name of the pipeline. Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.pipelines.status</td>
<td>Status of the pipeline association. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>result.messageDetails.plans</code></td>
<td>List of plans associated with the onboarding event and their corresponding details. Data type: Array</td>
</tr>
<tr>
<td><code>result.messageDetails.plans.errors</code></td>
<td>List of errors detected when associating the plan. Data type: Array</td>
</tr>
<tr>
<td><code>result.messageDetails.plans.message</code></td>
<td>Message that describes the status of the pipeline association. Data type: String</td>
</tr>
<tr>
<td><code>result.messageDetails.plans.name</code></td>
<td>Name of the plan. Data type: String</td>
</tr>
<tr>
<td><code>result.messageDetails.plans.status</code></td>
<td>Status of the plan association. Data type: String</td>
</tr>
<tr>
<td><code>result.messageDetails.repositories</code></td>
<td>List of repositories associated with the onboarding event and their corresponding details. Data type: Array</td>
</tr>
<tr>
<td><code>result.messageDetails.repositories.configureStatus</code></td>
<td>Details of the webhook configuration status for the repository. Data type: Object</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.messageDetails.repositories.configureStatus.message</td>
<td>Message that describes the current webhook configuration status. Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.repositories.configureStatus.status</td>
<td>Status of the webhook configuration. Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.repositories.message</td>
<td>Message that describes the current repository association status. Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.repositories.name</td>
<td>Name of the repository. Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.repositories.status</td>
<td>Status of the repository association. Data type: String</td>
</tr>
</tbody>
</table>
| result.messageDetails.status | Status of the onboarding request. Possible values:
  - Failed
  - Partial Success
  - Success Data type: String |
| result.messageDetails.toolResponse | Array of objects that define the tools to onboard within the associated ServiceNow instance. Data type: Array |

"toolResponse": [
  {
    "configureStatus": {},
    "connectStatus": {},
    "createStatus": {},
    "discoveryStatus": {},
    "name": "String",
    "status": "String",
    "type": "String"
  }
]
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.messageDetails.toolResponse.configureStatus</td>
<td>Describes the configuration status of the tool. In case of success, it returns the <code>status</code> and <code>message</code>. In case of error, it returns the <code>error</code> and <code>message</code>. Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.configureStatus.error</td>
<td>Errors detected during tool configuration. Data type: String</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.configureStatus.message</td>
<td>Message that describes the successful status of the tool configuration. Data type: String</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.configureStatus.status</td>
<td>Status of the tool configuration. Data type: String</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.connectStatus</td>
<td>Describes the connection status of the tool. In case of success, it returns the <code>status</code> and <code>message</code>. In case of error, it returns the <code>error</code> and <code>message</code>. Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.connectStatus.error</td>
<td>Errors detected during tool connection. Data type: String</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.connectStatus.message</td>
<td>Message that describes the successful status of the tool connection. Data type: String</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.connectStatus.status</td>
<td>Status of the tool connection. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.createStatus</td>
<td>Describes the creation status of the tool. In case of success, it returns the status and message. In case of error, it returns the status and error. Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.createStatus.error</td>
<td>Errors detected during tool creation. Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.createStatus.message</td>
<td>Message that describes the successful status of the tool creation. Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.createStatus.status</td>
<td>Status of the tool creation. Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.discoveryStatus</td>
<td>Describes the discovery status of the tool. In case of success, it returns the status and message. In case of error, it returns the status and error. Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.discoveryStatus.error</td>
<td>Errors detected during tool discovery. Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.discoveryStatus.imports</td>
<td>Array of objects that describe each entity that was discovered. Data type: Array</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;imports&quot;: [{</td>
<td>Details that describe the import performed.</td>
</tr>
<tr>
<td>&quot;details&quot;: &quot;String&quot;</td>
<td>Data type: String</td>
</tr>
<tr>
<td>&quot;id&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;status&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.discoveryStatus.imports.id</td>
<td>Sys_id of the import request. Located in the sn_devops_m2m_inbound_event table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.discoveryStatus.imports.status</td>
<td>Status of the import request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.discoveryStatus.message</td>
<td>Message that describes the status of the tool discovery.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.discoveryStatus.status</td>
<td>Status of the tool discovery.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.name</td>
<td>Name of the tool being onboarded.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.status</td>
<td>Status of the tool onboarding.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.messageDetails.toolResponse.type</td>
<td>Type of tool, such as Jira or Jenkins.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.onboardingRequestID</td>
<td>Unique identifier of the onboarding request. Use this value when calling the /devops/onboarding/status endpoint to obtain the status of the associated tool onboarding process.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.status</td>
<td>Status of the onboarding request.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
</tbody>
</table>
**Example: cURL request**

The following example shows the status request of a tool onboarding.

```bash
curl -X GET
   "https://servicenow-instance/api/sn_devops/v1/devops/onboarding/status?id=IBE0001603" \
   -H "Accept: application/json" \
   -H "Content-Type: application/json" \
   -u "username":"password" \
```

```json
{
   "result": {
      "code": "207",
      "onboardingRequestID": "IBE0001603",
      "status": "partial-success",
      "messageDetails": {
         "status": "Partial Success",
         "toolResponse": [
            {
               "name": "Primary Azure",
               "type": "Azure DevOps",
               "status": "Partial Success",
               "createStatus": { 
                  "status": "Success",
                  "message": "Created successfully"
               },
               "connectStatus": {
                  "status": "Failed",
                  "error": "Method failed: (/testFirst/_apis/work/boards) with code: 401 - Invalid username/password combo"
               }
            },
            {
               "name": "Primary Jenkins",
               "type": "Jenkins",
               "status": "Success",
            }
         ]
      }
   }
}
```
"createStatus": {
    "status": "Success",
    "message": "Created successfully"
},
"connectStatus": {
    "status": "Success",
    "message": "Connected successfully"
},
"discoveryStatus": {
    "status": "Success",
    "message": "Discovered successfully",
    "imports": [
        {
            "status": "completed",
            "id": "2a4d216db2d2450fe4fd6a75e9619c3",
            "details": "Discovered 1 new job(s) and 3 new pipeline(s)"
        }
    ]
},
"name": "Primary GitHub",
"type": "GitHub",
"status": "Success",
"createStatus": {
    "status": "Success",
    "message": "Created successfully"
},
"connectStatus": {
    "status": "Success",
    "message": "Connected successfully"
},
"discoveryStatus": {
    "status": "Success",
    "message": "Discovered successfully",
    "imports": [
        {
            "status": "completed",
            "id": "7f4d616db2d2450fe4fd6a75e9619c2",
            "details": "Discovered 5 new repositories."
        }
    ]
}
{
  "name": "Primary Bitbucket",
  "type": "BitBucket",
  "status": "Success",
  "createStatus": {
    "status": "Success",
    "message": "Created successfully"
  },
  "connectStatus": {
    "status": "Success",
    "message": "Connected successfully"
  },
  "discoveryStatus": {
    "status": "Success",
    "message": "Discovered successfully",
    "imports": [
      {
        "status": "completed",
        "id": "815da16bdb2d2450fe4fd6a75e9619d0",
        "details": "Discovered 11 new repositories."
      }
    ]
  }
},
{
  "name": "Primary Gitlab",
  "type": "GitLab",
  "status": "Partial Success",
  "createStatus": {
    "status": "Success",
    "message": "Created successfully"
  },
  "connectStatus": {
    "status": "Failed",
    "error": "The current operation ended in state: ERROR"
  }
},
"importRequestsSysIds": [
  "2a4d216bdb2d2450fe4fd6a75e9619c3",
  "7f4d616bdb2d2450fe4fd6a75e9619c2",
  "815da16bdb2d2450fe4fd6a75e9619d0"
]
Example: cURL request

The following example shows the status request of an application onboarding.

curl -X POST
  "https://servicenow-instance/api/sn_devops/v1/devops/onboarding/status?onboardingRequestID=IBE0001604" \
  -H "Accept: application/json" \
  -H "Content-Type: application/json" \
  -u "username":"password" \


```json
{
    "result": {
        "code": "207",
        "onboardingRequestID": "IBE0001604",
        "status": "partial-success",
        "messageDetails": {
            "status": "Partial Success",
            "apps": [
                {
                    "name": "Bike Rentals Service",
                    "status": "Success",
                    "appId": "c7fd696bdb2d2450fe4fd6a75e961993",
                    "message": "App by name 'Bike Rentals Service' has been successfully created"
                }
            ],
            "pipelines": [
                {
                    "name": "Bike Rentals",
                    "status": "Success",
                    "message": "Successfully associated with app 'Bike Rentals Service'",
                    "changeStepAssociation": [
                        {
                            "status": "Success",
                            "changeStepName": "Prod",
                            "summary": "Change step with name 'Prod' has been created"
                        }
                    ]
                }
            ]
        }
    }
}
DevOps - GET /devops/orchestration/changeControl
Checks if the orchestration task is under change control.

URL format
Versioned URL: /api/sn_devops/{api_version}/devops/orchestration/changeControl
Default URL: /api/sn_devops/devops/orchestration/changeControl

Supported request parameters

| Path parameters |
|-----------------|--------------------------------------------------|
| Name            | Description                                      |
| api_version     | Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. |
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>orchestrationTaskName</td>
<td>Name of the orchestration task to check.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>orchestrationTaskURL</td>
<td>Required. URL of the orchestration task to check.</td>
</tr>
<tr>
<td></td>
<td>Note: Must be percent (%) encoded, such as https%3A%2F%2Fjenkins.mycompany.com %3A8080%2Fjob%2FMobile-Platform-deploy%2F.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>testConnection</td>
<td>Flag that indicates whether to test the end-to-end connection between the instance and the DevOps tool. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Test the end-to-end connection.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not test the end-to-end connection.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>toolType</td>
<td>Type of tool. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• jenkins</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Invalid or missing authentication.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| `changeControl`    | Flag that indicates whether the orchestration task is under change control.
|                    | Possible values:                                                            |
|                    | • true: Under change control.                                               |
|                    | • false: Not under change control                                          |
| Data type: Boolean |                                                                            |

Example: Sample cURL request

```bash
curl -X GET
 "https://instance.servicenow.com/api/sn_devops/v1/devops/orchestration/changeControl?toolId=f0ca45679323008b52f3b457415ae6&orchestrationTaskName=Platform-Analytics-UAT-Deployment&orchestrationTaskURL=https%253A%252F%252Fjenkins.mycompany.com%25252Fjob%25252FMobile-Platform-deploy%252F&toolType=jenkins"
-H "accept: application/json"
-u "username":"password"
```

Under change control

```
{
 "result": {
  "changeControl": true,
  "status": "Success"
 }
}
```

Example: Sample cURL request

```bash
curl -X GET
 "https://instance.servicenow.com/api/sn_devops/v1/devops/orchestration/changeControl?toolId=f0ca45679323008b52f3b457415ae6&orchestrationTaskName=Platform-Analytics-UAT-Deployment&orchestrationTaskURL=https%253A%252F%252Fjenkins.mycompany.com%25252Fjob%25252FMobile-Platform-deploy%252F&toolType=jenkins"
-H "accept: application/json"
-u "username":"password"
```

Not under change control

```
{
 "result": {
```
© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**DevOps - GET /devops/orchestration/changeInfo**

Retrieves change request details for a specified orchestration pipeline execution.

**URL format**

Versioned URL: /api/sn_devops/{api_version}/devops/orchestration/changeInfo
Default URL: /api/sn_devops/devops/orchestration/changeInfo

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>branchName</td>
<td>Name of the branch associated with the orchestration task. <strong>Note:</strong> You should provide this parameter when referencing multi-branch pipelines or the response may include incorrect change details. Data type: String</td>
</tr>
<tr>
<td>buildNumber</td>
<td>Required. Pipeline build number within the orchestration tool. Data type: String</td>
</tr>
<tr>
<td>pipelineName</td>
<td>Optional for Jenkins Freestyle or Maven projects, required for all others. Orchestration pipeline name. Data type: String</td>
</tr>
<tr>
<td>projectName</td>
<td>Required for Jenkins Freestyle or Maven projects, optional for all others. Orchestration tool project name.</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stageName</td>
<td>Optional for Jenkins Freestyle or Maven projects, required for all others. Orchestration stage name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
## Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| changeFound   | Flag that indicates whether the change request was located. Possible values:  
• true: Change request was found.  
• false: Change request was not found.  
Data type: Boolean |
| number        | Change request number.  
Data type: String |
| phase         | Current phase.  
Data type: String |
| phase_state   | State of the current phase.  
Data type: String |
| priority      | Priority of the change request. Priority is based on the impact and urgency of the request. It identifies how quickly the service desk should address the request.  
Possible values:  
• 1: Critical  
• 2: High  
• 3: Moderate  
• 4: Low  
Data type: String |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>risk</td>
<td>Level of risk for the change. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Low</td>
</tr>
<tr>
<td></td>
<td>• High</td>
</tr>
<tr>
<td></td>
<td>• Moderate</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>State of the change request. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Assess</td>
</tr>
<tr>
<td></td>
<td>• Authorize</td>
</tr>
<tr>
<td></td>
<td>• Canceled</td>
</tr>
<tr>
<td></td>
<td>• Closed</td>
</tr>
<tr>
<td></td>
<td>• Implement</td>
</tr>
<tr>
<td></td>
<td>• New</td>
</tr>
<tr>
<td></td>
<td>• Review</td>
</tr>
<tr>
<td></td>
<td>• Scheduled</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: New</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the change request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Type of change request. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Emergency</td>
</tr>
<tr>
<td></td>
<td>• Normal</td>
</tr>
<tr>
<td></td>
<td>• Standard</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>url</td>
<td>URL of the change request within the associated ServiceNow instance.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

```
curl -X GET
  "https://instance.servicenow.com/api/sn_devops/v1/devops/orchestration/changeInfo?toolId=487f457db642810765555535e9619d3&pipelineName=Automation/apps HR/family/release&stageName=deployfamilyrelease&buildNumber=1639"
  -H "accept: application/json"
  -u "username":"password"
```

```json
{
    "result": {
        "changeFound": true,
        "sys_id": "ee89341bdb642810765555535e96196e",
        "number": "CHG0030001",
        "type": "normal",
        "state": "0",
        "phase": "requested",
        "phase_state": "open",
        "priority": "4",
        "risk": "4",
        "url": "https://instance.servicenow.com/change_request.do?sys_id=ee89341bdb642810765555535e96196e"
    }
}
```

DevOps - GET /devops/orchestration/schema

Returns the schema object for a specified orchestration resource.

Once you have the required schema object, use it to create the request body for your corresponding POST call: /orchestration/orchestrationTask or /orchestration/taskExecution.

⚠️ Note: This endpoint only returns the request body elements that are required.

**URL format**

Versioned URL: /api/sn_devops/{api_version}/devops/orchestration/schema

Default URL: /api/sn_devops/devops/orchestration/schema
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resource</td>
<td>Type of resource schema to return. Valid values (not case-sensitive):</td>
</tr>
<tr>
<td></td>
<td>• build_details</td>
</tr>
<tr>
<td></td>
<td>• callback</td>
</tr>
<tr>
<td></td>
<td>• orchestration_task</td>
</tr>
<tr>
<td></td>
<td>• task_execution</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>401</td>
<td>Invalid or missing authentication.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

**Response body parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callbackURL</td>
<td>URL of the callback. Data type: String</td>
</tr>
<tr>
<td>endDateTime</td>
<td>Date/time that the task execution ended. Data type: String</td>
</tr>
<tr>
<td>name</td>
<td>Name of the orchestration task. Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Task execution number. Same as an incident number. Data type: String</td>
</tr>
</tbody>
</table>
| orchestrationTaskDetails | Description of the orchestration task. Data type: Object 

```json
"orchestrationTaskDetails": {
    "message": "String",
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>orchestrationTaskDetails.message</td>
<td>Free-form text message. Data type: String</td>
</tr>
<tr>
<td>orchestrationTaskDetails.name</td>
<td>Name of the orchestration task. Data type: String</td>
</tr>
<tr>
<td>orchestrationTaskDetails.triggerType</td>
<td>The way the orchestration task was started. Data type: String</td>
</tr>
<tr>
<td>orchestrationTaskDetails.url</td>
<td>URL of the associated orchestration tool's orchestration task. Data type: String</td>
</tr>
<tr>
<td>orchestrationTaskURL</td>
<td>URL of the orchestration task. Data type: String</td>
</tr>
<tr>
<td>result</td>
<td>Execution result of the task execution. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• Success</td>
</tr>
<tr>
<td></td>
<td>• Failed</td>
</tr>
<tr>
<td>startDateTime</td>
<td>Date/time that the task execution started. Data type: String</td>
</tr>
<tr>
<td>taskExecutionURL</td>
<td>URL of the task to execute. Data type: String</td>
</tr>
<tr>
<td>triggerType</td>
<td>The way the orchestration task was started. Valid values:</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>• upstream</td>
<td>Upstream job triggered this job.</td>
</tr>
<tr>
<td>• user</td>
<td>User manually started the job.</td>
</tr>
<tr>
<td>• scm</td>
<td>Git/scm tool code commit triggered the job.</td>
</tr>
<tr>
<td>Data type</td>
<td>String</td>
</tr>
<tr>
<td>url</td>
<td>URL of the associated orchestration tool's orchestration task.</td>
</tr>
<tr>
<td>Data type</td>
<td>String</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request with resource=task_execution**

```
curl -X GET 
'https://instance.servicenow.com/api/sn_devops/v1/devops/orchestration/schema?resource=task_execution' 
-H 'Accept: application/json'
-u 'username':'password'
```

```
{
"result": {
 "number": 40,
 "url": "https://jenkins.mycompany.com:8080/job/Mobile-Platform-CI/40/",
 "startDateTime": "1970-01-01T08:15:30-05:00",
 "endDateTime": "1970-01-01T08:25:30-05:00",
 "triggerType": "upstream",
 "result": "Success",
 "orchestrationTaskDetails": {
 "name": "Mobile-Platform-CI",
 "url": "https://jenkins.mycompany.com:8080/job/Mobile-Platform-CI/"
 }
}
```

**Example: Sample cURL request with resource=orchestration_task**

```
curl -X GET 
-H 'Accept: application/json'
-u 'username':'password'
```
Example: Sample cURL request with resource=callback

curl -X GET \\
'https://instance.servicenow.com/api/sn_devops/v1/devops/orchestration/schema?resource=task_execution' \\
-H 'Accept: application/json' \\
-u 'username': 'password'

{
 "result": {
  "callbackURL": "http://127.0.0.1:8090/jenkins/sn-devops/9b0feb79-f0b9-4661-83ef-2861b8924784",
  "orchestrationTaskURL": "http://127.0.0.1:8090/jenkins/job/felipe-downstream/",
  "orchestrationTaskDetails": {
   "message": "Started by upstream project \"felipe-upstream\" build number 27",
   "triggerType": "upstream"
  }
 }
}

DevOps - GET /devops/orchestration/stepMapping

Verifies that the information being passed is valid for the creation of an orchestration task and is valid for association to a previously created step.

URL format

Versioned URL: /api/sn_devops/{api_version}/devops/orchestration/stepMapping

Default URL: /api/sn_devops/devops/orchestration/stepMapping
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
<td>String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>branchName</td>
<td>Name of the branch associated with the orchestration task.</td>
<td>String</td>
</tr>
<tr>
<td>isMultiBranch</td>
<td>If the orchestration tool is Jenkins, flag that indicates whether the project is of type Multibranch. Valid values:</td>
<td>Boolean</td>
</tr>
<tr>
<td></td>
<td>• true: Project is of type Multibranch.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• false: Project is not of type Multibranch.</td>
<td></td>
</tr>
<tr>
<td>parentStageName</td>
<td>If the orchestration task is a child nested inside a parent orchestration task, the name of the parent orchestration task.</td>
<td>String</td>
</tr>
<tr>
<td>parentStageURL</td>
<td>If the orchestration task is a child nested inside a parent orchestration task, the URL of the parent orchestration task.</td>
<td>String</td>
</tr>
<tr>
<td>orchestrationTaskName</td>
<td>Required. Name of the orchestration task.</td>
<td>String</td>
</tr>
<tr>
<td>orchestrationTaskURL</td>
<td>Required. URL of the orchestration task.</td>
<td>String</td>
</tr>
</tbody>
</table>
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Note:</strong> Must be percent (%) encoded, such as https%3A%2F%2Fjenkins.mycompany.com%3A8080%2Fjob%2FMobile-Platform-deploy%2F.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>testConnection</td>
<td>Flag that indicates whether to test the end-to-end connection between the instance and the DevOps tool. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Test the end-to-end connection.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not test the end-to-end connection.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>toolType</td>
<td>Required. Type of tool. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• jenkins</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>Flag that indicates whether the verification was successful.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>stepValid</td>
<td>Flag that indicates whether the information sent is valid.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
</tbody>
</table>

Example: Sample cURL request

```bash
curl -X GET
"https://instance.servicenow.com/api/sn_devops/v1/devops/orchestration/stepMapping?toolId=f0ca45679323008b52f3b457415ae6&orchestrationTaskName=Platform-Analytics-UAT-Deployment&orch"```
Results if information is valid.

```
{
    "result": {
        "stepValid": true,
        "status": "Success"
    }
}
```

Example: Sample cURL request

```
curl -X GET
"https://instance.servicenow.com/api/sn_devops/v1/devops/orchestration/stepMapping?toolId=f0ca45679323008b52f3b457415ae6&orchestrationTaskName=Platform-Analytics-UAT-Deployment&orchestrationTaskURL=https%253A%252F%252Fjenkins.mycompany.com%253A8080%252Fjob%252FMobile-Platform-deploy%252F&toolType=jenkins "
-H "Accept: application/json" \ 
-u "username":"password"
```

Results if information is invalid.

```
{
    "result": {
        "stepValid": false,
        "status": "Success"
    }
}
```

DevOps - GET /devops/plan/schema

Returns the schema object for a specific plan: app, version, or work item.

Once you have the required schema object, use it to create the request body for your corresponding POST call; POST /plan/app, POST /plan/version, or POST /plan/workitem.

⚠️ Note: This endpoint only returns the request body elements that are required.

URL format

Versioned URL: /api/sn_devops/{api_version}/devops/plan/schema
Default URL: /api/sn_devops/devops/plan/schema
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resource</td>
<td>Type of resource schema to return. Valid values (not case-sensitive):</td>
</tr>
<tr>
<td></td>
<td>• app</td>
</tr>
<tr>
<td></td>
<td>• version</td>
</tr>
<tr>
<td></td>
<td>• work_item</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Invalid or missing authentication.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| app               | Description of the application to associate with the version.  
Data type: Object  

```
"app": {
   "createdDateTime": "String",
   "id": "String",
   "shortDescription": "String",
   "team": {Object},
   "url": "String"
}
```
| app.createdDateTime | Date/time the application was created in the source DevOps tool.  
Data type: String |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app.id</td>
<td>Unique identifier of the application. Generated by the source planning tool. Data type: String</td>
</tr>
<tr>
<td>app.shortDescription</td>
<td>Brief description of the associated application. Data type: String</td>
</tr>
<tr>
<td>app.team</td>
<td>Description of the team associated with the application. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;team&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>app.team.id</td>
<td>Unique identifier of the team associated with the application. Generated by the source planning tool. Data type: String</td>
</tr>
<tr>
<td>app.team.name</td>
<td>Name of the team associated with the application. Data type: String</td>
</tr>
<tr>
<td>app.url</td>
<td>URL of the application on the source planning tool site. Data type: String</td>
</tr>
<tr>
<td>assignedTo</td>
<td>Description of the person to which the work item is assigned. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;assignedTo&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;email&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;userName&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>assignedTo.email</td>
<td>Email address of the person assigned to the work item.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>assignedTo.id</td>
<td>Unique identifier of the person assigned to the work item. Data type: String</td>
</tr>
<tr>
<td>assignedTo.name</td>
<td>Name of the person assigned to the work item. Data type: String</td>
</tr>
<tr>
<td>assignedTo.userName</td>
<td>User name of the person assigned to the work item. Data type: String</td>
</tr>
<tr>
<td>createDateTime</td>
<td>Date and time that the application, version, or work item record was created. Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>Unique identifier of the application, version, or work item. Generated by the source planning tool. Data type: String</td>
</tr>
<tr>
<td>name</td>
<td>Name of the application. Data type: String</td>
</tr>
<tr>
<td>path</td>
<td>Relative path to the application. Data type: String</td>
</tr>
<tr>
<td>shortDescription</td>
<td>Brief description of the application, version, or work item. Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>State of the work item. Generated by the source planning tool. Data type: String</td>
</tr>
<tr>
<td>team</td>
<td>Description of the team associated with the application. Data type: Object</td>
</tr>
</tbody>
</table>

```
"team": {
  "id": "String",
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>team.id</td>
<td>Unique identifier of the team associated with the application. Generated by the source planning tool. Data type: String</td>
</tr>
<tr>
<td>team.name</td>
<td>Name of the team associated with the application. Data type: String</td>
</tr>
<tr>
<td>type</td>
<td>Type of work item. Generated by the planning tool. Data type: String</td>
</tr>
<tr>
<td>url</td>
<td>URL to use to access the application, version, or work item. Data type: String</td>
</tr>
<tr>
<td>version</td>
<td>Array that describes the version. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;version&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;app&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;createdDateTime&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;shortDescription&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>version.app</td>
<td>Description of the application to associate with the version. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;app&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;createdDateTime&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;shortDescription&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;team&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;url&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>version.app.createdDateTime</td>
<td>Date/time that the version was created on the source DevOps tool.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>version.app.id</td>
<td>Unique identifier of the application. Generated by the source planning tool.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>version.app.shortDescription</td>
<td>Brief description of the associated application.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>version.app.team</td>
<td>Description of the team associated with the application.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;team&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>version.app.team.id</td>
<td>Unique identifier of the team associated with the application. Generated by the source planning tool.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>version.app.team.name</td>
<td>Name of the team associated with the application.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>version.app.url</td>
<td>URL of the application on the source planning tool site.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>version.createdDateTime</td>
<td>Date/time that the version was created on the source DevOps tool.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>version.id</td>
<td>Unique identifier of the version. Generated by the source planning tool.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>version.shortDescription</td>
<td>Brief description of the associated version.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>version.url</td>
<td>URL of the version on the source planning tool site. Data type: String</td>
</tr>
</tbody>
</table>

Example: Sample cURL request for resource=app

```bash
curl "https://instance.servicenow.com/api/sn_devops/v1/devops/plan/schema?resource=app" \
  --request GET \
  --header "Accept:application/json" \
  --user "username":"password"
```

```json
{
  "result": {
    "id": "PRODUCT1234",
    "shortDescription": "Mobile UI",
    "createdDateTime": "1970-01-01T08:15:30-05:00",
    "team": {
      "name": "Mobile UI",
      "id": "f0ca45679323008b52f3b457415ae6"
    },
    "url": "https://jira.com/mycompany/browse/PRODUCT-125"
  }
}
```

Example: Sample cURL request for resource=version

```bash
curl "https://instance.servicenow.com/api/sn_devops/v1/devops/plan/schema?resource=version" \
  --request GET \
  --header "Accept:application/json" \
  --user "username":"password"
```

```json
{
  "result": {
    "id": "REL1234",
    "shortDescription": "APIs Release",
    "createdDateTime": "1970-01-01T08:15:30-05:00",
    "app": {
      "id": "PRODUCT1234",
      "shortDescription": "Mobile UI",
      "createdDateTime": "1970-01-01T08:15:30-05:00",
      "team": {
```
Example: Sample cURL request for resource=work_item

curl
"https://instance.servicenow.com/api/sn_devops/v1/devops/plan/schema?resource=work_item" \
--request GET \
--header "Accept:application/json" \
--user "username":"password"

{
 "result": {
  "id": "STR1234",
  "type": "Story",
  "shortDescription": "Planning API Spec",
  "state": "In-progress",
  "createdDateTime": "1970-01-01T08:15:30-05:00",
  "assignedTo": {
    "name": "Leo Neo",
    "userName": "lenn",
    "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
    "email": "lenn@smithworksinc.com"
  },
  "version": {
    "id": "REL1234",
    "shortDescription": "APIs Release",
    "createdDateTime": "1970-01-01T08:15:30-05:00",
    "app": {
      "id": "PRODUCT1234",
      "shortDescription": "Mobile UI",
      "createdDateTime": "1970-01-01T08:15:30-05:00",
      "team": {
        "name": "Mobile UI",
        "id": "f0ca45679323008b52f3b457415ae6"
      },
      "url": "https://jira.com/mycompany/browse/PRODUCT-125"
    }
  }
}
DevOps - POST /devops/artifact/registration

Enables orchestration tools to register artifacts into a ServiceNow instance.

This endpoint creates new artifacts and artifact versions, and associates them to commits. The following describes the endpoint process flow:

- If the toolId is passed in the request, the endpoint validates that the tool exists within the instance. It also checks if the artifact repository specified in artifacts.repositoryName exists, and if not, creates it under the associated artifact tool.

- If the toolId is not passed in the request, a repository is created but it is not linked to any artifact tool.

- It then checks if the artifact specified in artifacts.name already exists, and if not, creates it under the associated repository.

- If the artifact.version is passed in the request:
  - If it does not exist, a new artifact version record is created in the associated Artifact Version [sn_devops_artifact_version] table.
  - The endpoint then looks for associated task executions based on the provided build details (pipelineName or projectName, taskExecutionNumber, branchName, and stageName) and the orchestrationToolId parameter.
  - If a task execution is found, and its post-processing is complete, the artifact version is linked to the task execution and stored in the Artifact to TaskExecution [sn_devops_m2m_artifact_execution] table. Commits linked to these task executions are also associated to the artifact version and stored in the Artifact Version to Commit [sn_devops_m2m_artifact_version_commit] table. Any commits that are
linked to previous task executions that did not create an artifact, are also associated with the artifact version.

- If a task execution is found, and its post-processing is not complete, then the request is persisted in the Artifact Staged Request [sn_devops_artifact_staging] table. The staged request contains a reference to the task execution.
- If a task execution is not found, the request is persisted in the staging table. The staged request contains the build details.

- If the `artifact.version` is not passed in, then depending on the tool configuration, the artifact version is either retrieved by calling the `/devops/tool/artifact` endpoint if your artifact tool supports webhooks or through a custom subflow that retrieves the artifact version from the artifact tool.

For additional information on artifacts, see Using DevOps change acceleration for releases.

**URL format**

Versioned URL: /api/sn_devops/{api_version}/devops/artifact/registration

Default URL: /api/sn_devops/devops/artifact/registration

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>orchestrationToolId</td>
<td>Required. Sys_id of the orchestration tool associated with the artifact. Located in the Orchestration Tool [sn_devops_orchestration_tool] table. The endpoint uses this information to locate task execution information.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>toolId</td>
<td>Sys_id of the artifact tool associated with the artifact. Located in the Artifact Tool [sn_devops_artifact_tool] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>artifacts</td>
<td>At least one entry required. Array of objects in which each object uniquely identifies an artifact version to register.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>artifacts.name</td>
<td>Required. Artifact file name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160 characters</td>
</tr>
<tr>
<td>artifacts.repositoryName</td>
<td>Required. Artifact repository name. If the specified repository record does not exist, it is created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160 characters</td>
</tr>
<tr>
<td>artifacts.semanticVersion</td>
<td>Semantic version of the artifact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Format: MAJOR.MINOR.PATCH</td>
</tr>
<tr>
<td></td>
<td>Examples of possible values:</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>artifacts.version</td>
<td>Required if <strong>toolId</strong> is not passed. Artifact version.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Format: MAJOR.MINOR</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160 characters</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>branchName</td>
<td>Required if it is a multi-branch project. Name of the branch in the repository where the artifact is stored.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>pipelineName</td>
<td>Required if <strong>projectName</strong> is not passed. Pipeline name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>projectName</td>
<td>Required if <strong>pipelineName</strong> is not passed. Free-form project name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>stageName</td>
<td>Required if <strong>pipelineName</strong> is passed. Stage name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Success. Success with response body.</td>
</tr>
<tr>
<td>201</td>
<td>Created. Success with response body.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. The request URI does not match the APIs in the system, or the operation failed for unknown reasons. Invalid headers can also cause this error.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>response</td>
<td>Results of the artifact registration request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>```json</td>
</tr>
<tr>
<td></td>
<td>&quot;response&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;artifact&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;artifact_register_requests&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;artifact_repository&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;artifact_version&quot;: {Object}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>response.artifact</td>
<td>Results for each processed artifact.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>```json</td>
</tr>
<tr>
<td></td>
<td>&quot;artifact&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;create&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;found&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;update&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>response.artifact.create</td>
<td>If the request created artifact records, the list of sys_ids of the created records in the Artifacts [sn_devops_artifact] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>```json</td>
</tr>
<tr>
<td></td>
<td>&quot;create&quot;: [</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>response.artifact.create.id</td>
<td>Sys_id of the artifact record created by the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>response.artifact.found</td>
<td>If an artifact already exists in the Artifacts table, the list of sys_ids of the existing artifact records.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> No modifications are made to existing packages.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>response.artifact.found.id</td>
<td>Sys_id of the existing artifact record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>response.artifact.update</td>
<td>Currently unused</td>
</tr>
<tr>
<td>response.artifact.update.id</td>
<td>Currently unused</td>
</tr>
<tr>
<td>response.artifact_register_requests</td>
<td>Results for any artifacts register requests that have not completed processing and have been stored in the Artifact Staged Request table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
</tbody>
</table>

```json
"artifact_register_requests": {
"create": [Array],
"found": [Array],
"update": [Array]
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>response.artifact_register_requests.create</td>
<td>If the request created artifact staging records, the list of sys_ids of the created records. Data type: Array</td>
</tr>
<tr>
<td>response.artifact_register_requests.create.id</td>
<td>Sys_id of the artifact staging record created by the request. Data type: String</td>
</tr>
<tr>
<td>response.artifact_register_requests.found</td>
<td>If artifacts already have associated artifact staging records, the list of sys_ids of the existing artifact staging records. <strong>Note:</strong> No modifications are made to existing packages. Data type: Array</td>
</tr>
<tr>
<td>response.artifact_register_requests.found.id</td>
<td>Sys_id of an existing artifact staging record. Data type: String</td>
</tr>
<tr>
<td>response.artifact_register_requests.update</td>
<td>Currently unused</td>
</tr>
<tr>
<td>response.artifact_register_requests.update.id</td>
<td>Currently unused</td>
</tr>
<tr>
<td>response.artifact_repository</td>
<td>Results for any artifact repository records that were manipulated during the processing of the request.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td><strong>Data type: Object</strong></td>
</tr>
<tr>
<td></td>
<td>&quot;artifact_repository&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;create&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;found&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;update&quot;: [Array]</td>
</tr>
<tr>
<td>response.artifact_repository.create</td>
<td>If artifact repositories were created by the request, the list of sys_ids of the created records.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type: Array</strong></td>
</tr>
<tr>
<td></td>
<td>&quot;create&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>response.artifact_repository.create.id</td>
<td>Sys_id of the artifact repository record created by the request.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type: String</strong></td>
</tr>
<tr>
<td>response.artifact_repository.found</td>
<td>If artifact repository records used in the register request already exist, the list of sys_ids of the existing records.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> No modifications are made to existing packages.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type: Array</strong></td>
</tr>
<tr>
<td></td>
<td>&quot;found&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>response.artifact_repository.found.id</td>
<td>Sys_id of an existing artifact repository record used by the request.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type: String</strong></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| response.artifact_repository.update | If the endpoint updates the track flag on a repository to true (from false), the list of sys_ids of the repository records that were modified.  
Data type: Array |
| | "update": [
| | | |
| | | | { |
| | | | | "id": "String"
| | | } |
| | ] |
| response.artifact_repository.update.id | Sys_id of the repository record whose track flag was modified.  
Data type: String |
| response.artifact_version | Results for any artifact version records that were manipulated during the processing of the request.  
Data type: Object |
| | "artifact_version": {
| | | "create": [Array],
| | | "found": [Array],
| | | "update": [Array]
| | } |
| response.artifact_version.create | Results for any artifact version records that were created during the processing of the request.  
Data type: Array |
| | "create": [
| | | |
| | | | { |
| | | | | "id": "String"
<p>| | | } |
| | ] |
| response.artifact_version.create.id | Sys_id of the artifact version record created by the request. |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>response.artifact_version.found</td>
<td>If artifact version records used in the register request already exist, the list of sys_ids of the existing records.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;found&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>response.artifact_version.found.id</td>
<td>Sys_id of the existing artifact version record used by the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>response.artifact_version.update</td>
<td>Currently unused</td>
</tr>
<tr>
<td>response.artifact_version.update.id</td>
<td>Currently unused</td>
</tr>
<tr>
<td>status</td>
<td>Status of the registration request. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Success: Response body contains return results.</td>
</tr>
<tr>
<td></td>
<td>• Error: Response body contains list of all errors.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

curl -X POST \\n'https://instance.servicenow.com/api/sn_devops/devops/artifact/registration?toolId=391b4ff6dba24010bc8cdd384b96198f&orchestrationToolId=99d70b36dba24010bc8cdd384b961985' \\
-H 'Accept: application/json' \\
-H 'Content-Type: application/json' \\
-u 'username':'password' \\
-d '{"artifacts": [\
  \\n  "name": "sa-web.jar",\n  \\n}
"version": "3.1",
"semanticVersion": "3.1.0",
"repositoryName": "services-1031"
},
{
"name": "sa-frontend.jar",
"version": "3.2",
"semanticVersion": "3.2.0",
"repositoryName": "services-1031"
}
"pipelineName": "spring-boot-app",
"taskExecutionNumber": "160",
"stageName": "Build",
"branchName": "master"
}}

{
"result": {
"status": "Success",
"response": {
"artifact_repository": {
"create": [
{
"id": "2e50ba8bdbaa4010bc8cdd384b961988"
}
],
"update": [],
"found": [
{
"id": "2e50ba8bdbaa4010bc8cdd384b961988"
}
]
},
"artifact": {
"create": [
{
"id": "6a50f603dbea4010bc8cdd384b96193d"
},
{
"id": "2e50f603dbea4010bc8cdd384b96193f"
}
],
"update": [],
"found": []
}
}
Example: Sample Python request

# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_devops/devops/artifact/registration?toolId=391b4ff6dba24010bc8cdd384b96198f&orchestrationToolId=99d70b36dba24010bc8cdd384b961985'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'
# Set proper headers
headers = {"Content-Type":"application/json","Accept":"application/json"}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers, data="{"artifacts": [
    {
        "name": "sa-web.jar",
        "version": "3.1",
        "semanticVersion": "3.1.0",
        "repositoryName": "services-1031"
    },
    {
        "name": "sa-frontend.jar",
        "version": "3.2",
        "semanticVersion": "3.2.0",
        "repositoryName": "services-1031"
    }
],
"pipelineName": "spring-boot-app",
"taskExecutionNumber": "160",
"stageName": "Build",
"branchName": "master"}
})

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
  "result": {
    "status": "Success",
    "response": {
      "artifact_repository": {
        "create": [
          {
            "id": "2e50ba8bdbaa4010bc8cdd384b961988"
          }
        ],
        "update": [],
        "found": [
          {
            "id": "2e50ba8bdbaa4010bc8cdd384b961988"
          }
        ]
      },
      "artifact": {
        "create": [
          {
            "id": "6a50f603dbea4010bc8cdd384b96193d"
          }
        ]
      }
    }
  }
}
DevOps - POST /devops/onboarding/app

Creates an onboarding app event that is asynchronously processed by the DevOps service.

This endpoint can perform the following actions:
- Create applications.
- Associate DevOps objects such as, plan, repository, and pipeline, to an application.
- Create a change step record and associate a change approval process to a pipeline.

All of these actions are tool and capability specific. The following table outlines the actions available for each supported tool.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Capabilities</th>
<th>App Association</th>
<th>Configure</th>
<th>Change Step creation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServiceNow Agile</td>
<td>Plans</td>
<td>✓</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Azure</td>
<td>Plans</td>
<td>✓</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td>Repos</td>
<td>✓</td>
<td>NA</td>
<td>✓</td>
</tr>
<tr>
<td>BitBucket</td>
<td>Repos</td>
<td>✓</td>
<td>✓</td>
<td>NA</td>
</tr>
<tr>
<td>GitHub</td>
<td>Repos</td>
<td>✓</td>
<td>✓</td>
<td>NA</td>
</tr>
<tr>
<td>GitLab</td>
<td>Repos</td>
<td>✓</td>
<td>✓</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td>Pipelines</td>
<td>✓</td>
<td>NA</td>
<td>✓</td>
</tr>
<tr>
<td>Jenkins</td>
<td>Pipelines</td>
<td>✓</td>
<td>NA</td>
<td>✓</td>
</tr>
<tr>
<td>Jira</td>
<td>Plans</td>
<td>✓</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

**URL format**

Versioned URL: `/api/sn_devops/{api_version}/devops/onboarding/app`

Default URL: `/api/sn_devops/devops/onboarding/app`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apps</td>
<td>Array of objects that define the apps to create within the associated ServiceNow instance.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>
|                    | "apps": [
|                    |   "bussinessApp": "String",
|                    |   "name": "String"
|                    | ]                                                                                                                                                 |
| apps.bussinessApp  | Name of the cmdb_ci_business_app to which map the application. This ties the DevOps application to the specified CMDB application.                                                                         |
|                    | Data type: String                                                                                                                                                                                           |
| apps.name          | Required. User friendly name of the application to create. An app is a DevOps product container object that ties plans, repositories, and pipelines together.                                                     |
|                    | Data type: String                                                                                                                                                                                           |
| credentials        | Required. Credentials to use for the webhook creation. The created webhook is used by the tool to send notifications.                                                                                      |
### Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>credentials</strong></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;credentials&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;password&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><strong>credentials.name</strong></td>
<td>Required. ServiceNow user ID of the user making the endpoint request. Located in the User [sys_user] table. This user must have the devops.integration.user role. Data type: String</td>
</tr>
<tr>
<td><strong>credentials.password</strong></td>
<td>Required. Password of the user specified in the credentials.name parameter. Data type: String</td>
</tr>
<tr>
<td><strong>pipelines</strong></td>
<td>Array of objects that define pipelines to associate with applications. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;pipelines&quot;: [{</td>
</tr>
<tr>
<td></td>
<td>&quot;app&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;tool&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><strong>pipelines.app</strong></td>
<td>Required if a pipelines object is specified. Name of the application to associate with the plan specified in the pipelines.name parameter.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>pipelines.changeSteps</strong></td>
<td>Array of objects that define the change steps to create for the pipeline specified in the <code>pipelines.name</code> field.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;changeSteps&quot;: [{</td>
</tr>
</tbody>
</table>
|                             |   "changeApprovalGroup": "String",
|                             |   "changeControlledBranches": "String",
|                             |   "changeType": "String",
|                             |   "name": "String",
|                             |   "type": "String"
<p>|                             | }]                                                                                                                                                                                                         |
|                             | Data type: String                                                                                                                                                                                             |
| <strong>pipelines.changeSteps.changeApprovalGroup</strong> | Required if a <code>changeSteps</code> object is specified. Name of the change approval group to associate with the pipeline. Located in the Group [sys_user_group] table. This group must already exist on the ServiceNow instance. |
|                             | Data type: String                                                                                                                                                                                             |
| <strong>pipelines.changeSteps.changeControlledBranches</strong> | Comma-separated list of code branches that are under change control.                                                                                                                                         |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request body parameters (continued)</td>
<td></td>
</tr>
<tr>
<td>pipelines.name</td>
<td>Required if a pipelines object is specified. Name of the pipeline to associate with the application specified in pipelines.app parameter. This pipeline must already</td>
</tr>
<tr>
<td>pipelines.changeSteps.name</td>
<td>Required if a changeSteps object is specified. Name of the step to create. The step name must match the name of the corresponding step in the orchestration tool's pipeline. Data type: String</td>
</tr>
<tr>
<td>pipelines.changeSteps.type</td>
<td>Required if a changeSteps object is specified. Type of step to create. The available step types are defined in the Type field of the Steps [sn_devops_step] table. Data type: String</td>
</tr>
<tr>
<td>pipelines.changeSteps.changeType</td>
<td>Required if a changeSteps object is specified. Type of change request to create. The available types are defined in the Type field of the Change Request [change_request] table. Data type: String</td>
</tr>
<tr>
<td>name</td>
<td>Required if a changeSteps object is specified. Name of the pipeline to associate with the application specified in pipelines.app parameter. This pipeline must already</td>
</tr>
<tr>
<td>description</td>
<td>This only applies for multi branch support. Data type: String Default: * (all branches)</td>
</tr>
</tbody>
</table>
## Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>be discovered on the ServiceNow instance.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> String</td>
</tr>
<tr>
<td>pipelines.tool</td>
<td>Required if a <code>pipelines</code> object is specified. Tool to associate with the pipeline to create a unique pipeline/tool identifier.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> String</td>
</tr>
<tr>
<td>plans</td>
<td>Array of objects that define a plan to associate with an application.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Array</td>
</tr>
<tr>
<td>&quot;plans&quot;: [</td>
<td><strong>Example:</strong></td>
</tr>
<tr>
<td>&quot;app&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;name&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;tool&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Required if a <code>plans</code> object is specified. Names of applications to associate with the plan specified in the <code>plans.name</code> parameter. These applications must have already been created or be specified in the <code>apps</code> object of this request.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Array of Strings</td>
</tr>
<tr>
<td>plans.apps</td>
<td>Required if a <code>plans</code> object is specified. Name of the plan to associate with the application specified in the <code>plans.app</code> parameter. This plan must already</td>
</tr>
<tr>
<td>plans.name</td>
<td></td>
</tr>
</tbody>
</table>
### Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>be discovered on the ServiceNow instance. Data type: String</td>
<td></td>
</tr>
<tr>
<td>plans.tool</td>
<td>Required if a plans object is specified. Tool to associate with the plan to create a unique plan/tool identifier. Data type: String</td>
</tr>
<tr>
<td>repositories</td>
<td>Array of objects that define the repositories associated with an application. If the repositories object contains an app parameter, then it maps the repository to the application. Data type: Array</td>
</tr>
<tr>
<td>repositories.app</td>
<td>Required if a repositories object is specified. Name of the application to associate with this repository. The application must have already been created or be specified in the apps object of this request. Data type: String</td>
</tr>
<tr>
<td>repositories.name</td>
<td>Required if a repositories object is specified.</td>
</tr>
</tbody>
</table>
### Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the repository to associate with the application specified in repositories.app parameter. This repository must already be discovered on the ServiceNow instance. Data type: String</td>
</tr>
<tr>
<td>repositories.tool</td>
<td>Required if a repositories object is specified. User friendly tool name to associate with the repository to create a unique repository/tool identifier. The tool must already be created on the ServiceNow instance. Data type: String</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>202</td>
<td>Onboarding request was successfully created.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Results of the onboarding request. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;code&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;details&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;onboardingRequestID&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>result.code</td>
<td>HTTP status code. Data type: String</td>
</tr>
<tr>
<td>result.details</td>
<td>Details about the onboarding request. Data type: String</td>
</tr>
<tr>
<td>result.errors</td>
<td>Description of any error that occurred. Data type: Object</td>
</tr>
<tr>
<td>result.error.message</td>
<td>Error message.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.onboardingRequestID</td>
<td>Unique identifier of the onboarding request. Use this value when calling the /devops/onboarding/status endpoint to obtain the status of the associated onboarding process.</td>
</tr>
<tr>
<td>result.status</td>
<td>Status of the onboarding request, either Success or Failed.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

This example creates the application Bike Rental Service and associates it with the Jenkins pipeline Bike Rentals, the GitHub repository nowDevops01/CorpSite, and the Azure plan testPlan_Bike_Rental.

```bash
curl -X POST "https://servicenow-instance/api/sn_devops/v1/devops/onboarding/app" \
-H "Accept: application/json" \
-H "Content-Type: application/json" \
-u "username":"password" \
-d "{
"apps": [ 
  { 
    "name": "Bike Rental Service"
  }
]
"pipelines": [ 
  {
    "tool": "Primary Jenkins",
    "name": "Bike Rentals",
    "app": "Bike Rental Service",
    "changeSteps": [ 
      {
        "name": "Prod",
        "type": "deploy",
        "changeApprovalGroup": "Change Management",
        "changeType": "normal"
      }
    ]
  }
],
```
"repositories": [
  {
    "tool": "Primary GitHub",
    "name": "nowDevops01/CorpSite",
    "app": "Bike Rental Service"
  }
]
"plans": [
  {
    "tool": "Primary Azure",
    "name": "testPlan_Bike_Rental",
    "apps": [
      "Bike Rental Service"
    ]
  }
]
"credentials": {
  "name": "devops_user",
  "password": "devops_password"
}
}

{
  "result": {
    "status": "Success",
    "details": "Onboarding request has been created. Please use the /status api and the onboarding request number to get the status of the request",
    "onboardingRequestID": "IBE0001045",
    "code": "202"
  }
}

DevOps - POST /devops/onboarding/tool

Creates an onboarding tool event that is asynchronously processed by the DevOps service.

Depending on the type of tool for which the request is posted, the endpoint performs the following actions:

- Creates the tool within the associated ServiceNow instance
- Connects to the tool
- Tries to configure the tool (only if the connection is successful)
- Discovers the tool (only if the connection is successful)
All of these actions are tool specific and are executed based on the integration capabilities for the specified tool.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Create</th>
<th>Connect</th>
<th>Discover</th>
<th>Configure</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServiceNow Agile</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>NA</td>
</tr>
<tr>
<td>Azure</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>BitBucket</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>GitHub</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>GitLab</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Jenkins</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Jira</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

The return results for this endpoint only indicate whether the request was posted, it does not indicate whether the associated tools were created. The actual tool creation is processed asynchronously. To obtain the status of the tool creation and all other actions, you must call the `/devops/onboarding/status` endpoint and pass in the `onboardingRequestID` parameter that is returned by this endpoint.

Before you are able to access this endpoint, your admin must have configured the DevOps connection and credential alias (CreateDevOpsTool) to setup tools in the instance. For details, see Setting up DevOps tools. In addition, the calling entity must have the devops.integration.user role.

**URL format**

Versioned URL: `/api/sn_devops/{api_version}/devops/onboarding/tool`

Default URL: `/api/sn_devops/devops/onboarding/tool`

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentials</td>
<td>Required. Credentials to use for the webhook creation. The tool uses the created webhook to send notifications. Data type: Object</td>
</tr>
<tr>
<td>credentials.name</td>
<td>Required. ServiceNow identifier of the user making the endpoint request. Located in the User [sys_user] table. This user must have the devops.integration.user role. Data type: String</td>
</tr>
<tr>
<td>credentials.password</td>
<td>Required. Password of the user specified in the credentials.name parameter. Data type: String</td>
</tr>
<tr>
<td>tools</td>
<td>Required. Array of objects that define the tools to create within the associated ServiceNow instance. You can create one or more tools in a single request. Data type: Array</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
|          | "username": "String",
|          | "url": "String"
|          | ]                                                                                                                                                                                                                                                                                                                                            |
| tools.name | Required. User friendly name or label of the tool to create. This name is used to identify the tool in the DevOps service.                                                                                                                                                                                                                     |
|          | Data type: String                                                                                                                                                                                                                                                                                                                         |
| tools.password | Required except for Agile tool. Password to use to access the tool. Supported authentications are basic and token; OAuth is not supported.                                                                                                                                                                                            |
|          | Data type: String                                                                                                                                                                                                                                                                                                                         |
| tools.type | Required. Type of tool to create, such as GitHub or BitBucket. The specified value must be one of the Tool labels in the Tool Integrations [sn_devops_tool_integration] table.                                                                                                                                                                       |
|          | Data type: String                                                                                                                                                                                                                                                                                                                         |
| tools.useMidServer | Flag that indicates whether the tool connects through a MID Server.                                                                                                                                                                                                             |
|          | Valid values:
|          | • true: Connects through a MID Server.
|          | • false: Does not connect through a MID Server.                                                                                                                                                                                                                                                                                           |
|          | Data type: Boolean                                                                                                                                                                                                                                                                                                                         |
| tools.username | Required except for Agile tool. User name to use to access the tool through this endpoint.                                                                                                                                                                                        |
|          | Data type: String                                                                                                                                                                                                                                                                                                                         |
| tools.url | Required except for Agile tool. URL of the tool to create. The DevOps service uses this URL to integrate with the tool.                                                                                                                                                               |
|          | Data type: String                                                                                                                                                                                                                                                                                                                         |
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>202</td>
<td>Onboarding request was successfully created.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Results of the onboarding request. Data type: Object</td>
</tr>
</tbody>
</table>

"result": {
  "code": "String",
}
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.code</td>
<td>HTTP status code. Data type: String</td>
</tr>
<tr>
<td>result.details</td>
<td>Detailed message about the onboarding request. Data type: String</td>
</tr>
<tr>
<td>result.errors</td>
<td>Description of any error that occurred. Data type: Object</td>
</tr>
<tr>
<td>result.error.message</td>
<td>Error message. Data type: String</td>
</tr>
<tr>
<td>result.onboardingRequestID</td>
<td>Unique identifier of the onboarding request. Use this value when calling the /devops/onboarding/status endpoint to obtain the status of the associated tool onboarding process. Data type: String</td>
</tr>
<tr>
<td>result.status</td>
<td>Status of the onboarding request, either Success or Failed. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

The following example shows how to onboard GitHub.

curl -X POST "https://servicenow-instance/api/sn_devops/v1/devops/onboarding/tool" \
-H "Accept: application/json" \
-H "Content-Type: application/json" \
-u "username":"password" \
-d "tool":{ "name": "Primary GitHub", "type": "GitHub", "url": "https://api.github.com", "username": "username", "password": "password", }
DevOps - POST /devops/orchestration/changeControl

Registers a callback with a ServiceNow instance for an orchestration task under change control, which can be called once a decision has been made on the change request.

Before you call this endpoint, you must have an orchestration task created in your ServiceNow instance. You can create this task either through Discovery or by calling the DevOps - POST /devops/orchestration/stepMapping endpoint. You must also ensure that the orchestration task is associated to a previously defined pipeline step and that change control is enabled (by selecting the Change control option on the App Stage form.) For details, see Accelerating DevOps change.

URL format

Versioned URL: /api/sn_devops/{api_version}/devops/orchestration/changeControl
Default URL: /api/sn_devops/devops/orchestration/changeControl
# Supported request parameters

## Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| api_version | Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.  
Data type: String |

## Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| toolId | Required. Sys_id of the DevOps tool for which to place the specified task under change control. Located in the Orchestration Tool [sn_devops_orchestration_tool] table.  
Data type: String |
| toolType | Type of tool. Valid value: Any value that identifies the tool, such as "jenkins".  
Data type: String |

## Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| callbackURL        | Required. URL that the ServiceNow instance can call back once the associated change request approval process is complete. This callback should let the pipeline execution continue or not, based on the approval result.  
Data type: String |
<p>| changeRequestDetails | Name-value pairs of the fields to set when closing a change request. This object can contain any fields in |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>the Change Request record</td>
<td>except the following:</td>
</tr>
<tr>
<td>• assignment_group</td>
<td></td>
</tr>
<tr>
<td>• business_service</td>
<td></td>
</tr>
<tr>
<td>• cmdb_ci</td>
<td></td>
</tr>
<tr>
<td>• impact</td>
<td></td>
</tr>
<tr>
<td>• number</td>
<td></td>
</tr>
<tr>
<td>• risk</td>
<td></td>
</tr>
<tr>
<td>• risk_impact_analysis</td>
<td></td>
</tr>
<tr>
<td>• standard_change_template</td>
<td></td>
</tr>
<tr>
<td>• sys_id</td>
<td></td>
</tr>
<tr>
<td>• template</td>
<td></td>
</tr>
<tr>
<td>• type</td>
<td></td>
</tr>
</tbody>
</table>

If these fields are included in the request, the endpoint returns a 400 error and the change request is not updated. Also, if you include fields that require other mandatory fields to be set, and those fields are not sent, the request will fail during later processing of the pipeline. This failure is noted in the work notes of the change request, along with an error message for that particular change request, such as:

Change request update failed with reason: <Error Message from platform>. Canceling change request and the related step execution.
Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>You can also pass the boolean parameter <strong>setCloseCode</strong> in this object to specify whether to set closure information on the associated Change Request record.</td>
</tr>
<tr>
<td></td>
<td>• If set to true, the close_code field is set to the status of the job and the close_notes field is set to a link to the step execution in the associated Change Request record. The default value is true.</td>
</tr>
<tr>
<td></td>
<td>• If set to false, the close_code and close_notes fields are not set and a link to the step execution is set in the work_notes field.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>isMultiBranch</td>
<td>If the orchestration tool is Jenkins, flag that indicates whether the project is of type Multibranch.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Project is of type Multibranch.</td>
</tr>
<tr>
<td></td>
<td>• false: Project is not of type Multibranch.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>orchestrationTaskDetails</td>
<td>Required. Description of the orchestration task.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>orchestrationTaskDetails.message</td>
<td>Free-form text message. Data type: String</td>
</tr>
<tr>
<td>orchestrationTaskDetails.triggerType</td>
<td>Required. The way the orchestration task was started. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• upstream: Upstream job triggered this job.</td>
</tr>
<tr>
<td></td>
<td>• user: User manually started the job.</td>
</tr>
<tr>
<td></td>
<td>• scm: Git/scm tool code commit triggered the job. Data type: String</td>
</tr>
<tr>
<td>orchestrationTaskDetails.upstreamTaskExecutionURL</td>
<td>Required if triggerType is set to upstream. Upstream task execution URL. For example: <a href="https://jenkins.mycompany.com:8080/job/Mobile-Platform-test/40/">https://jenkins.mycompany.com:8080/job/Mobile-Platform-test/40/</a> Data type: String</td>
</tr>
<tr>
<td>orchestrationTaskURL</td>
<td>Required. URL of the orchestration task for which to create the call back. For example: <a href="https://jenkins.mycompany.com:8080/orchestration_task/Mobile-Platform-deploy/">https://jenkins.mycompany.com:8080/orchestration_task/Mobile-Platform-deploy/</a></td>
</tr>
</tbody>
</table>
## Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parentNode</td>
<td>If the orchestration task is a child nested inside a parent orchestration task, this value contains the parent node details.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;parentNode&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;upstreamStageName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;upstreamTaskExecutionURL&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>parentNode.id</td>
<td>Unique ID for the parent node.</td>
</tr>
<tr>
<td></td>
<td>In Jenkins, this value corresponds to a pipeline stage ID.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>parentNode.name</td>
<td>Name of the parent node.</td>
</tr>
<tr>
<td></td>
<td>In Jenkins, this value corresponds to a pipeline stage name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>parentNode.upstreamStageName</td>
<td>Name of the parent node’s upstream orchestration task.</td>
</tr>
<tr>
<td></td>
<td>In Jenkins, this value corresponds to a pipeline stage name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>parentNode.upstreamTaskExecutionURL</td>
<td>URL of the parent node’s upstream task execution.</td>
</tr>
</tbody>
</table>
Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>In Jenkins, this value corresponds to a pipeline stage execution URL. Data type: String</td>
</tr>
<tr>
<td>parentStageName</td>
<td>If the orchestration task is a child nested inside a parent orchestration task, the name of the parent orchestration task. Data type: String</td>
</tr>
<tr>
<td>parentStageURL</td>
<td>If the orchestration task is a child nested inside a parent orchestration task, the URL of the parent orchestration task. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Invalid or missing authentication.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>changeControl</td>
<td>Flag that indicates whether the orchestration task is under change control. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Under change control.</td>
</tr>
<tr>
<td></td>
<td>• false: Not under change control</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
</tbody>
</table>

Example: Sample cURL request

```
curl -X POST "https://instance.servicenow.com/api/sn_devops/v1/devops/orchestration/changeControl?toolId=f0ca45679323008b52f3b457415ae6&toolType=jenkins" -H "accept: application/json" -H "Content-Type: application/json" -u "username":"password"
-d "{"orchestrationTaskURL":"https://jenkins.mycompany.com:8080/orchestration task/Mobile-Platform-deploy/", 
"callbackURL":"https://jenkins.mycompany.com:8080/sn-devops/964aca59-0cae-4d9b-a36d-5929f426cf31"}
```
"orchestrationTaskDetails":{
    "message":"Started by upstream project \"Mobile-Platform-test\" build number 40",
    "triggerType":"upstream",
    "upstreamTaskExecutionURL":"https://jenkins.mycompany.com:8080/job/Mobile-Platform-test/40/*"}
}*

Response:

```
{
  "result": {
    "changeControl": true,
    "status": "Success"
  }
}
```

Example: The following example shows how to pass fields in the changeRequestDetails object when closing a change request.

```
curl -X POST
  "https://instance.servicenow.com/api/sn_devops/v1/devops/orchestration/changeControl?toolId=f0ca4567932308b52f3b457415ae6&toolType=jenkins"
  -H "accept: application/json"
  -H "Content-Type: application/json"
  -u "username":"password" 
  -d "{
    "callbackURL":"http://192.168.0.4:3000/jenkins/sn-devops/pipeline_839b7605-b98d-4831-bc87-96829de7da37",
    "orchestrationTaskURL":"http://192.168.0.4:3000/jenkins/job/java_sample_tests#deploy",
    "isMultiBranch":"false",
    "orchestrationTaskName":"java_sample_tests#deploy",
    "orchestrationTaskDetails":{
      "triggerType":"upstream",
      "upstreamTaskExecutionURL":"http://192.168.0.4:3000/jenkins/job/java_sample_tests/129/execution/node/35/wfapi/describe",
      "taskExecutionURL":"http://192.168.0.4:3000/jenkins/job/java_sample_tests/129/execution/node/50/wfapi/describe"
    },
    "changeRequestDetails":{
      "setCloseCode":false,
```
DevOps - POST /devops/orchestration/stepMapping

Verifies that the information being passed is valid for the creation of an orchestration task and is valid for association to a previously created step. If both are valid, it creates the orchestration task and makes the association.
URL format

Versioned URL: /api/sn_devops/{api_version}/devops/orchestration/stepMapping
Default URL: /api/sn_devops/devops/orchestration/stepMapping

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>toolType</td>
<td>Required. Type of tool. Valid values: jenkins Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>branchName</td>
<td>Name of the branch associated with the orchestration task. Data type: String</td>
</tr>
<tr>
<td>isMultiBranch</td>
<td>If the orchestration tool is Jenkins, flag that indicates whether the project is of type Multibranch. Valid values:</td>
</tr>
</tbody>
</table>
## Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
|                       | • true: Project is of type Multibranch.  
|                       | • false: Project is not of type Multibranch.  
|                       | Data type: Boolean  
|                       | Default: false |
| parentNode             | If the orchestration task is a child nested inside a parent orchestration task, this value contains the parent node details.  
|                       | Data type: Object  
|                       | ```json  
|                       | "parentNode": {  
|                       |   "id": "String",  
|                       |   "name": "String",  
|                       |   "upstreamStageName": "String",  
|                       |   "upstreamTaskExecutionURL": "String"  
|                       | }  
| parentNode.id          | Unique ID for the parent node.  
|                       | In Jenkins, this value corresponds to a pipeline stage ID.  
|                       | Data type: String |
| parentNode.name        | Name of the parent node.  
|                       | In Jenkins, this value corresponds to a pipeline stage name.  
|                       | Data type: String |
| parentNode.upstreamStageName | Name of the parent node’s upstream orchestration task.  
|                       | In Jenkins, this value corresponds to a pipeline stage name.  
|                       | Data type: String |
| parentNode.upstreamTaskExecutionURL | URL of the parent node’s upstream task execution. |
### Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>In Jenkins, this value corresponds to a pipeline stage execution URL. Data type: String</td>
</tr>
<tr>
<td>parentStageName</td>
<td>If the orchestration task is a child nested inside a parent orchestration task, the name of the parent orchestration task. Data type: String</td>
</tr>
<tr>
<td>parentStageURL</td>
<td>If the orchestration task is a child nested inside a parent orchestration task, the URL of the parent orchestration task. Data type: String</td>
</tr>
<tr>
<td>orchestrationTaskName</td>
<td>Required. Name of the orchestration task. This is the same task referenced in the <code>orchestrationTaskURL</code>. Data type: String</td>
</tr>
<tr>
<td>orchestrationTaskURL</td>
<td>Required. URL of the orchestration task. For example: <a href="https://jenkins.mycompany.com:8080/orchestration_task/Mobile-Platform-deploy/">https://jenkins.mycompany.com:8080/orchestration_task/Mobile-Platform-deploy/</a> Data type: String</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>Flag that indicates whether the verification was successful. Data type: Boolean</td>
</tr>
<tr>
<td>stepValid</td>
<td>Flag that indicates whether the information sent is valid. Data type: Boolean</td>
</tr>
</tbody>
</table>
Example: Sample cURL request

```
curl -X POST
  "https://instance.servicenow.com/api/sn_devops/v1/devops/orchestration/stepMapping?toolId=
f0ca45679323008b52f3b457415ae6&toolType=jenkins"
-H "Accept: application/json"
-H "Content-Type: application/json"
-u "username":"password"
-d "{
  \"orchestrationTaskURL\":\" http://127.0.0.1:8090/jenkins/job/CorpSite/job/master#deploy
  PROD/\",
  \"orchestrationTaskName\":\" master#deploy PROD\"
}"
```

Results if information is valid.

```
{
  "result": {
    "stepValid": true,
    "status": "Success"
  }
}
```

Results if information is invalid.

```
{
  "result": {
    "stepValid": false,
    "status": "Success"
  }
}
```

DevOps - POST /devops/package/registration

Enables orchestration tools to create new artifact packages that contain specified artifact versions.

Artifact packages are used to display commits to change approvers when they view a change request. For commits to show in change requests, the package must be created in the stage prior to the stage containing the change approval.

Any artifact versions included in the package registration request should already be registered with DevOps. For details on registering artifact versions, see DevOps - POST /devops/artifact/registration.

Typically, you will pass in the `artifacts.version` and `artifacts.name` parameters when identifying an artifact version to include in a package. You can also
use the advanced features of this endpoint which allows you to pass the build details for each artifact version instead of the version itself. Artifact-specific build details include the following parameters:

- `artifacts.pipelineName` or `artifacts.projectName`
- `artifacts.taskExecutionNumber`
- `artifacts.stageName`
- `artifacts.branchName`

The endpoint uses these artifact-specific build details to look up task execution information and then copies the associated versions to the package. It uses non-artifact specific build details (`pipelineName` or `projectName`, `taskExecutionNumber`, `branchName`, and `stageName`) to associate the created package to a pipeline execution.

ℹ️ **Note**: This endpoint does not support updates. It always creates a new package even if one with same name already exists.

For additional information on artifact packages, see [Using DevOps change acceleration for releases](#).

**URL format**

Versioned URL: `/api/sn_devops/{api_version}/devops/package/registration`

Default URL: `/api/sn_devops/devops/package/registration`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
</tbody>
</table>

Data type: String

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>orchestrationToolId</td>
<td>Required. Sys_id of the orchestration tool associated with the artifact package. Located in the Orchestration Tool</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>toolId</td>
<td>Sys_id of the artifact tool associated with the artifact package. Located in the Artifact Tool [sn_devops_artifact_tool] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>artifacts</td>
<td>At least one entry required. Array of objects in which each object uniquely identifies an artifact version to include in the package.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>artifacts.branchName</td>
<td>Required if artifact.version is not passed and it is a multi-branch setup. Name of branch in which the artifact versions are stored.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>artifacts.name</td>
<td>Required. Artifact file name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
### Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>artifacts.pipelineName</td>
<td>Required if <code>artifact.version</code> and <code>artifacts.projectName</code> are not passed. Pipeline name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>artifacts.projectName</td>
<td>Required if <code>artifact.version</code> and <code>artifact.pipelineName</code> are not passed. Free-form project name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>artifacts.repositoryName</td>
<td>Required. Artifact repository name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160 characters</td>
</tr>
<tr>
<td>artifacts.stageName</td>
<td>Required if <code>artifact.version</code> is not passed. Stage name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 160</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>artifacts.taskExecutionNumber</td>
<td>Required if <code>artifact.version</code> is not passed. Pipeline run number.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>artifacts.version</td>
<td>Required if build details are not provided. Version of the artifacts.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Format: <code>&lt;Major&gt;..&lt;Minor&gt;</code></td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
</tbody>
</table>
### Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| branchName          | Required if it is multi-branch project. Name of the branch in the repository where the artifacts are stored. Data type: String  
Maximum length: 200 characters  
Default: Null     |
| name                | Required. Name of package. Data type: String  
Maximum length: 160 characters     |
| pipelineName        | Required if **projectName** is not passed. Pipeline name. Data type: String  
Maximum length: 160 characters  
Default: Null     |
| projectName         | Required if **pipelineName** is not passed. Free-form project name. Data type: String  
Maximum length: 160  
Default: Null     |
| stageName           | Required if **pipelineName** is passed. Stage name. Data type: String  
Maximum length: 160 characters  
Default: Null     |
| taskExecutionNumber | Required. Pipeline run number. Data type: String  
Maximum length: 128 characters  
Default: Null     |
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Success. Success with response body.</td>
</tr>
<tr>
<td>201</td>
<td>Created. Success with response body.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. The request URI does not match the APIs in the system, or the operation failed for unknown reasons. Invalid headers can also cause this error.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
## Response body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>response</td>
<td>Results of the package registration request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;response&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;artifact_register_requests&quot;: {},</td>
</tr>
<tr>
<td></td>
<td>&quot;sn_devops_m2m_artifact_version&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;sn_devops_package&quot;: {Object}</td>
</tr>
<tr>
<td>response.artifact_register_requests</td>
<td>Results for any register requests that have not completed processing and</td>
</tr>
<tr>
<td></td>
<td>been stored in the Artifact Staging Request table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;artifact_register_requests&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;create&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;found&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;update&quot;: [Array]</td>
</tr>
<tr>
<td>response.artifact_register_requests.create</td>
<td>If the request created package staging records, the list of sys_ids of the</td>
</tr>
<tr>
<td></td>
<td>staging records.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;create&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>response.artifact_register_requests.create.id</td>
<td>Sys_id of the package staging records created by the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>response.artifact_register_requests.found</td>
<td>If the associated package staging records already exist, the list of sys_ids of the staging records.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>Note:</td>
<td>No modifications are made to existing packages.</td>
</tr>
</tbody>
</table>

Data type: Array

```
"found": [
{
   "id": "String"
}
]
```

response.artifact_register_requests.found.id | Sys_id of an existing package staging record.  
Data type: String

response.artifact_register_requests.update | Currently unused.  
response.artifact_register_requests.update.id | Currently unused.  
response.sn_devops_m2m_artifact_version_package | Results for each artifact version associated with a package.  
Data type: Object

```
"sn_devops_m2m_artifact_version_package": {
   "create": [Array],
   "found": [Array],
   "update": [Array]
}
```

response.sn_devops_m2m_artifact_version_package.create | If the request created records linking artifact versions to packages, the list of sys_ids of the created records in the Artifact Version to Package [sn_devops_m2m_artifact_version_package] table.  
Data type: Array

```
"create": [
{
   "id": "String"
}
]
```

response.sn_devops_m2m_artifact_version_package.create.id | Sys_id of the Artifact Version to Package record created by the request.  

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>response.sn_devops_m2m_artifact_version_package.found</td>
<td>If a linking record already exists in the Artifacts Version to Package table, the list of sys_ids of the existing linking records.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> No modifications are made to existing records.</td>
</tr>
<tr>
<td>response.sn_devops_m2m_artifact_version_package.found.id</td>
<td>Sys_id of the existing Artifact Version to Package record.</td>
</tr>
<tr>
<td>response.sn_devops_m2m_artifact_version_package.update</td>
<td>Currently unused.</td>
</tr>
<tr>
<td>response.sn_devops_m2m_artifact_version_package.update.id</td>
<td>Currently unused.</td>
</tr>
<tr>
<td>response.sn_devops_package</td>
<td>Results for any package records in the Packages [sn_devops_package] table that were manipulated during the processing of the request.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Object</td>
</tr>
<tr>
<td></td>
<td>&quot;sn_devops_package&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;create&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;found&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;update&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Array</td>
</tr>
<tr>
<td></td>
<td>&quot;create&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>response.sn_devops_package.create.id</td>
<td>Sys_id of the package record created by the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>response.sn_devops_package.found</td>
<td>If package records used in the request already exist, the list of existing records.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;found&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>response.sn_devops_package.found.id</td>
<td>Sys_id of an existing package record referenced by the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>response.sn_devops_package.update</td>
<td>Currently unused.</td>
</tr>
<tr>
<td>response.sn_devops_package.update.id</td>
<td>Current unused.</td>
</tr>
<tr>
<td>status</td>
<td>Status of the registration request.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• Success: Response body contains results.</td>
</tr>
<tr>
<td></td>
<td>• Error: Response body contains errors.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: Sample cURL request

```bash
# Example passing the artifact version
curl -X POST
  "https://instance.servicenow.com/api/sn_devops/devops/package/registration?toolId=f0ca45679323008b52f3b457415ae6"
-H "accept: application/json"
-H "Content-Type: application/json"
-u "username":"password"
-d "{
  "name": "sentimentpackage",
  "artifacts": [
    {
      "name": "sa-web.jar",
      "repositoryName": "services-1031",
      "version": "3.1"
    },
    {
      "name": "sa-frontend.jar",
      "repositoryName": "services-1031",
      "version": "9.2"
    }
  ],
  "pipelineName": "spring-boot-app",
  "taskExecutionNumber": "160",
  "stageName": "Build",
  "branchName": "master"
}"
```

Response - package created:

```json
"result": {
  "status": "Success",
  "response": {
    "sn_devops_package": {
      "create": [
        {
          "id": "0d57367a532e00107109ddeeef7b122a"
        }
      ],
      "update": [],
      "found": []
    },
    "sn_devops_m2m_artifact_version_package": {
      "create": [
      
```
Example: Sample cURL request

# Example passing the build details

curl -X POST
  "https://instance.servicenow.com/api/sn_devops/devops/package/registration?toolId=f0ca45679323008b52f3b457415ae6"
  -H "accept: application/json"
  -H "Content-Type: application/json"
  -u "username":"password"
  -d "{"name": "sentimentpackage",
    "artifacts": ["sa-web.jar",
    {"name": "sa-frontend.jar",
    "repositoryName": "services-1031",
    "pipelineName": "spring-boot-fe",
    "taskExecutionNumber": "991",
    "stageName": "Build",
    "branchName": "master"}],
    "repositoryName": "services-1031",
    "pipelineName": "spring-boot-app",
    "taskExecutionNumber": "160",
    "stageName": "Build",
    "branchName": "master"}"
}
Response - request created a staged package:

```
{

  "result": {
    "status": "Success",
    "response": {
      "artifact_register_requests": {
        "create": [
          {
            "id": "4628f276532e00107109ddeeff7b1245"
          },
          {
            "id": "4e28f276532e00107109ddeeff7b1245"
          },
          {
            "id": "ce28f276532e00107109ddeeff7b1245"
          }
        ],
        "update": [],
        "found": []
      }
    }
  },
  "responseCode": 201
}
```

**DevOps - POST /devops/tool/{capability}**

Posts the passed in payload to the Inbound Event [sn_devops_inbound] table for processing by a custom DevOps subflow.

Call this endpoint from the associated webhook in your tool, specified by the passed in `capability` parameter. A custom subflow, to process the passed in payload, must already be defined in your ServiceNow instance. Based on a predefined schedule, the DevOps Master Flow picks up the webhook event information from the Inbound Event table and passes it to the custom subflow, which then process the payload. For more information on creating a custom subflow, see Creating DevOps subflows.
**URL format**

Versioned URL: `/api/sn_devops/{api_version}/devops/tool/{capability}`

Default URL: `/api/sn_devops/devops/tool/{capability}`

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>
| capability | Type of tool passing in the payload to store in the Inbound Event table. Valid values (not case-sensitive):  
  • artifact  
  • code  
  • orchestration  
  • plan  
  • test  
  Data type: String |

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>testType</td>
<td>Required if capability is set to test. The type of test being run. The valid values for this parameter are those in the Test type column of the Test Types [sn_devops_test_type] table.</td>
</tr>
<tr>
<td>toolId</td>
<td>Required. Sys_id of the tool for which the endpoint is being called.</td>
</tr>
</tbody>
</table>

### Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tool dependant</td>
<td>The tool determines the content of the payload.</td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Invalid or missing authentication.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>details</td>
<td>Entry for all error conditions.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Data type: Object</td>
<td>&quot;details&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>details.errors</td>
<td>Description of a single error.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td>&quot;errors&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>details.errors.message</td>
<td>Detailed error message.</td>
</tr>
<tr>
<td></td>
<td>• The request does not have any path parameters: No path parameters are</td>
</tr>
<tr>
<td></td>
<td>included in the call.</td>
</tr>
<tr>
<td></td>
<td>• The request does not have capability path parameter: The capability</td>
</tr>
<tr>
<td></td>
<td>path parameter was not included in the call.</td>
</tr>
<tr>
<td></td>
<td>• Unsupported capability type: The specified capability parameter is not a</td>
</tr>
<tr>
<td></td>
<td>valid value.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>status</td>
<td>Response status such as &quot;Success&quot; or &quot;Data Error&quot;.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the record created in the Inbound Event table.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

```
curl -X POST
  "https://instance.servicenow.com/api/sn_devops/v1/tool/artifact?toolId=3b59043ee4ea556e6850d61bc1a77e8a"
  -H "accept: application/json"
  -H "Content-Type: application/json"
  -u "username":"password"
  -d "{
```
"artifacts": [  
  
  {  
    "name": "sp-boot-app",  
    "version": "1.320",  
    "semanticVersion": "1.320.0",  
    "repositoryName": "sp-boot-app-repo"  
  },  
  
  "pipelineName": "spring-boot-app-gitea",  
  "taskExecutionNumber": "320",  
  "stageName": "Build"  
  
  }]  

Success response:

{
  "result": {  
    "status": "Success",  
    "sysId": "33c57245dbf68410bc8cdd384b961966"  
  }  
}

Example: Sample cURL request

curl -X POST  
  "https://instance.servicenow.com/api/sn_devops/v1/tool/artifact?toolId=3b59043ee4ea556e6850d61bc1a77e8a"  
  -H "accept: application/json"  
  -H "Content-Type: application/json"  
  -u "username":"password"  
  -d "{"  
    "timestamp": "2020-01-11T00:52:01.541+0000",  
    "nodeId": "137BEECF-E1BF7BC1-90212D1C-0749A6D4-BC670F6B","  
    "initiator": "admin/172.17.0.1",  
    "repositoryName": "maven-releases29",  
    "action": "CREATED",  
    "component": {  
      "id": "2a59043ed2ea556e6850d61bc1a77c7b","  
      "componentId": "c3AtYm9vdC1hcHAtcmVwbzoyYTU5MDQ2ZWQyZWE1NTZlNjg1MGQ2MWJjMWJiMWe3N2M3Yg",  
      "format": "maven2",  
      "name": "sentiment-analysis-web",  
      "group": "com.spboot",  
      "version": "1.1"  
    }  
"
Error response:

```
{
  "result": {
    "status": "Data Error",
    "details": {
      "errors": [
        {
          "message": "Unsupported capability type"
        }
      ]
    }
  }
}
```

DevOps - PUT /devops/orchestration/changeControl/{changeControlId}

Updates the change control information for the specified orchestration task.

**URL format**

**Versioned URL**: /api/sn_devops/{api_version}/devops/orchestration/changeControl/{changeControlId}

**Default URL**: /api/sn_devops/devops/orchestration/changeControl/{changeControlId}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>changeControlId</td>
<td>Sys_id of the change control task. Located in the Callback [sn_devops_callback] table. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>toolId</td>
<td>Required. Sys_id of the DevOps tool associated with the change control task. Located in the Orchestration Tool [sn_devops_orchestration_tool] table. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callbackURL</td>
<td>Required. URL that the ServiceNow instance can call back once the associated change request approval process is complete. This callback should let the pipeline execution continue or not, based on the approval result. Data type: String</td>
</tr>
</tbody>
</table>
| orchestrationTaskDetails | Required. Description of the orchestration task. Data type: Object  

  "orchestrationTaskDetails": {  
    "message": "String",  
    "triggerType": "String",  
    "upstreamTaskExecutionURL": "String"  
  }

<table>
<thead>
<tr>
<th>orchestrationTaskDetails.message</th>
<th>Free-form text message. Data type: String</th>
</tr>
</thead>
<tbody>
<tr>
<td>orchestrationTaskDetails.triggerType</td>
<td>Required. The way the orchestration task was started. Valid values:</td>
</tr>
</tbody>
</table>
### Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>orchestrationTaskDetails.upstreamTaskExecutionURL</td>
<td>Required if <code>triggerType</code> is set to <code>upstream</code>. Upstream task execution URL.</td>
</tr>
<tr>
<td></td>
<td>For example: <a href="https://jenkins.mycompany.com:8080/job/Mobile-Platform-test/40/">https://jenkins.mycompany.com:8080/job/Mobile-Platform-test/40/</a></td>
</tr>
<tr>
<td>orchestrationTaskURL</td>
<td>Required. URL of the orchestration task.</td>
</tr>
<tr>
<td></td>
<td>For example: <a href="https://jenkins.mycompany.com:8080/orchestration_task/Mobile-Platform-deploy/">https://jenkins.mycompany.com:8080/orchestration_task/Mobile-Platform-deploy/</a></td>
</tr>
<tr>
<td>toolType</td>
<td>Type of DevOps tool.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• jenkins</td>
</tr>
<tr>
<td>taskExecutionURL</td>
<td>URL of the task to execute.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

- **upstream**: Upstream job triggered this job.
- **user**: User manually started the job.
- **scm**: Git/scm tool code commit triggered the job.
Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Invalid or missing authentication.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found. The specified record could not be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
# Response body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>change_control</td>
<td>Flag that indicates whether the orchestration task is under change control. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Under change control.</td>
</tr>
<tr>
<td></td>
<td>• false: Not under change control.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>details</td>
<td>Entry for all error conditions.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;details&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [Array]</td>
</tr>
<tr>
<td>details.errors</td>
<td>Description of a single error.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;errors&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>details.errors.message</td>
<td>Detailed error message.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

```
curl -X PUT
   "https://instance.service-now.com/api/sn_devops/v1/devops/orchestration/change/f0ca45679323008b52f3b457415ae6?toolId=f0ca45679323008b52f3b457415ae6"
   -H "accept: application/json"
   -H "Content-Type: application/json"
   -u "username":"password"
   -d "{
     "orchestrationTaskURL":"https://jenkins.mycompany.com:8080/job/Mobile-Platform-deploy/
                      
     "toolType":"
```
Response:

```
{
   "result": {
      "change_control": true
   }
}
```

**Email API**

With the Email API you can receive and send email messages using REST.

**Security**

Users must have the email_api_send role to send email.

**Email - GET /now/email/{id}**

Returns the email details for the specified email record.

**URL format**

Versioned URL: /api/now/{api_version}/email/{id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Sys_id of the email for which to return details. Located in the Email [sys_email] table.</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_fields</td>
<td>Comma-separated list of fields to return in the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>403</td>
<td>Indicates the record is not found or the requesting user does not have access to the record. Verify the user has the proper role and access permissions.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bcc</td>
<td>List of the email addresses of the blind copied recipients of the email message. Maps to the blind_copied field. Data type: Array</td>
</tr>
<tr>
<td>cc</td>
<td>List of the email addresses of the copied recipients for the email message. Maps to the copied field. Data type: Array</td>
</tr>
<tr>
<td>headers</td>
<td>Name-value pairs of the headers associated with the message and their values. Data type: Object</td>
</tr>
<tr>
<td>html</td>
<td>HTML-enabled body of the email message. Maps to the body field. Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>Sys_id of the email record. Data type: String</td>
</tr>
<tr>
<td>importance</td>
<td>Importance of the email message. Maps to the importance field. Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>Processing state of the email message. Indicates whether the system scheduled jobs have processed the email message.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| Values include: | • error  
• ignored  
• processed  
• ready  |
| Data type: String | |
| subject | Subject of the email message. Maps to the subject field.  
Data type: String |
| text | Text-only body of the email message. Maps to the body_text field.  
Data type: String |
| to | List of the email addresses of the direct recipients of the email message. Maps to the recipients field.  
Data type: Array |
| type | Current state of the email message as incoming or outgoing mail.  
Values include:  
• received  
• received-ignored  
• send-failed  
• send-ignored  
• send-ready  
• sent  |
| Data type: String | |

**Example: Sample cURL request**

```bash
curl "http://instance.servicenow.com/api/now/email/06e095427f0022007f005212bdfa91b3" \
   --request GET \
   --header "Accept:application/json" \
   --user "user-name":"password"
```

```json
{
   "result" : {

```
Incident has been closed.

Summary details:
Closed by: System Administrator
Closed notes: Fixed

You can view all the details of the incident by following the link below:

Take me to the Incident

Thank you.

Ref: MSG0000006

Example: Sample Python request

# Need to install requests package for python
import requests

```python
# Set the request parameters
url = 'http://instance.service-now.com/api/now/email/06e095427f0022007f005212bdfa91b3'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept': 'application/json'}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "headers": {
            "X-ServiceNow-SysEmail-Version": "2.0",
            "X-ServiceNow-Source": "Notification-86f34b54c61122aa0108c1b7a33697cf"
        },
        "cc": [],
        "type": "send-ready",
        "html": "<html><head></head><body><p><font size="5" color="#808080" face="helvetica">Incident has been closed.</font></p><strong>Summary details</strong><p><font size="3" color="#808080" face="helvetica">Closed by: System Administrator</font></p><p><font size="3" color="#808080" face="helvetica">Closed notes: Fixed</font></p>You can view all the details of the incident by following the link below:<p><a href="incident.do?sys_id=e8e875b0c0a80164009dc852b4d677d5\&\&sysparm_stack=incident_list do?sysparm_query-active=true" style="background-color: #278efc;border: 1px" width="100%" height="100%"></a></p>
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Email - POST /now/email

Creates an email record using the passed information.

URL format

Versioned URL: /api/now/{api_version}/email

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bcc</td>
<td>List of the email addresses of the blind copied recipients of the email message. Maps to the blind_copied field.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> You can only specify up to 100 addresses in this field.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>cc</td>
<td>List of the email addresses of the copied recipients for the email message. Maps to the copied field.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> You can only specify up to 100 addresses in this field.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>headers</td>
<td>Name-value pairs of the headers associated with the message and their values.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>html</td>
<td>HTML-enabled body of the email message. Maps to the body field.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>importance</td>
<td>Importance of the email message. Maps to the importance field.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>subject</td>
<td>Subject of the email message. Maps to the subject field.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>table_name</td>
<td>Name of the table to save the email. Use this parameter to associate an email message to a particular related record elsewhere in the system.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This parameter also requires specifying the table_record_id parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>table_record_id</td>
<td>Target-related record to which the email applies. Use this parameter to associate an email message to a particular related record elsewhere in the system.</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong> This parameter also requires specifying the \textit{table_name} parameter.</td>
<td>Data type: String</td>
</tr>
<tr>
<td>\textbf{text}</td>
<td>Text-only body of the email message. Maps to the body_text field. Data type: String</td>
</tr>
<tr>
<td>\textbf{to}</td>
<td>Required. List of the email addresses of the direct recipients for the email message. Maps to the recipients field. \textit{Note:} You can only specify up to 100 addresses in this field. Data type: Array</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textbf{Accept}</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>\textbf{Content-Type}</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>403</td>
<td>Requesting user does not have access to the record. Verify that the user has the proper role and access permissions.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

**Response body parameters (JSON or XML)**

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
</table>
| href    | Link to the email record as an Email API GET request.  
Data type: String |
| id      | Sys_id of the Email record.  
Data type: String |
| links   | List of links to the Email record.  
Data type: Array |
| rel     | Type of link listed in the href parameter.  
Possible values:  
• self: The Email API GET request for the Email record.  
• status: The Email API GET request for the Email record showing only the id, type, state, and error fields.  
Data type: String |
Example: Sample cURL request

```bash
curl "http://instance.servicenow.com/api/now/email" \
--request POST \n--header "Accept:application/json" \n--header "Content-Type:application/json" \n--user 'username':'password'\n--data "{
  "to": [\
    "User1 <user1@example.com>",\
    "User2 <user2@example.com>"
  ],
  "cc": [\
    "User3 <user3@example.com>",\
    "User4 <user4@example.com>"
  ],
  "bcc": [\
    "User5 <user5@example.com>",\
    "User6 <user6@example.com>"
  ],
  "subject": "Hello There",
  "text": "Test Message",
  "html": "<b>Test Message</b>",
  "table_name": "incident",
  "table_record_id": "136b2140bd0312004d7d1371f1abdb6",
  "headers": {
    "X-Custom": "header"
  }
}
```

```
{
  "result": {
    "id": "b963219a44b02200964f63773cd6adfc",
    "links": [
      { "rel": "self", "href": "/now/v1/email/b963219a44b02200964f63773cd6adfc" },
      { "rel": "status", "href": "/now/v1/email/b963219a44b02200964f63773cd6adfc?sysparm_fields=id,type,state,error" }
    ]
  }
}
```
# Example: Sample Python request

```python
# Need to install requests package for python
# easy_install requests

import requests

# Set the request parameters
url = 'http://instance.servicenow.com/api/now/v1/email'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {"Content-Type": "application/json", "Accept": "application/json"}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, data="{
    "to": [
    "User1 <user1@example.com>",
    "User2 <user2@example.com>"
    ],
    "cc": [
    "User3 <user3@example.com>",
    "User4 <user4@example.com>"
    ],
    "bcc": [
    "User5 <user5@example.com>",
    "User6 <user6@example.com>"
    ],
    "subject": "Hello There",
    "text": "Test Message",
    "html": "<b>Test Message</b>",
    "table_name": "incident",
    "table_record_id": "136b2140bd0312004d7d1371f1abbdb6",
    "headers": {
    "X-Custom": "header"
    }
}
")

# Check for HTTP codes other than 200
```
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "id": "b963219a44b02200964f63773cd6adfc",
        "links": [
            {
                "rel": "self",
                "href": "/now/v1/email/b963219a44b02200964f63773cd6adfc"
            },
            {
                "rel": "status",
                "href": "/now/v1/email/b963219a44b02200964f63773cd6adfc?sysparm_fields=id,type,state,error"
            }
        ]
    }
}

External Content Ingestion API

The External Content Ingestion API provides endpoints that enable ingestion of content from sources outside of your ServiceNow® instance into the ServiceNow® AI Search application's index.

External Content Ingestion API – DELETE /ais/external_content/deleteByQuery/
{schema_table_name}

Deletes all external documents that match the specified query from the AI Search index.

URL format

Versioned URL: /api/now/{api_version}/ais/external_content/deleteByQuery/
{schema_table_name}?query={query}
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>schema_table_name</td>
<td>The name of the external content schema table that defines the schema for the indexed documents to delete. For example, u_ext_content.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>A valid URL-escaped query for the schema table specified by <code>schema_table_name</code>. For example, title=Introduction to query for documents whose title is Introduction.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Object describing the error encountered during processing of the request.</td>
</tr>
<tr>
<td>error.detail</td>
<td>Details of the error encountered during processing of the request.</td>
</tr>
<tr>
<td>error.message</td>
<td>Message for the error encountered during processing of the request.</td>
</tr>
<tr>
<td>result</td>
<td>Result from a successfully processed request.</td>
</tr>
</tbody>
</table>

Data type: Object

```json
"error": {
  "detail": "String",
  "message": "String"
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| status | Status of an unsuccessful request.  
Valid values:  
* failure  
Data type: String |

**Example: cURL request**
Delete documents with title field values *Introduction* and *Report for 31 October 2020* from the u_ext_content schema table.

curl  
"https://instance.service-now.com/api/now/v2/ais/external_content/deleteByQuery/u_ext_content?query=title%3DIntroduction%20OR%20title%3DReport%20for%2031%20October%202020"  
--request DELETE  
--user "username":"password"  
--header "Accept: application/json"

```json
{
  "result": "Delete By Query Successfully Executed"
}
```

**Example: Python request**
Delete documents with title field values *Introduction* and *Report for 31 October 2020* from the u_ext_content schema table.

```python
#Need to install requests package for python  
import requests  
import json

# Set the API endpoint URL for the request  
url =  
"https://instance.servicenow.com/api/now/v2/ais/external_content/deleteByQuery/u_ext_content"

# Set credentials  
user = "username"  
pwd = "password"

# Set query parameters  
parameters = {
```
"query": "title=Introduction OR title=Report for 31 October 2020"
}

# Set HTTP headers
headers = {
    "Accept": "application/json"
}

# Issue the HTTP request
response = requests.delete(url, auth=(user, pwd), params=parameters, headers=headers)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status:", response.status_code, "Headers:", response.headers, "Error Response:",
    response.content)
    exit()

# Display the JSON response
print(json.dumps(response.json()))

{
    "result": "Delete By Query Successfully Executed"
}

External Content Ingestion API – DELETE /ais/external_content/deleteDocument/
{schema_table_name}/{document_id}

Deletes the external document with a specified unique identifier from the AI
Search index.

**URL format**

**Versioned URL:** /api/now/{api_version}/ais/external_content/deleteDocument/
{schema_table_name}/{document_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>document_id</td>
<td>The unique identifier for the external document to delete. For example, ADMIN-2027858531-16. Data type: string</td>
</tr>
<tr>
<td>schema_table_name</td>
<td>The name of the external content schema table that defines the schema for the indexed document to delete. For example, u_ext_content. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Object describing the error encountered during processing of the request. Data type: Object</td>
</tr>
<tr>
<td>error.detail</td>
<td>Details of the error encountered during processing of the request. Data type: String</td>
</tr>
<tr>
<td>error.message</td>
<td>Message for the error encountered during processing of the request. Data type: String</td>
</tr>
<tr>
<td>result</td>
<td>Result from a successfully processed request. Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>Status of an unsuccessful request. Valid values: failure</td>
</tr>
</tbody>
</table>
**Example: cURL request**

Delete the document with identifier `ADMIN-2587918521-27` from the `u_ext_content` schema table.

```
curl
  'https://instance.service-now.com/api/now/v2/ais/external_content/deleteDocument/u_ext_content/ADMIN-2587918521-27' 
  --request DELETE 
  --user 'username':'password' 
  --header 'Accept: application/json'

{
  "result":"Document removed"
}
```

**Example: Python request**

Delete the document with identifier `ADMIN-2587918521-27` from the `u_ext_content` schema table.

```
# Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request
url =

# Set credentials
user = "username"
pwd = "password"

# Set HTTP headers
headers = {
  "Accept": "application/json",
}

# Issue the HTTP request
response = requests.delete(url, auth=(user, pwd), headers=headers)
```
# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status:", response.status_code, "Headers:", response.headers, "Error Response:", response.content)
exit()

# Display the JSON response
print(json.dumps(response.json()))

{
    "result":"Document removed"
}

External Content Ingestion API – POST /ais/external_content/ingestDocument/
{schema_table_name}

Sends a list of external documents to the AI Search ingestion batcher for indexing. After indexing completes, content from the ingested documents becomes searchable.

You can associate stored binary content with an external document by following these steps:

1. Store the binary content in AI Search using the POST /ais/external_content/
   storeContent endpoint. Record the value of the result response body parameter.

2. When sending the external document to AI Search using this endpoint, set its content_pointer request body parameter to match the recorded result response body parameter value.

During ingestion, AI Search parses the binary content and adds its searchable content to the indexed record that represents the external document. Parsing removes the stored content object.

URL format

Versioned URL: /api/now/{api_version}/ais/external_content/ingestDocument/
{schema_table_name}
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>schema_table_name</td>
<td>The name of the external content schema table that defines the schema for external documents in the request. For example, u_ext_content. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[array]</td>
<td>Required. Unnamed array of objects in which each object represents an external document to ingest for indexing. Data type: Array</td>
</tr>
<tr>
<td>array.content_pointer</td>
<td>Identifier for an instance of binary content stored using the POST /ais/external_content/storeContent endpoint. During ingestion, AI Search parses the binary content and adds</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>its searchable content to the indexed record that represents the external document. Parsing removes the stored content object.</td>
</tr>
<tr>
<td>![Note:]</td>
<td>This identifier should match the result response body element returned by the storeContent endpoint.</td>
</tr>
<tr>
<td>![Data type:]</td>
<td>String</td>
</tr>
<tr>
<td>[array].document_id</td>
<td>Required. Unique identifier for the external document in the external content schema table specified by the schema_table_name path parameter.</td>
</tr>
<tr>
<td>![Note:]</td>
<td>When you ingest a document, it overwrites any existing document in the same external content schema table that has the same document_id. If two or more documents in the same ingestion request have the same document_id, the request fails.</td>
</tr>
<tr>
<td>![Data type:]</td>
<td>String</td>
</tr>
<tr>
<td>[array].principals</td>
<td>Object containing key-value pairs that describe the external document’s access permissions for externally defined security principals (users and groups).</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image.png" alt="Image" /></td>
<td><img src="description.png" alt="Description" /></td>
</tr>
</tbody>
</table>

**Note:**

If you omit this parameter for a document, the request succeeds, but AI Search rejects the document with the ingestion feedback message *The principal of the document is missing.*

If this parameter object does not include any key-value pairs that grant access to a document, the request succeeds, but AI Search rejects the document with the ingestion feedback message *The principal of the document is invalid.*

**Data type:** Object

```
"principals": {
  "everyone": Boolean,
  "groups.deny": [Array],
  "groups.read": [Array],
  "none": Boolean,
  "users.deny": [Array],
  "users.read": [Array]
}
```

**Versions supported:** Available starting in v2 of the API.

**[array].principals.everyone**

Flag that indicates whether access to the external document is allowed for all users. When this parameter is set to **true**, all Now Platform users can view the indexed record created from the document.

**Note:** If you set both this parameter and **[array].principals.none** to **true** for a document, the request succeeds, but AI Search rejects the document with ingestion feedback message *The principal of the document is invalid.* You can only set one of these two parameters to **true** in a request.
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Valid values:</strong></td>
<td></td>
</tr>
<tr>
<td>• <strong>true</strong>: Allow access to the document for all users. AI Search ignores all <code>[array].principals.groups.*</code> and <code>[array].principals.users.*</code> parameter settings for the document.</td>
<td></td>
</tr>
<tr>
<td>• <strong>false</strong>: Do not allow all users to access the external document. Users can only access the document if <code>[array].principals.none</code> is set to false and if allowed by the interaction of their user mappings and the <code>[array].principals.groups.*</code> and <code>[array].principals.users.*</code> parameters.</td>
<td></td>
</tr>
<tr>
<td><strong>Data type</strong>: Boolean</td>
<td></td>
</tr>
<tr>
<td><strong>Default</strong>: <strong>true</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Versions supported</strong>: Available starting in v2 of the API.</td>
<td></td>
</tr>
<tr>
<td><code>[array].principals.groups.deny</code></td>
<td>Array of strings where each string is the name of an externally defined group that is denied access to the external document. Now Platform users mapped to any of these external groups cannot view the indexed search result record created from the document.</td>
</tr>
<tr>
<td></td>
<td>If either <code>[array].principals.everyone</code> or <code>[array].principals.none</code> is set to <code>true</code>, this parameter has no effect.</td>
</tr>
<tr>
<td></td>
<td>This parameter takes precedence over <code>[array].principals.groups.read</code>. If the same user is mapped to external groups with both read and deny access permissions for a document, AI Search denies that user access to the indexed record.</td>
</tr>
<tr>
<td></td>
<td>By default, <code>[array].principals.users.read</code> takes precedence over this parameter. To reverse this precedence order for an indexed source, see Change the precedence of user read and</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group deny permissions for an external content indexed source.</td>
<td></td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>Values can be in any format, depending on the names of the externally defined groups specified. Examples include:</td>
<td></td>
</tr>
<tr>
<td>&quot;groups.deny&quot;: [</td>
<td></td>
</tr>
<tr>
<td>&quot;hr-admin&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;legal&quot;</td>
<td></td>
</tr>
<tr>
<td>]</td>
<td></td>
</tr>
<tr>
<td>Versions supported: Available starting in v2 of the API.</td>
<td></td>
</tr>
<tr>
<td>[array].principals.groups.read</td>
<td></td>
</tr>
<tr>
<td>Array of strings where each string is the name of an externally defined group that is allowed to access the external document. Now Platform users mapped to any of these external groups can view the indexed search result record created from the document.</td>
<td></td>
</tr>
<tr>
<td>If either [array].principals.everyone or [array].principals.none is set to true, this parameter has no effect.</td>
<td></td>
</tr>
<tr>
<td>[array].principals.groups.deny takes precedence over this parameter. If the same user is mapped to external groups with both read and deny access permissions for a document, AI Search denies that user access to the indexed record.</td>
<td></td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>Values can be in any format, depending on the names of the externally defined groups specified. Examples include:</td>
<td></td>
</tr>
<tr>
<td>&quot;groups.read&quot;: [</td>
<td></td>
</tr>
<tr>
<td>&quot;devops&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;it&quot;</td>
<td></td>
</tr>
<tr>
<td>]</td>
<td></td>
</tr>
</tbody>
</table>
## Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;report-admins&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Versions supported: Available starting in v2 of the API.</td>
</tr>
<tr>
<td>[array].principals.none</td>
<td>Boolean option indicating whether access to the external document is denied for all users. When this parameter is set to true, only Now Platform users with the ais_high_security_admin elevated privilege role can view the indexed record created from the document.</td>
</tr>
<tr>
<td></td>
<td>Note: If you set both this parameter and [array].principals.everyone to true for a document, AI Search rejects the document during ingestion with error message The principal of the document is invalid. You can only set one of these two parameters to true in a request.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Deny access to the document for all users except those with the ais_high_security_admin elevated privilege role. AI Search ignores all [array].principals.groups.* and [array].principals.users.* parameter settings for the document.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not deny access to the document for all users. Users can access the document if [array].principals.everyone is set to true, or if allowed by the interaction of their user mappings and the [array].principals.groups.read and [array].principals.users.read parameters.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| [array].principals.users.deny | Array of strings where each string is the name of an externally defined user that is denied access to the external document. Now Platform users mapped to any of these external users cannot view the indexed search result record created from the document.  
If either [array].principals.everyone or [array].principals.none is set to true, this parameter has no effect.  
This parameter takes precedence over [array].principals.users.read. If the same user is mapped to external users with both read and deny access permissions for a document, AI Search denies that user access to the indexed record.  
Data type: Array  
Values can be in any format, depending on the names of the externally defined users specified. Examples include: |
| | "users.deny": [  
"ad\bow-ruggeri",  
"abel-tuter@sharepoint"  
]  
Versions supported: Available starting in v2 of the API. |

| [array].principals.users.read | Array of strings where each string is the name of an externally defined user that is allowed to access the external document. Now Platform users mapped to any of these external users can view the indexed search result record created from the document.  
If either [array].principals.everyone or [array].principals.none is set to true, this parameter has no effect. |
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[array].principals.users.deny</td>
<td>Takes precedence over this parameter. If the same user is mapped to external users with both read and deny access permissions for a document, AI Search denies that user access to the indexed record.</td>
</tr>
<tr>
<td></td>
<td>By default, this parameter takes precedence over [array].principals.groups.deny. To reverse this precedence order for an indexed source, see Change the precedence of user read and group deny permissions for an external content indexed source.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>Values can be in any format, depending on the names of the externally defined users specified. Examples include:</td>
</tr>
<tr>
<td></td>
<td>&quot;users.read&quot;: [</td>
</tr>
</tbody>
</table>
|                          |   "abel-tuter",
|                          |     "beth-anglin@sharepoint"                                                                  |                                                                                             |
|                          | ]                                                                                                                                                 |                                                                                             |
|                          | Versions supported: Available starting in v2 of the API.                                                                                          |                                                                                             |
| [array].properties       | Object containing name-value pairs where each pair represents a field name and value to ingest for the document. All field names and values must be specified as strings.                                            |
|                          | After ingestion, these document field values are accessible through the indexed source defined for the external content schema table specified by the schema_table_name path parameter. Users can search for these field values in search sources derived from this indexed source. |
|                          | Data type: Object                                                                                                                                  |                                                                                             |
|                          | Field names can only contain lowercase letters and underscores. Values can include                                                                 |                                                                                             |
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>any characters. Value length is limited by the <strong>max_length</strong> attribute defined for the field in the external content schema table. Examples of field name-value pairs include:</td>
<td></td>
</tr>
<tr>
<td>&quot;properties&quot;: {</td>
<td></td>
</tr>
<tr>
<td>&quot;creation_date&quot;: &quot;2020-11-03 12:27:43&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;file_size&quot;: &quot;10285&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;title&quot;: &quot;Introduction&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;url&quot;: &quot;file:///myhost/reports/Introduction.pdf&quot;</td>
<td></td>
</tr>
</tbody>
</table>
| }

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <strong>application/json.</strong></td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports <strong>application/json.</strong></td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
## Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>202</td>
<td>Partial Success. The request was processed. Some documents have ingestion feedback warning or error messages.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| error         | Object describing the error encountered during processing of the request.  

Data type: Object

```
"error": {
   "detail": "String",
   "message": "String"
}
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| error.detail  | Details of the error encountered during processing of the request.  

Data type: String

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| error.message | Message for the error encountered during processing of the request.  

Data type: String

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| result        | JSON-formatted string representing the result object for the ingestion request.  

Data type: String

```
"result": "{\"duration_in_ms\": Number,  
\"feedback\": [Array] }"
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.duration_in_ms</td>
<td>Time spent ingesting the external documents.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Unit: Milliseconds</td>
</tr>
<tr>
<td>result.feedback</td>
<td>Array of objects in which each object represents feedback for a document ingested from the request body.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;feedback&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;document_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;messages&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>result.feedback.document_id</td>
<td>Unique identifier for the external document as specified in the request body.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.feedback.messages</td>
<td>Array of unnamed objects in which each object represents an ingestion feedback message logged during indexing of the external document.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;messages&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;code&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;component&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;level&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>result.feedback.messages.code</td>
<td>Code for an exception thrown by the indexing workflow component that logged the ingestion feedback message.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.feedback.messages.component</td>
<td>Identifier for the indexing workflow component that logged the ingestion feedback message. Data type: String</td>
</tr>
<tr>
<td>result.feedback.messages.level</td>
<td>Logging level for the ingestion feedback message. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• ERROR</td>
</tr>
<tr>
<td></td>
<td>• INFO</td>
</tr>
<tr>
<td></td>
<td>• MINOR_ERROR</td>
</tr>
<tr>
<td></td>
<td>• WARN</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.feedback.messages.message</td>
<td>Text logged for the ingestion feedback message. Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>Status of an unsuccessful request. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• failure</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Precedence order for principal permissions**

The precedence order for `array.principals` permissions depends on the value of the `user_read_takes_precedence_over_group_deny` attribute for the indexed source used to ingest an external document.

<table>
<thead>
<tr>
<th>Attribute value</th>
<th>Precedence order for principal permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>From highest precedence to lowest:</td>
</tr>
<tr>
<td></td>
<td>1. <code>array.principals.everyone,</code></td>
</tr>
<tr>
<td></td>
<td>2. <code>array.principals.none</code></td>
</tr>
<tr>
<td></td>
<td>3. <code>array.principals.users.deny</code></td>
</tr>
<tr>
<td></td>
<td>4. <code>array.principals.users.read</code></td>
</tr>
</tbody>
</table>
### Attribute value

<table>
<thead>
<tr>
<th>Precedence order for principal permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. <code>[array].principals.groups.deny</code></td>
</tr>
<tr>
<td>5. <code>[array].principals.groups.read</code></td>
</tr>
</tbody>
</table>

**Note:** This is the default attribute value for external content indexed sources.

<table>
<thead>
<tr>
<th>false</th>
</tr>
</thead>
</table>

From highest precedence to lowest:

1. `[array].principals.everyone`, `[array].principals.none`  
2. `[array].principals.users.deny`, `[array].principals.groups.deny`  
3. `[array].principals.users.read`, `[array].principals.groups.read`  

**Note:** For instructions on setting this attribute value, see Change the precedence of user read and group deny permissions for an external content indexed source.

### Example: cURL request

Feed two external documents (with content pointers to binary content objects previously stored using the `POST /ais/external_content/storeContent` endpoint) for indexing into the `u_ext_content` schema table.

```bash
curl  
'https://instance.servicenow.com/api/now/v2/ais/external_content/ingestDocument/u_ext_content'  
--request POST  
--user 'username':'password'  
--header 'Accept: application/json'  
--header 'Content-Type: application/json'  
--data '  
{  
  "document_id": "ADMIN-2027858531-16",  
  "content_pointer": "749b52a1-baa8-4556-a4f3-00404c95e6a8",  
  "properties": {  
    "title": "Introduction",  
  
```
Example: Python request

Feed two external documents (with content pointers to binary content objects previously stored using the POST /ais/external_content/storeContent endpoint) for indexing into the u_ext_content schema table.

```python
# Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request
url = "https://instance.servicenow.com/api/now/v2/ais/external_content/ingestDocument/u_ext_content"

# Set credentials
user = "username"
pwd = "password"

# Set HTTP headers
headers = {
    "Accept": "application/json",
    "Content-Type": "application/json"
}

data = json.dumps(
    [
        {
            "document_id": "ADMIN-2027858531-16",
            "content_pointer": "749b52a1-baa8-4556-a4f3-00404c95e6a8",
            "properties": {
                "title": "Introduction",
                "url": "file:///myhost/reports/Introduction.pdf",
                "file_name": "Introduction.pdf",
                "file_size": "10285",
```
# Issue the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, data=data)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status:", response.status_code, "Headers:", response.headers, "Error Response:", response.content)
### External Content Ingestion API – POST ais/external_content/storeContent

Stores binary content as a content object in AI Search.

You can associate stored binary content with an external document by following these steps:

1. Store the binary content using this endpoint. Record the value of the `result` response body parameter.

2. Send the external document to AI Search using the **POST /ais/external_content/ingestDocument/** endpoint. Set the document's `content_pointer` request body parameter to match the recorded `result` response body parameter value.

During ingestion, AI Search parses the binary content and adds its searchable content to the indexed record that represents the external document. Parsing removes the stored content object.

### URL format

**Versioned URL:** /api/now/{api_version}/ais/external_content/storeContent
### Supported request parameters

#### Path parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

#### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

#### Request body parameters

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[binary data]</td>
<td>Required. Binary content to store as a content object in AI Search.</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• application/msword</td>
</tr>
<tr>
<td></td>
<td>• application/octet-stream</td>
</tr>
<tr>
<td></td>
<td>• application/pdf</td>
</tr>
<tr>
<td></td>
<td>• application/vnd.ms-excel</td>
</tr>
</tbody>
</table>
Request headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• application/vnd.ms-powerpoint</td>
<td></td>
</tr>
<tr>
<td>• application/vnd.ms-powerpoint.presentation.macroenabled.12</td>
<td></td>
</tr>
<tr>
<td>• application/vnd.openxmlformats-officedocument.presentationml.presentation</td>
<td></td>
</tr>
<tr>
<td>• application/vnd.openxmlformats-officedocument.presentationml.template</td>
<td></td>
</tr>
<tr>
<td>• application/vnd.openxmlformats-officedocument.spreadsheetml.sheet</td>
<td></td>
</tr>
<tr>
<td>• application/vnd.openxmlformats-officedocument.wordprocessingml.document</td>
<td></td>
</tr>
<tr>
<td>• application/vnd.openxmlformats-officedocument.wordprocessingml.template</td>
<td></td>
</tr>
<tr>
<td>• text/html</td>
<td></td>
</tr>
<tr>
<td>• text/plain</td>
<td></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Object describing the error encountered during processing of the request. Data type: Object.</td>
</tr>
</tbody>
</table>
|              | ```json
"error": {
"detail": "String",
"message": "String"
}
```                                                                                      |
| error.detail | Details of the error encountered during processing of the request. Data type: String.                                                      |
| error.message| Message for the error encountered during processing of the request. Data type: String.                                                       |
| result       | Identifier for the binary content object stored by successful request. Data type: String.                                                   |
|              | ```json
"result": "91841766-2a5f-4c64-a20a-27ca485eca21"
```                                                                                      |

**Note:** To attach the stored content to an ingested external document, specify this identifier as the `content_pointer` request body element for a request to the `ingestDocument` endpoint.

<table>
<thead>
<tr>
<th>status</th>
<th>Status of an unsuccessful request. Valid values:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• <strong>failure</strong></td>
</tr>
<tr>
<td></td>
<td>Data type: String.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

Store binary content for a PDF file as a content object.

```bash
curl 'https://instance.servicenow.com/api/now/v2/ais/external_content/storeContent' \   
--request POST 
```
--user 'username':'password' \
--header 'Content-Type: application/pdf' \
--data-binary '@Report-2020-08-31.pdf'

The response body includes the unique identifier for the new content object.

```
{
    "result" : "fb439a4f-62ad-4dab-9654-5088d99a6ff9"
}
```

**Example: Python request**

Store binary content for a PDF file as a content object.

```python
# Need to install requests package for python
import requests
import json

# Set the API endpoint URL for the request
url = "https://instance.servicenow.com/api/now/v2/ais/external_content/storeContent"

# Set credentials
user = "username"
pwd = "password"

# Set HTTP headers
headers = {
    "Content-Type": "application/pdf",
}

# Read binary data from file on local filesystem
with open('./Introduction.pdf', 'rb') as file:
    data = file.read()

# Issue the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, data=data)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print("Status:" , response.status_code, "Headers:" , response.headers, "Error Response:" , response.content)
    exit()

# Display the JSON response
print(json.dumps(response.json()))
```
The response body includes the unique identifier for the new content object.

```
{
    "result": "749b52a1-baa8-4556-a4f3-00404c95e6a8"
}
```

**HR REST API**

This API provides methods that return information about employees that work or live in the United States.

To access this API you must have the sn_hr_core.usa_employee_only role and the Human Resources Scoped App: Core (com_sn_hr_core) plugin activated.

**HR REST - GET /get_usa_employee_profile**

Returns employee profile information for employees based in the United States.

The returned information can be either:

- Employee profile information for a specified employee ID.
- Information for employees that have been added, or whose information has been updated, on or after a specified date.

ℹ️ **Note:** Employees must either have their work or home country specified as the United States for this endpoint to include them in the return results.

In addition, the endpoint sends the following parameters to the integrated CIC Plus application (see the Returns section for descriptions):

- EmployeeIdentifier
- FirstName
- MiddleName
- LastName
- Email
- WorkAddress
- WorkCity
- WorkState
- WorkPostalCode
- WorkCountry
- HomeAddress
- HomeCity
- HomeState
- HomePostalCode
- HomeCountry
- BirthDate
- HireDate

**URL format**

Default URL: `/api/sn_hr_core/hr_rest_api/get_usa_employee_profile`

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>employee_identifier</td>
<td>Unique identifier of an employee. The calling system generates this value and can be in whatever format is consistent with their system. To obtain this value, first call the endpoint using the <code>updated_after</code> parameter.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> This parameter is mutually exclusive with the <code>updated_after</code> parameter. This parameter defines a unique employee, whereas <code>updated_after</code> returns a group of employees created/updated after a specified date.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. For requests that exceed this number of records, use the <code>sysparm_offset</code> parameter to paginate record retrieval.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Default: 100,000</td>
</tr>
<tr>
<td>sysparm_offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval.</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_offset</td>
<td>This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks.</td>
</tr>
<tr>
<td></td>
<td>For example, the first time you call this endpoint, <code>sysparm_offset</code> is set to &quot;0&quot;. To simply page through all available records, use <code>sysparm_offset=sysparm_offset +sysparm_limit</code>, until you reach the end of all records.</td>
</tr>
</tbody>
</table>

| updated_after   | Date to use to determine the employee data to return. All United States-based employees (work or home) that were added to your instance, or whose information was updated on or after this date, are included in the return results. |
|                 | **Note:** This parameter is mutually exclusive with the `employee_identifier` parameter.                                                                                                                        |
|                 | Data type: String                                                                                                                                                                                            |
|                 | Format: YYYY-MM-DD HH:MM:SS or YYYY-MM-DD (time defaults to 00:00:00)                                                                                                                                       |

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next-Page</td>
<td>Flag that indicates whether there is additional data to obtain in a subsequent call. For example, if there are a total of 120 employee records that match the search criteria, and sysparm_limit is set to 100, then Next-Page is true. If no additional results are available, Next-Page is false. Valid values: • true: Additional data is available, another call must be made to obtain the next chunk of data. • false: No additional data.</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BirthDate</td>
<td>Employee's birth date. Data type: String</td>
</tr>
<tr>
<td>Company</td>
<td>Company name. Data type: String</td>
</tr>
<tr>
<td>Email</td>
<td>Employee's email address.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>EmployeeIdentifier</td>
<td>Unique employee identifier.</td>
</tr>
<tr>
<td>FirstName</td>
<td>Employee's first name.</td>
</tr>
<tr>
<td>HireDate</td>
<td>Date on which the employee was hired.</td>
</tr>
<tr>
<td>LastName</td>
<td>Employee's last name.</td>
</tr>
<tr>
<td>MiddleName</td>
<td>Employee's middle name.</td>
</tr>
<tr>
<td>ReHire</td>
<td>For future use.</td>
</tr>
<tr>
<td>ResidenceAddress</td>
<td>Employee's home street address.</td>
</tr>
<tr>
<td>ResidenceCity</td>
<td>Employee's home city.</td>
</tr>
<tr>
<td>ResidenceCountry</td>
<td>Employee's home country.</td>
</tr>
<tr>
<td>ResidencePostalCode</td>
<td>Employee's home zip code.</td>
</tr>
<tr>
<td>ResidenceState</td>
<td>Employee's home state.</td>
</tr>
<tr>
<td>Suffix</td>
<td>Employee's suffix, such as Mr., Mrs., or Dr.</td>
</tr>
<tr>
<td>WorkAddress</td>
<td>Employee's work street address.</td>
</tr>
<tr>
<td>WorkCity</td>
<td>Employee's work city</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>WorkCountry</td>
<td>Employee’s work country. Data type: String</td>
</tr>
<tr>
<td>WorkPostalCode</td>
<td>Employee’s work zip code. Data type: String</td>
</tr>
<tr>
<td>WorkState</td>
<td>Employee’s work state. Data type: String</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request for a specific employee’s information**

```bash
curl -X GET 
"https://instance.servicenow.com/api/sn_hr_core/V1/hr_rest_api/get_usa_employee_profile?employee_identifier=231"
-H "accept: application/json"
-u "username":"password"

{"result": [
{"EmployeeIdentifier": "231",
"FirstName": "Harold",
"MiddleName": "William",
"LastName": "Lewis",
"Suffix": "Mr.",
"Email": "Howard.Lewis@example.com",
"BirthDate": ",
"HireDate": "2021-02-18",
"Company": "ServiceNow Inc",
"ResidenceAddress": ",
"ResidenceCity": "SD",
"ResidenceState": "CA",
"ResidencePostalCode": ",
"ResidenceCountry": ",
"WorkAddress": "2617 South Robinson Avenue, Oklahoma City",
"WorkCity": "Oklahoma",
"WorkState": "OK",
"WorkPostalCode": "73109",
"WorkCountry": "USA",
"ReHire": ",
}]
```
Example: Sample cURL request for a list of employees added/updated on or after 05/18/2019

```bash
curl -X GET
  "https://instance.servicenow.com/api/sn_hr_core/v1/hr_rest_api/get_usa_employee_profile?update_after=2019-05-18"
-H "accept: application/json"
-u "username":"password"
```

```json
{"result": [
  {
    "EmployeeIdentifier": "2345",
    "FirstName": "Cheryl",
    "MiddleName": "Lynn",
    "LastName": "Smith",
    "Suffix": "Mrs.",
    "Email": "Cheryl.Smith@example.com",
    "BirthDate": "",
    "HireDate": "2019-06-18",
    "Company": "ServiceNow Inc",
    "ResidenceAddress": "",
    "ResidenceCity": "SD",
    "ResidenceState": "CA",
    "ResidencePostalCode": "",
    "ResidenceCountry": "",
    "WorkAddress": "2617 South Robinson Avenue, Oklahoma City",
    "WorkCity": "Oklahoma",
    "WorkState": "OK",
    "WorkPostalCode": "73109",
    "WorkCountry": "USA",
    "ReHire": ""
  },
  {
    "EmployeeIdentifier": "1111",
    "FirstName": "Thomas",
    "MiddleName": "",
    "LastName": "Doe",
    "Suffix": "Mr.",
    "Email": "Thomas.Doe@example.com",
    "BirthDate": "1978-05-22",
    "HireDate": "2019-08-14",
    "Company": "",
    "ResidenceAddress": "",
    "ResidenceCity": "San Jose",
    "ResidenceState": "CA",
    "ResidencePostalCode": "",
    "ResidenceCountry": ""
  }
]}
```
Example: Sample Python request

```python
# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_hr_core/V1/hr_rest_api/get_usa_employee_profile?employee_identifier=231'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {"Accept": "application/json"}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{"result": [
}]
```
Identification and Reconciliation API

The Identification and Reconciliation API uses the Identification and Reconciliation engine (IRE) to minimize creation of duplicate Configuration Items (CIs) and to reconcile CI attributes by only accepting information from authorized sources when updating the Configuration Management Database (CMDB).

Users must have the itil or asset role to use this API.

For more information on the IRE, see Identification and Reconciliation engine (IRE).

Identification and Reconciliation - POST /now/identifyreconcile

Inserts or updates configuration items (CIs) in the CMDB by applying identification and reconciliation rules to the specified payload (request body). Use this API instead of updating the CMDB directly.

Use the Identification and Reconciliation - POST /now/identifyreconcile/query endpoint to simulate submission of a payload to this endpoint without committing changes to the database.

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Use the Identification and Reconciliation - POST /now/identifyreconcile/enhanced endpoint instead if you need to specify Enhanced IRE options. For more on Enhanced IRE options, see Identification and Reconciliation engine (IRE).

**URL format**

Versioned URL: /api/now/{api_version}/identifyreconcile

Default URL: /api/now/identifyreconcile

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>api_version</td>
</tr>
<tr>
<td>Data type: String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>sysparm_data_source</td>
</tr>
<tr>
<td>Data type: String</td>
</tr>
<tr>
<td>Default: Insert the API payload into the incomplete payloads table.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Request body parameters (XML or JSON)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>items</td>
</tr>
<tr>
<td>Data type: Array</td>
</tr>
</tbody>
</table>
## Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.className</td>
<td>Required. The <code>sys_class_name</code> (class/table name) of the CI to create or update. This value can be any CMDB table, such as <code>cmdb_ci_linux_server</code> or <code>cmdb_ci_win_server</code>. Data type: String</td>
</tr>
<tr>
<td>items.internal_id</td>
<td>Unique identifier for this item in this payload. Can be any value, but must be unique within the payload. Data type: String</td>
</tr>
<tr>
<td>items.lookup</td>
<td>Array of objects in which each object describes a lookup-based identification record. These records are used to identify the top-level configuration item based on a lookup table that has a reference back to the Configuration Item [cmdb_ci] table. Data type: Array</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>items.lookup.className</td>
<td>Required. The sys_class_name or class/table name of the CI to create or update. This value can be any CMDB class/table, such as cmdb_serial_number or cmdb_ci_network_adapter or a non-CMDB hierarchy class. Data type: String</td>
</tr>
<tr>
<td>items.lookup.internal_id</td>
<td>Unique identifier for this lookup record in this payload. Can be any value, but must be unique within the payload. Data type: String</td>
</tr>
<tr>
<td>items.lookup.sys_object_source_info</td>
<td>Object describing a unique CI identifier for a specified data source. Different sources may have different name-value pairs for the same CI. Data type: Object</td>
</tr>
</tbody>
</table>

```
"lookup": [  
  {  
    "className": "String",
    "internal_id": "String",
    "sys_object_source_info": {Object},
    "values": {Object}
  }  
]
```
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.lookup.sys_object_source_info.source_feed</td>
<td>Name that uniquely identifies the feed sending this CI. Use if the source can have multiple feeds. Data type: String</td>
</tr>
<tr>
<td>items.lookup.sys_object_source_info.source_name</td>
<td>Data source for the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>items.lookup.sys_object_source_info.source_native_key</td>
<td>Unique key/id from the source for the CI. Can be any string that is unique to the item. Data type: String</td>
</tr>
<tr>
<td>items.lookup.sys_object_source_info.source_recency_timestamp</td>
<td>UTC date and time that CI was scanned. Data type: String Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>items.lookup.values</td>
<td>Object describing fields this lookup item as name/value pairs, where the name is the field name. Data type: Object</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>items.related</strong></td>
<td>Array of objects in which each object describes an item record to create or update, based on a related table that has a reference to the top-level CI that is being identified. Rules in the Related Entry [cmdb_related_entry] table define what type of record can be in this array. The related table may or may not extend the Configuration Item [cmdb_ci] table. These records are not used to identify the configuration item. Data type: Array</td>
</tr>
<tr>
<td><strong>values</strong>: {</td>
<td></td>
</tr>
<tr>
<td>&quot;ip_address&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;mac_address&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;serial_number&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;serial_number_type&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;valid&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>items.related.className</strong></td>
<td>Required. The sys_class_name or class/table name of the related item to create or update. Data type: String</td>
</tr>
<tr>
<td><strong>items.related.internal_id</strong></td>
<td>Unique identifier for this related item in this payload. Can be any value, but must be unique within the payload. Data type: String</td>
</tr>
<tr>
<td><strong>items.related.sys_object_source_info</strong></td>
<td>Object describing a unique CI identifier for a specific data source. Different sources may have different name-value pairs for the same CI. Data type: Object</td>
</tr>
</tbody>
</table>
|                                           | "sys_object_source_info": {
|                                           |   "source_feed": "String",
|                                           |   "source_name": "String",
|                                           |   "source_native_key": "String",
|                                           |   "source_recency_timestamp": "String"
<p>|                                           | }                                                                                                                                                                                                          |
| <strong>items.related.sys_object_source_info.source_feed</strong> | Name that uniquely identifies the feed sending this related item. Use if the source can have multiple feeds. Data type: String                                                                                     |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.related.sys_object_source_info.source_name</td>
<td>Data source for the related item information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>items.related.sys_object_source_info.source_native_key</td>
<td>Unique key/id from the source for the related item. Can be any string that is unique to the item. Data type: String</td>
</tr>
<tr>
<td>items.related.sys_object_source_info.source_recency_timestamp</td>
<td>UTC date and time that the related item was scanned. Data type: String Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>items.related.values</td>
<td>Object describing fields to create or update for this related item as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced sys_id. Data type: Object</td>
</tr>
</tbody>
</table>

Field names and types depend on the fields selected by the user, such as:

```json
"values": {
  "host_name": "String",
  "ip_address": "String",
  "name": "String",
...}
```
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.settings</td>
<td>Object containing parameters defining the types of updates that are permitted. Data type: Object</td>
</tr>
<tr>
<td>items.settings.skipReclassificationRestrictionRules</td>
<td>Flag that indicates whether IRE should skip running the Reclassification Restriction rule that matches the class for the payload item. Valid values:</td>
</tr>
</tbody>
</table>
|                                              |  • true: Skip running the rule.  
|                                              |  • false: Run the rule.  
|                                              | Default: false                                                                                                                              |
| items.settings.updateWithoutDowngrade        | Indicates whether update and downgrade are both permitted for this CI. Valid values:                                                        |
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
|      | • true: Update the item but downgrade is not permitted.  
     | • true: Update the item but downgrade is not permitted.  
     | • false: Both item update and downgrade are permitted. |
|      | Data type: Boolean  
     | Default: false |
|      | Indicators whether this CI can be updated and its class switched.  
     | Valid values:  
     | • true: Update the item but class switching is not permitted.  
     | • false: Both item update and class switching are permitted. |
|      | Data type: Boolean  
     | Default: false |
| items.settings.updateWithoutSwitch | Indicates whether update and upgrade are both permitted for this CI.  
     | Valid values:  
     | • true: Update the item upgrade is not permitted.  
     | • false: Both item update and upgrade are permitted. |
|      | Data type: Boolean  
     | Default: false |
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.sys_object_source_info</td>
<td>Object describing a unique CI identifier. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_object_source_info&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;source_feed&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;source_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;source_native_key&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;source_recency_timestamp&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>items.sys_object_source_info.source_feed</td>
<td>Name that uniquely identifies the feed sending this CI. Use if the source</td>
</tr>
<tr>
<td></td>
<td>can have multiple feeds. Data type: String</td>
</tr>
<tr>
<td>items.sys_object_source_info.source_name</td>
<td>Data source for the CI information. This value must be one of the choice</td>
</tr>
<tr>
<td></td>
<td>values defined for the discovery_source field of the Configuration Item</td>
</tr>
<tr>
<td></td>
<td>[cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>items.sys_object_source_info.source_native_key</td>
<td>Unique key/id from the source for the CI. Can be any string that is unique</td>
</tr>
<tr>
<td></td>
<td>the item. Data type: String</td>
</tr>
<tr>
<td>items.sys_object_source_info.source_recency_timestamp</td>
<td>UTC date and time that CI was scanned. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.values</td>
<td>Object describing fields to create or update for this CI as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced sys_id.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Object</td>
</tr>
<tr>
<td></td>
<td>Field names and types depend on the fields selected by the user, such as:</td>
</tr>
<tr>
<td></td>
<td>```json</td>
</tr>
<tr>
<td></td>
<td>&quot;values&quot;: {</td>
</tr>
</tbody>
</table>
|                              |             "host_name": "String",
|                              |             "ip_address": "String",
|                              |             "name": "String",
|                              |             "os_name": "String",
|                              |             "sys_class_name": "String"   |
| referenceItems                | Array of objects in which each object describes a reference between two items in the payload.                                                                                                               |
|                              | **Data type:** Array                                                                                                                                                                                       |
|                              | ```json |
|                              |     "referenceItems": [ |
|                              |         { |
|                              |             "referenced": "String",
|                              |             "referencedBy": "String",
<p>|                              |             &quot;referenceField&quot;: &quot;String&quot; |
|                              |     } |
|                              | ```json |
| referenceItems.referenced     | The <strong>internal_id</strong> defined for the item being referenced by another item.                                                                                                                                |
|                              | <strong>Data type:</strong> String                                                                                                                                                                                    |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>referenceItems.referencedBy</td>
<td>The <code>internal_id</code> defined for the item that references another item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>referenceItems.referenceField</td>
<td>Name of the reference field in the class/table for the <code>referencedBy</code> item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>relations</td>
<td>Array of objects in which each object describes a relationship between two items defined in the payload.</td>
</tr>
<tr>
<td></td>
<td>An object in this array can use either of two formats:</td>
</tr>
<tr>
<td></td>
<td>• The object can define a relationship between two top-level items (only) using <code>parent</code> and <code>child</code> name-value pairs, with values representing item indexes from the payload <code>items</code> array.</td>
</tr>
<tr>
<td></td>
<td>• The object can define a relationship between any two items, including top-level, related, or lookup items, using <code>parent_id</code> and <code>child_id</code> key/value pairs, with values representing <code>internal_id</code> values defined for those items.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>

"relations": [ } }
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;child&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;parent&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_rel_source_info&quot;: (Object),</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td>&quot;relations&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;child_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;parent_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_rel_source_info&quot;: (Object),</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>relations.child</td>
<td>The integer index of the object in the items array that represents the child in the relationship.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>relations.child_id</td>
<td>The internal_id of the child item in the relationship.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>relations.parent</td>
<td>The integer index of the object in the items array that represents the parent in the relationship.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>relations.parent_id</td>
<td>The internal_id of the parent item in the relationship.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>relations.sys_rel_source_info</td>
<td>Discovery source information for the</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>relationship. For non-dependency relationships, this information is in the Relationship Sources [sys_rel_source] table. Data type: Object</td>
</tr>
<tr>
<td>relations.sys_rel_source_info.source_name</td>
<td>Discovery source name. Default: Discovery source passed in the API method parameter.</td>
</tr>
<tr>
<td>relations.sys_rel_source_info.source_feed</td>
<td>Any string that is a sub-discovery/scan within the discovery source. Default: ‘UNKNOWN’ is stored in the source_feed column when creating a record in sys_rel_source table.</td>
</tr>
<tr>
<td>relations.type</td>
<td>The type of relationship existing between the parent and child items. This must be a name field value from the CI Relationship Type [cmdb_rel_type] table. Data type: String</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Array of results produced by the query.</td>
</tr>
<tr>
<td></td>
<td>Data type</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: [</td>
</tr>
<tr>
<td></td>
<td>&quot;additionalCommittedItems&quot;: [</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>result.additionalCommittedItems</td>
<td>Array of objects in which each object describes a CI not included in the request body to insert or update. Data type: Array</td>
</tr>
<tr>
<td>result.additionalCommittedItems.className</td>
<td>The sys_class_name of this additional CI. Data type: String</td>
</tr>
<tr>
<td>result.additionalCommittedItems.errorCount</td>
<td>The number of errors encountered while processing this additional CI. Data type: Number</td>
</tr>
<tr>
<td>result.additionalCommittedItems.errors</td>
<td>Array of objects in which each object describes an error encountered while processing this additional CI. Data type: Array</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.additionalCommittedItems.errors.error</td>
<td>The type of error encountered while processing this additional CI. Data type: String</td>
</tr>
<tr>
<td>result.additionalCommittedItems.errors.message</td>
<td>The error message encountered while processing this additional CI. Data type: String</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identificationAttempts</td>
<td>Array of objects describing the attempts made to identify this additional CI. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;identificationAttempts&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;attemptResult&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;attributes&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;hybridEntryCiAttributes&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;identifierName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;searchOnTable&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identificationAttempts.attemptResult</td>
<td>The outcome of this additional CI identification attempt. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• MATCHED: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• MULTI_MATCH: Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• NO_MATCH: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identificationAttempts.attributes</td>
<td>Array of CI identifier entry attributes used during this additional CI identification attempt. Data type: Array. Attribute names and types depend on the request body data and the identifier in use, such as:</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identifierEntrySysId</td>
<td>The sys_id to identify this additional CI.</td>
</tr>
<tr>
<td></td>
<td>Notable values:</td>
</tr>
<tr>
<td></td>
<td>• Unknown: Identification of this additional CI failed. See errors for details.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.inputIndices</td>
<td>Array of index values for CIs from the request body items array that correspond to this additional CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.additionalCommittedItems.markers</td>
<td>Array of marker values for internal use.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.additionalCommittedItems.mergedPayloads</td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this additional CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.additionalCommittedItems.operation</td>
<td>The operation performed for this additional CI.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.additionalCommittedItems.sysId</td>
<td>The sys_id found for this additional CI through identification. Notable values: Unknown - Identification of this additional CI failed. See errors for details.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.warnings</td>
<td>Array of objects in which each object describes a warning encountered while processing this additional CI.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>result.additionalCommittedItems.warnings.error</code></td>
<td>The type of warning encountered while processing this additional CI. Only supported value: cmdb_rel_ci. Data type: String</td>
</tr>
<tr>
<td><code>result.additionalCommittedItems.warnings.message</code></td>
<td>The warning message encountered while processing this additional CI. Only supported value: cmdb_rel_ci. Data type: String</td>
</tr>
<tr>
<td><code>result.additionalCommittedRelations</code></td>
<td>Array of objects describing dependent relationship CIs not included in the request body list to insert or update. Data type: Array</td>
</tr>
<tr>
<td><code>result.additionalCommittedRelations.className</code></td>
<td>The sys_class_name of this additional dependent relationship CI. Only supported value: cmdb_rel_ci. Data type: String</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.additionalCommittedRelations.errorCount</td>
<td>The number of errors encountered while processing this additional dependent relationship CI. Data type: Number</td>
</tr>
<tr>
<td>result.additionalCommittedRelations.errors</td>
<td>Array of objects in which each object describes an error encountered while processing this additional dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td>result.additionalCommittedRelations.errors.error</td>
<td>The type of error encountered while processing this additional dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td>result.additionalCommittedRelations.errors.message</td>
<td>The error message encountered while processing this additional dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td>result.additionalCommittedRelations.inputIndices</td>
<td>Array of index values for dependent relationship CI objects in the request body relations array that correspond to this additional dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td>result.additionalCommittedRelations.markers</td>
<td>Array of marker values for internal use.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.additionalCommittedRelations.mergedPayloadIds</td>
<td>Array of system partial payloads that were merged during processing of this additional dependent relationship CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;mergedPayloadIds&quot;:</td>
</tr>
<tr>
<td></td>
<td>&quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.additionalCommittedRelations.operation</td>
<td>The operation performed for this additional dependent relationship CI. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• INSERT: The dependent relationship CI is inserted into the target table.</td>
</tr>
<tr>
<td></td>
<td>• INSERT_AS_INCOMPLETE: The dependent relationship CI had errors and is inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.</td>
</tr>
<tr>
<td></td>
<td>• INSERT_AS_PARTIAL: The dependent relationship CI had errors and is inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>result.items</td>
<td>Array of objects in which each object describes</td>
</tr>
<tr>
<td></td>
<td>a CI included in the request body</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems</td>
<td>Array of objects in which each object describes</td>
</tr>
<tr>
<td></td>
<td>a related CI (table lookup CI) from the CMDB</td>
</tr>
<tr>
<td></td>
<td>IRE Partial Payloads [cmdb_ire_partial_payloads]</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.className</td>
<td>The sys_class_name of the related item. Data type: String</td>
</tr>
</tbody>
</table>
| result.items.additionalRelatedItems.inputIndices         | Array of index values for CIs from the request body items array that correspond to this related item. Data type: Array |}
<p>| result.items.additionalRelatedItems.markers             | Array of marker values for internal use. Data type: Array                    |
| result.items.additionalRelatedItems.mergedPayloadIds     | Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this related item. Data type: Array |
| result.items.additionalRelatedItems.sysId                | The sys_id of this related CI. Data type: String                             |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.className</td>
<td>The sys_class_name of this CI. Data type: String</td>
</tr>
<tr>
<td>result.items.duplicateLookupIndices</td>
<td>Object containing arrays of index values for items.lookup elements that reference the same related CI. The array name indicates the first element from the request body items.lookup array that references a given related CI. Its values are the set of additional elements from the same array that reference the same related CI. Data type: Object</td>
</tr>
<tr>
<td>result.items.errorCount</td>
<td>The number of errors encountered while processing this CI. Data type: Number</td>
</tr>
<tr>
<td>result.items.errors</td>
<td>Array of objects in which each object describes an error encountered while processing this CI. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"results": 

"duplicateLookupIndices": 

"2": [Array]"
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.errors.error</td>
<td>The type of error encountered while processing this CI. Data type: String</td>
</tr>
<tr>
<td>result.items.errors.message</td>
<td>The error message encountered while processing this CI. Data type: String</td>
</tr>
<tr>
<td>result.items.identificationAttempts</td>
<td>Array of objects. Each object describes an attempt made to identify this CI. Data type: Array</td>
</tr>
<tr>
<td>result.items.identificationAttempts.attemptResult</td>
<td>The outcome of this CI identification attempt. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• MATCHED: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• MULTI_MATCH: Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• NO_MATCH: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>result.items.identificationAttempts.attributes</td>
<td>Array of CI identifier entry attributes used during this CI identification attempt. Attribute names and types depend on the request body data and the identifier in use, such as:</td>
</tr>
<tr>
<td>result.items.identificationAttempts.hybridEntryCiAttributes</td>
<td>Array of CI identifier entry attributes used during this CI identification attempt. Attribute names and types depend on the request body data and the identifier in use, such as:</td>
</tr>
<tr>
<td>result.items.identificationAttempts.identifierName</td>
<td>The identifier rule used for this CI identification attempt.</td>
</tr>
<tr>
<td>result.items.identificationAttempts.searchOnTable</td>
<td>The name of the table searched for this CI identification attempt.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.items.identifierEntrySysId</td>
<td>The sys_id for the identifier rule used to identify this CI.</td>
</tr>
<tr>
<td></td>
<td>Notable values:</td>
</tr>
<tr>
<td></td>
<td>• Unknown: Identification of this CI failed. See errors for details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.items.info</td>
<td>List of objects that contains additional information about the processing.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;info&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;code&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;ruleSysId&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.items.info.code</td>
<td>Reclassification type that was skipped.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• SKIPPED_CLASS_SWITCH</td>
</tr>
<tr>
<td></td>
<td>• SKIPPED_CLASS_DOWNGRADE</td>
</tr>
<tr>
<td></td>
<td>• SKIPPED_CLASS_UPGRADE</td>
</tr>
<tr>
<td>result.items.info.message</td>
<td>Message that provides additional insights into the reason for skipping the reclassification.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.items.info.ruleSysId</td>
<td>Sys_id of the reclassification restriction rule that was matched.</td>
</tr>
<tr>
<td></td>
<td>Applicable only when IRE skips reclassification due to reclassification restriction rule. This value is empty if the reclassification is skipped due to a payload or global flag.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.items.inputIndices</td>
<td>Array of index values for CIs from the request body that correspond to this CI.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.items.markers</td>
<td>Array of marker values for internal use.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.items.mergedPayloadIds</td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.items.operation</td>
<td>The operation performed for this CI. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• DELETE: An existing CI is removed from the target table.</td>
</tr>
<tr>
<td></td>
<td>• INSERT: The CI is inserted into the target table as a new record.</td>
</tr>
<tr>
<td></td>
<td>• NO_CHANGE: No operation is performed for the CI.</td>
</tr>
<tr>
<td></td>
<td>• UPDATE: An existing CI in the target table is updated.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.items.relatedItems</td>
<td>Array of objects in which each object describes a related CI (table lookup CI) from the request</td>
</tr>
<tr>
<td>result.items.relatedItems.className</td>
<td>The sys_class_name of the related item.</td>
</tr>
<tr>
<td>result.items.relatedItems.inputIndices</td>
<td>Array of index values for CIs and lookup items from the request body that correspond to this related item.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.items.relatedItems.inputIndices.mainIndex</td>
<td>Index value from the request body items array that corresponds to the CI parent of the related item.</td>
</tr>
<tr>
<td>result.items.relatedItems.inputIndices.subIndex</td>
<td>Index value from the request body items.lookup array that corresponds to the related item.</td>
</tr>
<tr>
<td>result.items.relatedItems.markers</td>
<td>Array of marker values for internal use.</td>
</tr>
<tr>
<td>result.items.relatedItems.mergedPayloadIds</td>
<td>Array of sys_id values for entries in the CMDB [cmdb_ire_partial_payloads] table that were merged into this CI during processing.</td>
</tr>
<tr>
<td>result.items.relatedSysIds</td>
<td>Array of sys_id values for related items (table lookup items) from the request body items.lookup array.</td>
</tr>
</tbody>
</table>

Notable values:
- **null**: No sys_id was identified for this related item.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.sysId</td>
<td>The sys_id found for this CI through identification.</td>
</tr>
<tr>
<td></td>
<td>Notable values:</td>
</tr>
<tr>
<td></td>
<td>• Unknown: Identification of this CI failed. See errors for details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.items.warnings</td>
<td>Array of objects in which each object describes a warning encountered while processing this CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>result.items.warnings.error</td>
<td>The type of warning encountered while processing this CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.items.warnings.message</td>
<td>The warning message encountered while processing this CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.logContextId</td>
<td>Context ID reported for this payload.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.relations</td>
<td>Array of objects in which each object describes a dependent relationship CI from the request body relations array.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>result.relations.className</code></td>
<td>The sys_class_name of this dependent relationship CI. Only supported values:</td>
</tr>
<tr>
<td></td>
<td>• cmdb_rel_ci: The CI Relationship table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>result.relations.errorCount</code></td>
<td>The number of errors encountered while processing this dependent relationship CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td><code>result.relations.errors</code></td>
<td>Array of objects in which each object describes an error encountered while processing this dependent relationship CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>

```json
"relations": {
  "className": "String",
  "errorCount": Number,
  "errors": [Array],
  "inputIndices": [Array],
  "markers": [Array],
  "mergedPayloadIds": [Array],
  "operation": "String",
  "warnings": [Array]
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.relations.errors.error</td>
<td>The type of error encountered while processing this dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td>result.relations.errors.message</td>
<td>The error message encountered while processing this dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td>result.relations.inputIndices</td>
<td>Array of index values for the dependent relationship CI objects in the request body relations array that correspond to this dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td>result.relations.markers</td>
<td>Array of marker values for internal use. Data type: Array</td>
</tr>
<tr>
<td>result.relations.mergedPayloadIds</td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this dependent relationship CI. Data type: Array</td>
</tr>
</tbody>
</table>
| result.relations.operation | The operation performed for this dependent relationship CI. Possible values: © 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSERT</td>
<td>The dependent relationship CI was inserted into the target table.</td>
</tr>
<tr>
<td>INSERT_AS_INCOMPLETE</td>
<td>The dependent relationship CI had errors and was inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.</td>
</tr>
<tr>
<td>INSERT_AS_PARTIAL</td>
<td>The dependent relationship CI had errors and was inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.</td>
</tr>
<tr>
<td>NO_CHANGE</td>
<td>No operation was performed for the dependent relationship CI.</td>
</tr>
<tr>
<td>UPDATE</td>
<td>An existing dependent relationship CI in the target table was updated.</td>
</tr>
</tbody>
</table>

Data type: String

result.relations.warnings

Array of objects in which each object describes a warning encountered while processing this dependent relationship CI.

Data type:

```json
{  
  "warnings": [  
    {  
      "error": "String",  
      "message": "String"  
    }  
  ]
}
```

result.relations.warnings.error

The type of warning encountered while processing this dependent relationship CI.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>result.relations.warnings.message</code></td>
<td>The warning message encountered while processing this dependent relationship CI.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

Submit a payload containing a Linux server (with serial number and network adapter related-item lookups) and an IP switch, with an IP Connection dependent relationship between them.

```bash
curl "https://instance.servicenow.com/api/now/identifyreconcile" \
--request POST \
--header "Accept: application/json" \
--header "Content-Type: application/json" \
--user "username:password" \
--data '{ "items": [{ "className": "cmdb_ci_linux_server", "values": { "sys_class_name": "cmdb_ci_linux_server", "name": "lnux101", "host_name": "lnux101.servicenow.com", "ip_address": "10.11.12.173", "os_name": "Linux Red Hat" }, }, "lookup": [{ "className": "cmdb_serial_number", "values": { "serial_number": "lnux101", "serial_number_type": "uuid", "valid": "true" } }, { "className": "cmdb_ci_network_adapter", "values": { "name": "eth0", "ip_address": "10.11.12.173", "mac_address": "00:55:51:21:26:2f" } } ] }
```
The Linux server CI is identified via a Hardware Rule search for its name on the Hardware [cmdb_ci_hardware] table and is processed as an update to an existing CI in the Linux Server [cmdb_ci_linux_server] table. The IP switch is similarly identified via a Hardware Rule search for its name on the Hardware table and updates an existing CI in the IP Switch [cmdb_ci_ip_switch] table. Finally, the IP Connection dependent relationship between the server and the switch is inserted as a new record in the CI Relationship [cmdb_rel_ci] table.
"mergedPayloadIds": [],
"className": "cmdb_ci_network_adapter"
],
"additionalRelatedItems": [],
"identifierEntrySysId": "556eb250c3400200d8d4bea192d3ae92",
"identificationAttempts": [
  
],
"identifierName": "Hardware Rule",
"attemptResult": "NO.Match",
"attributes": ["serial_number", "serial_number_type"],
"searchOnTable": "cmdb_serial_number",
"hybridEntryCiAttributes": []
],
"identifierName": "Hardware Rule",
"attemptResult": "SKIPPED",
"attributes": ["serial_number"],
"searchOnTable": "cmdb_ci_hardware",
"hybridEntryCiAttributes": []
],
"identifierName": "Hardware Rule",
"attemptResult": "MATCHED",
"attributes": ["name"],
"searchOnTable": "cmdb_ci_hardware",
"hybridEntryCiAttributes": []
]},
"markers": [],
"inputIndices": [0],
"mergedPayloadIds": [],
"errorCount": 0
],
"className": "cmdb_ci_ip_switch",
"operation": "UPDATE",
"sysId": "304481257f701200bee45f19befa915b",
"identifierEntrySysId": "556eb250c3400200d8d4bea192d3ae92",
"identificationAttempts": [
  
],
"identifierName": "Hardware Rule",
"attemptResult": "SKIPPED",
"attributes": ["serial_number", "serial_number_type"],
"searchOnTable": "cmdb_serial_number",
"hybridEntryCiAttributes": []
],
"identifierName": "Hardware Rule",
"attemptResult": "NO_MATCH",
"attributes": ["serial_number"],
"searchOnTable": "cmdb_ci_hardware",
"hybridEntryCiAttributes": []
]
"hybridEntryCiAttributes": [],
],
"identifierName": "Hardware Rule",
"attemptResult": "MATCHED",
"attributes": ["name"],
"searchOnTable": "cmdb_ci_hardware",
"hybridEntryCiAttributes": []
],
"markers": [],
"inputIndices": [1],
"mergedPayloadIds": [],
"errorCount": 0
],
"additionalCommittedItems": [],
"relations": [{
  "className": "cmdb_rel_ci",
  "operation": "INSERT",
  "markers": [],
  "inputIndices": [0],
  "mergedPayloadIds": [],
  "errorCount": 0
}]
"additionalCommittedRelations": []
}
}

**Example: Python request**

Submit a payload containing entries for two load-balancer pool member CIs, two load-balancer pool CIs, a BigIP load balancer CI, and two load-balancer service instance CI. Each pool owns one pool member and is owned by the BigIP load balancer, and both service instances run on the load balancer.

```python
# Need to install requests package for python
import requests

# Set the API endpoint URL for the request
url = 'https://instance.servicenow.com/api/now/identifyreconcile'

# Set credentials
user = 'username'
pwd = 'password'

# Set HTTP headers
headers = {
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
"Accept": "application/json",
"Content-Type": "application/json"
}

# Define JSON payload
payload = {
    "items": [
        {
            "className": "cmdb_ci_lb_pool_member",
            "values": {
                "name": "TestCI_name_0__0",
                "ip_address": "ip_address_0__0",
                "server_id": "server_id_0__0",
                "object_id": "object_id_0__0",
                "service_port": "0"
            }
        },
        {
            "className": "cmdb_ci_lb_pool_member",
            "values": {
                "name": "TestCI_name_1__0",
                "ip_address": "ip_address_1__0",
                "server_id": "server_id_1__0",
                "object_id": "object_id_1__0",
                "service_port": "1"
            }
        },
        {
            "className": "cmdb_ci_lb_pool",
            "values": {
                "name": "TestCI_name_0__0",
                "server_id": "server_id_0__0",
                "object_id": "object_id_0__0"
            }
        },
        {
            "className": "cmdb_ci_lb_pool",
            "values": {
                "name": "TestCI_name_1__0",
                "server_id": "server_id_1__0",
                "object_id": "object_id_1__0"
            }
        },
        {
            "className": "cmdb_ci_lb_bigip",
            "values": {
                "name": "TestCI_name_0__0",
                "serial_number": "serial_number_0__0"
            }
        }
    ]
}
# Issue the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, json=payload)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

All CIs and dependent relationship CIs are inserted as new records into their respective tables.

{  
    "result": {  
        "items": [{  
            "className": "cmdb_ci_lb_pool_member",  
            "operation": "INSERT",  
            "identificationAttempts": [],  
            "markers": [],  
            "inputIndices": [0],  
            "mergedPayloadIds": [],  
            "errorCount": 0  
        }, {  
            "className": "cmdb_ci_lb_pool_member",  
            "operation": "INSERT",  
            "identificationAttempts": [],  
            "markers": [],  
            "inputIndices": [1],  
            "mergedPayloadIds": [],  
            "errorCount": 0  
        }, {  
            "className": "cmdb_ci_lb_pool",  
            "operation": "INSERT",  
            "identificationAttempts": [],  
            "markers": [],  
            "inputIndices": [2],  
            "mergedPayloadIds": [],  
            "errorCount": 0  
        }, {  


© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
"className": "cmdb_ci_lb_pool",
"operation": "INSERT",
"identificationAttempts": [],
"markers": [],
"inputIndices": [3],
"mergedPayloadIds": [],
"errorCount": 0 },

"className": "cmdb_ci_lb_bigip",
"operation": "INSERT",
"identificationAttempts": [{
"identifierName": "Load Balancer",
"attemptResult": "SKIPPED",
"attributes": ["serial_number", "serial_number_type"],
"searchOnTable": "cmdb_serial_number",
"hybridEntryCiAttributes": ["name", "serial_number"]
}],

"identifierName": "Load Balancer",
"attemptResult": "NO_MATCH",
"attributes": ["name", "serial_number"],
"searchOnTable": "cmdb_ci_lb",
"hybridEntryCiAttributes": []
}],

"markers": [],
"inputIndices": [4],
"mergedPayloadIds": [],
"errorCount": 0 },

"className": "cmdb_ci_lb_service",
"operation": "INSERT",
"identificationAttempts": [],
"markers": [],
"inputIndices": [5],
"mergedPayloadIds": [],
"errorCount": 0 },

"className": "cmdb_ci_lb_service",
"operation": "INSERT",
"identificationAttempts": [],
"markers": [],
"inputIndices": [6],
"mergedPayloadIds": [],
"errorCount": 0 ]}
"additionalCommittedItems": [],
"relations": [{
  "className": "cmdb_rel_ci",
  "operation": "INSERT",
  "markers": [],
  "inputIndices": [0],
  "mergedPayloadIds": [],
  "errorCount": 0
}, {
  "className": "cmdb_rel_ci",
  "operation": "INSERT",
  "markers": [],
  "inputIndices": [1],
  "mergedPayloadIds": [],
  "errorCount": 0
}, {
  "className": "cmdb_rel_ci",
  "operation": "INSERT",
  "markers": [],
  "inputIndices": [2],
  "mergedPayloadIds": [],
  "errorCount": 0
}, {
  "className": "cmdb_rel_ci",
  "operation": "INSERT",
  "markers": [],
  "inputIndices": [3],
  "mergedPayloadIds": [],
  "errorCount": 0
}, {
  "className": "cmdb_rel_ci",
  "operation": "INSERT",
  "markers": [],
  "inputIndices": [4],
  "mergedPayloadIds": [],
  "errorCount": 0
}, {
  "className": "cmdb_rel_ci",
  "operation": "INSERT",
  "markers": [],
  "inputIndices": [5],
  "mergedPayloadIds": [],
  "errorCount": 0
}]
"additionalCommittedRelations": []
}
}

Identification and Reconciliation - POST /now/identifyreconcile/enhanced

Inserts or updates configuration items (CIs) in the CMDB by applying identification and reconciliation rules to the specified payload (request body). Use this API instead of updating the CMDB directly.

Use the Identification and Reconciliation - POST /now/identifyreconcile/queryEnhanced endpoint to simulate submission of a payload to this endpoint without committing changes to the database.

This method is similar to the Identification and Reconciliation - POST /now/identifyreconcile method, but adds the following functionality:

- Partial payloads
  - In case of an item having a warning or error, indicates if an item operation is INSERT_AS_PARTIAL or INSERT_INCOMPLETE.
  - Returns the sys_ids of partial payloads that were merged with existing partial payloads.
- Supports payload deduplicate feature.
- Generates a summary.

URL format

Versioned URL: /api/now/{api_version}/identifyreconcile/enhanced

Default URL: /api/now/identifyreconcile/enhanced

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>
### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Comma-separated list of name-value pairs representing Enhanced IRE options. As an example, to override the default values for the partial payloads, deduplicate payloads, and generate summary Enhanced IRE options: partial_payloads: false, partial_commits: false, deduplicate_payloads: false.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> By default or if partial_payloads is set to true, both partial_commits and deduplicate_payloads are enabled, even if they are set to false, as those features are essential for partial payloads functionality. For more on Enhanced IRE options, see Identification and Reconciliation engine (IRE).</td>
</tr>
<tr>
<td>sysparm_data_source</td>
<td>Identifies the source of the CI information. This must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
</tbody>
</table>

Valid values for each name-value pair:

- true: Enable this Enhanced IRE option.
- false: Disable this Enhanced IRE option.

Data type: String

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>Array of objects in which each object describes a CI to create or update.</td>
</tr>
</tbody>
</table>

Data type: Array
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.className</td>
<td>Required. The sys_class_name (class/table name) of the CI to create or update. This value can be any CMDB table, such as cmdb_ci_linux_server or cmdb_ci_win_server. Data type: String</td>
</tr>
<tr>
<td>items.internal_id</td>
<td>Unique identifier for this item in this payload. Can be any value, but must be unique within the payload. Data type: String</td>
</tr>
<tr>
<td>items.lookup</td>
<td>Array of objects in which each object describes a lookup-based identification record. These records are used to identify the top-level configuration item based on a lookup table that has a reference back to the Configuration Item [cmdb_ci] table. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"lookup": [
  {
    "className": "String",
    "internal_id": "String",
    "sys_object_source_info": {Object},
    "values": {Object}
  }
]
```
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| items.lookup.className                     | Required. The sys_class_name or class/table name of the CI to create or update. This value can be any CMDB class/table, such as cmdb_serial_number or cmdb_ci_network_adapter or a non-CMDB hierarchy class.  
Data type: String                                                                                                                                 |
| items.lookup.internal_id                  | Unique identifier for this lookup record in this payload. Can be any value, but must be unique within the payload.  
Data type: String                                                                                                                                 |
| items.lookup.sys_object_source_info       | Object describing a unique CI identifier for a specified data source. Different sources may have different name-value pairs for the same CI.  
Data type: Object                                                                                                                                 |
| items.lookup.sys_object_source_info.source_feed | Name that uniquely identifies the feed sending the data.  
© 2021 ServiceNow, Inc. All rights reserved.  
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.  
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated. |
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>this CI. Use if the source can have multiple feeds. Data type: String</td>
</tr>
<tr>
<td>items.lookup.sys_object_source_info.source_name</td>
<td>Data source for the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>items.lookup.sys_object_source_info.source_native_key</td>
<td>Unique key/id from the source for the CI. Can be any string that is unique to the item. Data type: String</td>
</tr>
<tr>
<td>items.lookup.sys_object_source_info.source_recency_timestamp</td>
<td>UTC date and time that CI was scanned. Data type: String Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>items.lookup.values</td>
<td>Object describing fields this lookup item as name/value pairs, where the name is the field name. Data type: Object Field names and types depend on the fields selected by the user, such as:</td>
</tr>
</tbody>
</table>

```json
"values": {
  "ip_address": "String",
  "mac_address": "String",
  "serial_number": "String"
}
```
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;serial_number_type&quot;:</td>
<td>&quot;String&quot;, &quot;valid&quot;: &quot;String&quot;</td>
</tr>
</tbody>
</table>

items.related

Array of objects in which each object describes an item record to create or update, based on a related table that has a reference to the top-level CI that is being identified. Rules in the Related Entries [cmdb_related_entry] table define what type of records can be in this array.

The related table may or may not extend the Configuration Item [cmdb_ci] table. These records are not used to identify the configuration item.

Data type: Array

"related": [
  {
    "className": "String",
    "internal_id": "String",
    "sys_object_source_info": {Object},
    "values": {Object}
  }
]

items.related.className

Required. The sys_class_name or class/table name of the related item to create or update.

Data type: String
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.related.internal_id</td>
<td>Unique identifier for this related item in this payload. Can be any value, but must be unique within the payload. Data type: String</td>
</tr>
<tr>
<td>items.related.sys_object_source_info</td>
<td>Object describing a unique CI identifier for a specified data source. Different sources may have different name-value pairs for the same CI. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>{ &quot;sys_object_source_info&quot;: { &quot;source_feed&quot;: &quot;String&quot;, &quot;source_name&quot;: &quot;String&quot;, &quot;source_native_key&quot;: &quot;String&quot;, &quot;source_recency_timestamp&quot;: &quot;String&quot; } }</td>
</tr>
<tr>
<td>items.related.sys_object_source_info.source_feed</td>
<td>Name that uniquely identifies the feed sending this related item. Use if the source can have multiple feeds. Data type: String</td>
</tr>
<tr>
<td>items.related.sys_object_source_info.source_name</td>
<td>Data source for the related item information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
</tbody>
</table>
## Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>items.related.sys_object_source_info.source_native_key</code></td>
<td>Unique key/id from the source for the related item. Can be any string that is unique to the item. Data type: String</td>
</tr>
<tr>
<td><code>items.related.sys_object_source_info.source_recency_timestamp</code></td>
<td>UTC date and time that the related item was scanned. Data type: String. Format: <code>YYYY-MM-DD HH:mm:ss</code></td>
</tr>
<tr>
<td><code>items.related.values</code></td>
<td>Object describing fields to create or update for this related item as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced <code>sys_id</code>. Data type: Object. Field names and types depend on the fields selected by the user, such as:</td>
</tr>
</tbody>
</table>

```json
"values": {
  "host_name": "String",
  "ip_address": "String",
  "name": "String",
  "os_name": "String",
  "sys_class_name": "String"
}
```

| `items.settings` | Object containing parameters defining the types of updates that are permitted. Data type: Object |

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;settings&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;items.settings.skipReclassificationRestrictionRules&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;items.settings.updateWithoutDowngrade&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;items.settings.updateWithoutSwitch&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;items.settings.updateWithoutUpgrade&quot;: Boolean</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>items.settings.skipReclassificationRestrictionRules</td>
<td>Flag that indicates whether IRE should skip running the Reclassification Restriction rule that matches the class for the payload item. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Skip running the rule.</td>
</tr>
<tr>
<td></td>
<td>• false: Run the rule.</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>items.settings.updateWithoutDowngrade</td>
<td>Indicates whether update and downgrade are both permitted for this CI. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Update the item but downgrade is not permitted.</td>
</tr>
<tr>
<td></td>
<td>• false: Both item update and downgrade are permitted.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>
## Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| `items.settings.updateWithoutSwitch` | Indicates whether this CI can be updated and its class switched. Valid values:  
- true: Update the item but class switching is not permitted.  
- false: Both item update and class switching are permitted.  
Data type: Boolean  
Default: false |
| `items.settings.updateWithoutUpgrade` | Indicates whether update and upgrade are both permitted for this CI. Valid values:  
- true: Update the item but upgrade is not permitted.  
- false: Both item update and upgrade are permitted.  
Data type: Boolean  
Default: false |
| `items.sys_object_source_info` | Object describing a unique CI identifier.  
Data type: Object  

```json
"items.sys_object_source_info": {  
"source_feed": "String",  
"source_name": "String",  
"source_native_key": "String",  
"source_id": "String",  
"source_system": "String",  
"source_url": "String",  
"source_type": "String",  
"source_version": "String",  
"source_weight": "String"}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.sys_object_source_info.source_feed</td>
<td>Name that uniquely identifies the feed sending this CI. Use if the source can have multiple feeds. Data type: String</td>
</tr>
<tr>
<td>items.sys_object_source_info.source_name</td>
<td>Data source for the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>items.sys_object_source_info.source_native_key</td>
<td>Unique key/id from the source for the CI. Can be any string that is unique to the item. Data type: String</td>
</tr>
<tr>
<td>items.sys_object_source_info.source_recency_timestamp</td>
<td>UTC date and time that CI was scanned. Data type: String Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>items.values</td>
<td>Object describing fields create or update for this CI as name/value pairs, where the name is the field name. For a reference field, the value must be referenced sys_id. Data type: Object</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Field names and types depend on the fields selected by the user, such as:</td>
</tr>
<tr>
<td></td>
<td>&quot;values&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;host_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;ip_address&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;os_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_class_name&quot;: &quot;String&quot; }</td>
</tr>
<tr>
<td>referenceItems</td>
<td>Array of objects in which each object describes a reference between two items in the payload. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;referenceItems&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;referenced&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;referencedBy&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;referenceField&quot;: &quot;String&quot; }</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>referenceItems.referenced</td>
<td>The <strong>internal_id</strong> defined for the item being referenced by another item. Data type: String</td>
</tr>
<tr>
<td>referenceItems.referencedBy</td>
<td>The <strong>internal_id</strong> defined for the item that references another item. Data type: String</td>
</tr>
<tr>
<td>referenceItems.referenceField</td>
<td>Name of the reference field in the class/table for the <strong>referencedBy</strong> item. Data type: String</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| relations | Array of objects in which each object describes a relationship between two items defined in the payload. An object in this array can use either of two formats:  
  - The object can define a relationship between two top-level items (or using `parent` and `child` name-value pairs, with values representing item indexes from the payload items array.  
  - The object can define a relationship between any two items, including top-level, related, or lookup items, using `parent_id` and `child_id` key/value pairs, with values representing `internal_id` values defined for those items.  
  
  **Data type:** Array  
  ```json
  "relations": [
    {
      "child": Number,
      "parent": Number,
      "sys_rel_source_info": (Object),
      "type": "String"
    }
  ]
  ```
|
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relations.child</td>
<td>The integer index of the object in the items array that represents the child in the relationship.</td>
</tr>
<tr>
<td>relations.child_id</td>
<td>The internal_id of the child item in the relationship.</td>
</tr>
<tr>
<td>relations.parent</td>
<td>The integer index of the object in the items array that represents the parent in the relationship.</td>
</tr>
<tr>
<td>relations.parent_id</td>
<td>The internal_id of the parent item in the relationship.</td>
</tr>
<tr>
<td>relations.sys_rel_source_info</td>
<td>Discovery source information for the relationship. For non-dependency relationship this information is in the Relationship Sources [sys_rel_source] table.</td>
</tr>
</tbody>
</table>

```json
"child_id": "String",
"parent_id": "String",
"sys_rel_source_info": {
  (Object),
  "type": "String"
}
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relations.sys_rel_source_info.source_name</td>
<td>Discovery source name. Default: Discovery source passed in the API method parameter.</td>
</tr>
<tr>
<td>relations.sys_rel_source_info.source_feed</td>
<td>Any string that is a sub-discovery/scan within the discovery source. Default: ‘UNKNOWN’ is stored in the source_feed column when creating a record in sys_rel_source table.</td>
</tr>
<tr>
<td>relations.type</td>
<td>The type of relationship existing between the parent and child items. This must be a name field value from the CI Relationship Type [cmdb_rel_type] table. Data type: String</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

Error message detail "Chunk [...] is not a valid entry" indicates that the **options** query parameter is not in the correct name-value pair format.

For example, specifying an **options** parameter of **partial** produces a 500 status code with "Chunk [partial] is not a valid entry" in the message detail.

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Array of results produced by the query.</td>
</tr>
</tbody>
</table>

Data type:

```
"result": [
  "addition",
  "addition",
  [Array],
  "items":
  "relation"
]
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>result.additionalCommittedItems</code></td>
<td>Array of objects, each object describes a CI not included in the request body, to insert or update.</td>
</tr>
<tr>
<td><code>result.additionalCommittedItems.className</code></td>
<td>The sys_class_name of this additional CI. Data type: String</td>
</tr>
<tr>
<td><code>result.additionalCommittedItems.errorCount</code></td>
<td>The number of errors encountered while processing this additional CI. Data type: Number</td>
</tr>
<tr>
<td><code>result.additionalCommittedItems.errors</code></td>
<td>Array of objects, each object describes an error encountered processing this additional CI.</td>
</tr>
<tr>
<td><code>result.additionalCommittedItems.errors.error</code></td>
<td>The type of error encountered while processing this additional CI.</td>
</tr>
</tbody>
</table>

```
"additional": [  
  
  "additional": 
  
  "class": "String",
  "error": "String",
  "id": "String",
  "input": "String",
  "markers": "String",
  "operates": "String",
  "sysId": "String"
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.additionalCommittedItems.errors.message</td>
<td>The error message encountered while processing this additional CI.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identificationAttempts</td>
<td>Array of objects, each object describes an attempt made to identify CI.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identificationAttempts.attemptResult</td>
<td>The outcome of this identification attempt. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• MATCHED: Identification succeeded. A unique CI was found in the rule table which exactly matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• MULTI_MATCH: Identification failed with an error. Duplicate CIs were found in the rule table when matching against the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• NO_MATCH: Identification failed. No CI was found in the rule table which matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• SKIPPED: Identification attempt not attempted. The attributes were specified.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identificationAttempts.attributes</td>
<td>Array of CI identifier entry attributes used during this additional CI identification attempt. Data type: Array. Attribute names and types depend on the request body data and the identifier in use, such as:</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identificationAttempts.hybridEntryCiAttributes</td>
<td>Array of CI identifier entry attributes used during this additional CI identification attempt. Data type: Array. Attribute names and types depend on the request body data and the identifier in use, such as:</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identificationAttempts.identifierName</td>
<td>The identifier rule used for this additional CI identification attempt. Data type: String.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identificationAttempts.searchOnTable</td>
<td>The name of the table searched for this additional CI identification attempt. Data type: String.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identifierEntrySysId</td>
<td>The sys_id for the identifier rule used to identify this additional CI.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.additionalCommittedItems.inputIndices</td>
<td>Array of index values for CIs from the request body items array that correspond to this additional CI.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.markers</td>
<td>Array of marker values for internal use.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.mergedPayloads</td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this additional CI.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.operation</td>
<td>The operation performed for this additional CI. Possible values:</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>• DELETE:</td>
<td>An existing CI is removed from the target table.</td>
</tr>
<tr>
<td>• INSERT:</td>
<td>The additional CI is inserted into the target table as a new record.</td>
</tr>
<tr>
<td>• NO_CHANGE:</td>
<td>No operation is performed for the additional CI.</td>
</tr>
<tr>
<td>• UPDATE:</td>
<td>An existing CI in the target table is updated.</td>
</tr>
<tr>
<td>• UPDATE_WITH_DOWNGRADE:</td>
<td>An existing CI in the target table is updated and its class is changed to a more generic class.</td>
</tr>
<tr>
<td>• UPDATE_WITH_SWITCH:</td>
<td>An existing CI in the target table is updated and its class is changed to another class that is not an ancestor or descendant of the original class.</td>
</tr>
<tr>
<td>• UPDATE_WITH_UPGRADE:</td>
<td>An existing CI in the target table is updated and its class is changed to a more specialized class (descendent class).</td>
</tr>
</tbody>
</table>

Data type: String

result.additionalCommittedItems.sysId

The sys_id found for this additional CI through identification.

Notable values:

• Unknown: Identification of this additional CI failed. See errors for details.

Data type: String

result.additionalCommittedItems.warnings

Array of objects in which each object describes a warning encountered while processing this additional CI.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.additionalCommittedItems.warnings.error</td>
<td>The type of warning encountered while processing this additional CI. Data type: String</td>
</tr>
<tr>
<td>result.additionalCommittedItems.warnings.message</td>
<td>The warning message encountered while processing this additional CI. Data type: String</td>
</tr>
<tr>
<td>result.additionalCommittedRelations</td>
<td>Array of objects in which each object describes a dependent relationship CI not included in the request body list to insert or update. Data type: Array</td>
</tr>
<tr>
<td>result.additionalCommittedRelations.className</td>
<td>The sys_class_name of this additional dependent relationship CI. Only supported value: • cmdb_rel_ci: The CI Relationship table. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>result.additionalCommittedRelations.errorCount</code></td>
<td>The number of errors encountered while processing this additional dependent relationship CI. Data type: Number</td>
</tr>
<tr>
<td><code>result.additionalCommittedRelations.errors</code></td>
<td>Array of objects in which each object describes an error encountered while processing this additional dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td><code>result.additionalCommittedRelations.errors.error</code></td>
<td>The type of error encountered while processing this additional dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td><code>result.additionalCommittedRelations.errors.message</code></td>
<td>The error message encountered while processing this additional dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td><code>result.additionalCommittedRelations.inputIndices</code></td>
<td>Array of index values for dependent relationship CI objects in the request body <code>relations</code> array that correspond to this additional dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td><code>result.additionalCommittedRelations.markers</code></td>
<td>Array of markers for internal use.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>result.additionalCommittedRelations.mergedPayloadIds</code></td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this additional dependent relationship CI.</td>
</tr>
</tbody>
</table>
| `result.additionalCommittedRelations.operation` | The operation performed for this additional dependent relationship CI. Possible values:  
  - **INSERT**: The dependent relationship CI is inserted into the target table as a new record.  
  - **INSERT_AS_INCOMPLETE**: The dependent relationship CI had errors and is inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.  
  - **INSERT_AS_PARTIAL**: The dependent relationship CI had errors and is inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table. |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• NO_CHANGE: No operation is performed for the dependent relationship CI.</td>
<td></td>
</tr>
<tr>
<td>• UPDATE: An existing dependent relationship CI in the target table is updated.</td>
<td></td>
</tr>
</tbody>
</table>

**result.items**

Array of objects in which each object describes a CI included in the request body `items` array.

Data type: Array

```json
"items": [
  {
    "additionalRelatedItems": [],
    "className": "String",
    "duplicateLookupIndices": {},
    "errorCount": Number,
    "errors": [],
    "identificationAttempts": [],
    "identifierEntrySysId": "String",
    "info": [],
    "inputIndices": [],
    "markers": [],
    "mergedPayloadIds": [],
    "operation": "String",
    "relatedItems": [],
    "relatedSysIds": [],
    "sysId": "String",
    "warnings": []
  }
]
```

**result.items.additionalRelatedItems**

Array of objects in which each object describes a related CI (table lookup CI) from the CMDB IRE Partial Payloads `[cmdb_ire_partial_payloads]` table.

Data type: Array

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.additionalRelatedItems.className</td>
<td>The sys_class_name of the related item. Data type: String</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.inputIndices</td>
<td>Array of index values for CIs from the request body items array that correspond to this related item. Data type: Array</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.markers</td>
<td>Array of marker values for internal use. Data type: Array</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.mergedPayloadIds</td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing. Data type: Array</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.sysId</td>
<td>The sys_id of this related CI. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.items.className</td>
<td>The <code>sys_class_name</code> of this CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.items.duplicateLookupIndices</td>
<td>Object containing arrays of index values for elements that reference the same related CI. The array name indicates the first element from the request body <code>items.lookup</code> array that references a given related CI, and the set of additional elements from the same array that reference the same related CI. Data type: Object.</td>
</tr>
<tr>
<td></td>
<td>This example shows a case where the second element in the <code>items.lookup</code> array references the same related CI as a later element from that array:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
|                         |   | "duplicateLookupIndices": 
<p>|                         |   |   &quot;2&quot;: [                                                                             |                                                                                           |                                                                                           |
|                         |   |     ]                                                                                     |                                                                                           |                                                                                           |
| result.items.errorCount | The number of errors encountered while processing this CI.                                                                                                                                                                                                                                                                                                                                                         |
|                         | Data type: Number                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| result.items.errors     | Array of objects in which each object describes an error encountered while processing CI.                                                                                                                                                                                                                                                                                                                                                                                        |
|                         | Data type: Array                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|                         |   | &quot;errors&quot;: [                                |                                                                                           |                                                                                           |
|                         |   |     {                                      |                                                                                           |                                                                                           |
|                         |   |       &quot;error&quot;: &quot;String&quot;,                  |                                                                                           |                                                                                           |
|                         |   |       &quot;message&quot;: &quot;String&quot;                   |                                                                                           |                                                                                           |
|                         |   |     }                                                                                     |                                                                                           |                                                                                           |
|                         |                                                                                       |                                                                                           |                                                                                           |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.errors.error</td>
<td>The type of error encountered while processing this CI. Data type: String</td>
</tr>
<tr>
<td>result.items.errors.message</td>
<td>The error message encountered while processing this CI. Data type: String</td>
</tr>
<tr>
<td>result.items.identificationAttempts</td>
<td>Array of objects in which each object describes an attempt made to identify this CI. Data type: Array</td>
</tr>
<tr>
<td>result.items.identificationAttempts.attemptResult</td>
<td>The outcome of this CI identification attempt. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• MATCHED: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• MULTI_MATCH: Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• NO_MATCH: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>• SKIPPED: Identification attempts</td>
<td>SKIPPED: Identification attempts were not attempted. The attributes required for this identifier rule table search were not provided, so the rule was not applied.</td>
</tr>
<tr>
<td>result.items.identificationAttempts.attributes</td>
<td>Array of CI identifier entry attributes used during this CI identification attempt.</td>
</tr>
<tr>
<td>result.items.identificationAttempts.hybridEntryCiAttributes</td>
<td>Array of CI identifier entry attributes used during this CI identification attempt.</td>
</tr>
<tr>
<td>result.items.identificationAttempts.identifierName</td>
<td>The identifier rule used for this CI identification attempt.</td>
</tr>
<tr>
<td>result.items.identificationAttempts.searchOnTable</td>
<td>The name of the table searched for this CI identification attempt.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| `result.items.identifierEntrySysId`            | The sys_id for the identifier rule used to identify this CI. Notable values:  
  • Unknown: Identification of this CI failed. See errors for details.  
  Data type: String                                                                                                                                                                                                                                                     |
| `result.items.info`                            | List of objects that contains additional information about the processing of the item.  
  Data type: Array  
  ```json
  "info": [
    {
      "code": "String",
      "message": "String",
      "ruleSysId": "String"
    }
  ]
  ```                                                                                                                                                                                                                                                                     |
| `result.items.info.code`                       | Reclassification type that was skipped. Possible values:  
  • SKIPPED_CLASS_SWITCH  
  • SKIPPED_CLASS_DOWNGRADE  
  • SKIPPED_CLASS_UPGRADE  
  Data type: String                                                                                                                                                                                                                                                     |
| `result.items.info.message`                    | Message that provides additional insights into the reason for skipping the reclassification.  
  Data type: String                                                                                                                                                                                                                                                     |
| `result.items.info.ruleSysId`                  | Sys_id of the reclassification restriction rule that was matched.  
  Applicable only when IRE skips reclassification due to reclassification restriction rule. This value is empty if the reclassification is skipped due to a payload or global flag.  
  Data type: String                                                                                                                                                                                                                                                     |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>result.items.inputIndices</code></td>
<td>Array of index values for CIs from the request body that correspond to this CI. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
</tbody>
</table>
|                               |   ```**inputIndices**```  
|                               |   ```{  
|                               |   | Number  ```  ```  
|                               | ```}  ```                                                                    |
| `result.items.markers`       | Array of marker values for internal use. Data type: Array                                                                                   |
|                               | ```                                                                                                                                         |
|                               |   ```markers```                                                                  |
| `result.items.mergedPayloadIds` | Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing. Data type: Array |
|                               | ```                                                                                                                                         |
|                               |   ```mergedPayloadIds```  
|                               |   ```String```                                                                  |
| `result.items.operation`     | The operation performed for this CI. Possible values:  
|                               | • DELETE: An existing CI is removed from the target table.  
|                               | • INSERT: The CI is inserted into the target table as a new record.  
|                               | • NO_CHANGE: No operation is performed for the CI.  
<p>|                               | • UPDATE: An existing CI in the target table is updated.  |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• <strong>UPDATE</strong> WITH <strong>DOWNGRADE</strong>: An existing CI in the target table is updated and its class is changed to a more generic class (ancestor class).</td>
<td></td>
</tr>
<tr>
<td>• <strong>UPDATE</strong> WITH <strong>SWITCH</strong>: An existing CI in the target table is updated and its class is changed to another class which is not an ancestor or descendent of the class.</td>
<td></td>
</tr>
<tr>
<td>• <strong>UPDATE</strong> WITH <strong>UPGRADE</strong>: An existing CI in the target table is updated and its class is changed to a more specialized class (descendent class).</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

```json
result.items.relatedItems
```

Array of objects in which each object describes a related CI (table lookup CI) from the request body items.lookup array.

Data type:

```json
"relatedItems": [
  {
    "className": "String",
    "inputIndices": [Array],
    "markers": [Array],
    "mergedPayloadIds": [Array],
    "sysId": "String"
  }
]
```

result.items.relatedItems.className

The sys_class_name of the related item.

Data type: String

result.items.relatedItems.inputIndices

Array of index values for CIs and lookup items from the request body items and items.lookup arrays that correspond to this related item.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.relatedItems.inputIndices.mainIndex</td>
<td>Index value from the request body <code>items</code> array. CI parent to the related item.</td>
</tr>
<tr>
<td>result.items.relatedItems.inputIndices.subIndex</td>
<td>Index value from the request body <code>items.lookup</code> array.</td>
</tr>
<tr>
<td>result.items.relatedItems.markers</td>
<td>Array of marker values for internal use.</td>
</tr>
<tr>
<td>result.items.relatedItems.mergedPayloadIds</td>
<td>Array of <code>String</code> values for entries in the CMDB IRE Partial Payloads <code>[cmdb_ire_partial_payloads]</code> table that were merged into this CI during processing.</td>
</tr>
<tr>
<td>result.items.relatedSysIds</td>
<td>Array of <code>String</code> values for related items (table lookup items) from the request body <code>items.lookup</code> array.</td>
</tr>
</tbody>
</table>

**Notable values:**
- `null`: No sys_id was identified for this related item.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.sysId</td>
<td>The sys_id found for this CI through identification. Notable values:</td>
</tr>
<tr>
<td></td>
<td>• Unknown: Identification of this CI failed. See errors for details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.items.warnings</td>
<td>Array of objects in which each object describes a warning encountered while</td>
</tr>
<tr>
<td></td>
<td>processing this CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>result.items.warnings.error</td>
<td>The type of warning encountered while processing this CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.items.warnings.message</td>
<td>The warning message encountered while processing this CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.logContextId</td>
<td>Context ID reported for this payload.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.relations</td>
<td>Array of objects in which each object describes a dependent relationship CI</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>result.relations.className</code></td>
<td>The <code>sys_class_name</code> of this dependent relationship CI. Only supported value:</td>
</tr>
<tr>
<td></td>
<td>• <code>cmdb_rel_ci</code>: The CI Relationship table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>result.relations.errorCount</code></td>
<td>The number of errors encountered while processing this dependent relationship.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td><code>result.relations.errors</code></td>
<td>Array of objects in which each object describes an error encountered while processing this dependent relationship.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>

```json
"errors": [  
  {   
    "error": "String",   
    "message": "String"  
  }  
]
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.relations.errors.error</td>
<td>The type of error encountered while processing this dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td>result.relations.errors.message</td>
<td>The error message encountered while processing this dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td>result.relations.inputIndices</td>
<td>Array of index values for the dependent relationship CI objects in the request body relations array that correspond to this dependent relationship CI. Data type: Array of Number</td>
</tr>
<tr>
<td>result.relations.markers</td>
<td>Array of markers for internal use. Data type: Array of String</td>
</tr>
<tr>
<td>result.relations.mergedPayloadIds</td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this dependent relationship CI. Data type: Array of String</td>
</tr>
<tr>
<td>result.relations.operation</td>
<td>The operation performed for this dependent relationship CI. Possible values: merge, replace, remove, add, update, move, create, delete, link,.unlink, resolve, defer, complete</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSERT:</td>
<td>The dependent relationship CI was inserted into the target table as a new record.</td>
</tr>
<tr>
<td>INSERT_AS_INCOMPLETE</td>
<td>The dependent relationship CI had errors and was inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.</td>
</tr>
<tr>
<td>INSERT_AS_PARTIAL</td>
<td>The dependent relationship CI had errors and was inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.</td>
</tr>
<tr>
<td>NO_CHANGE</td>
<td>No operation was performed for the dependent relationship CI.</td>
</tr>
<tr>
<td>UPDATE:</td>
<td>An existing dependent relationship CI in the target table was updated.</td>
</tr>
</tbody>
</table>

**Data type:** Array

result.relations.warnings

Array of objects in which each object describes a warning encountered while processing this dependent relationship CI.

**Data type:** Array

```
"warnings": [
  {
    "error": "String",
    "message": "String"
  }
]
```

result.relations.warnings.error

The type of warning encountered while processing this dependent relationship CI.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.relations.warnings.message</td>
<td>The warning message encountered while processing this dependent relationship CI.</td>
</tr>
</tbody>
</table>

**Example: cURL request**
Submit a partial payload for a computer CI.

```
curl "https://instance.servicenow.com/api/now/identifyreconcile/enhanced" \  
  --request POST \  
  --header "Accept: application/json" \  
  --header "Content-Type: application/json" \  
  --user "username":"password" \  
  --data '{ \  
    "items": [ \  
      { \  
        "className": "cmdb_ci_computer", \  
        "sys_object_source_info": { \  
          "source_name": "ServiceNow", \  
          "source_native_key": "ncomputer3367" \  
        }, \  
        "values": { \  
          "cpu_core_count": "6", \  
          "ram": "8192", \  
          "manufacturer": "DELL", \  
          "company": "My Company" \  
        } \  
      } \  
    ] \  
  }'  
```

The response shows that the payload has been inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.

```json
{
  "result": {
    "items": [{
      "className": "cmdb_ci_computer",
      "operation": "INSERT_AS_PARTIAL",
      "sysId": "Unknown",
      "partialSysIds": ["1aec79151bb45010593876a61a4bcb28"]
    }
  ]
}
```
"identifierEntrySysId": "Unknown",
"errors": [],
"warnings": [{
  "error": "MISSING_MATCHING_ATTRIBUTES",
  "message": "In payload missing minimum set of input values for criterion (matching) attributes from identify rule for table [cmdb_ci_hardware]. Add these input values in payload item '{"className":"cmdb_ci_computer","values":{"discovery_source":"ServiceNow","company":"My Company","cpu_core_count":"6","ram":"8192","manufacturer":"DELL","sys_object_source_info":{"source_name":"ServiceNow","source_native_key":"ncomputer3367"},"settings":{},"sys_ire_info":{"ire_received_time":"2020-05-14 20:14:51"}}'",
  "identificationAttempts": [{
    "info": "sys_object_source NO_MATCH",
    "identifierName": "",
    "attemptResult": "NO_MATCH",
    "attributes": [],
    "hybridEntryCiAttributes": []
  }, {
    "identifierName": "Hardware Rule",
    "attemptResult": "SKIPPED",
    "attributes": ["serial_number", "serial_number_type"],
    "searchOnTable": "cmdb_serial_number",
    "hybridEntryCiAttributes": []
  }, {
    "identifierName": "Hardware Rule",
    "attemptResult": "SKIPPED",
    "attributes": ["serial_number"],
    "searchOnTable": "cmdb_ci_hardware",
    "hybridEntryCiAttributes": []
  }, {
    "identifierName": "Hardware Rule",
    "attemptResult": "SKIPPED",
    "attributes": ["name"],
    "searchOnTable": "cmdb_ci_hardware",
    "hybridEntryCiAttributes": []
  }, {
    "identifierName": "Hardware Rule",
    "attemptResult": "SKIPPED",
    "attributes": ["mac_address", "name"],
    "searchOnTable": "cmdb_ci_network_adapter",
    "hybridEntryCiAttributes": []
  }],
}]}
Submit a second partial payload that completes the description of the computer CI, with the summary generation Enhanced IRE option enabled.

```
curl  
    "https://instance.servicenow.com/api/now/identifyreconcile/enhanced?options=generate_summary:true" 
    --request POST 
    --header "Accept: application/json" 
    --header "Content-Type: application/json" 
    --user "username":"password" 
    --data '{ 
    
    }'
```

The response shows that the computer CI is inserted, and displays a summary of operations performed.

```
{
    "result": {
        "relations": [],
        "hasError": false,
        "additionalCommittedItems": [],
        "hasWarning": false,
        "items": [
            {
                "relatedSysIds": [735c79151bb45010593876a61a4bcb67],
                "identifierEntrySysId": "Unknown",
                "sysId": "ff5c79151bb45010593876a61a4bcb64",
                "errorCount": 0,
                "markers": [],
                "className": "cmdb_ci_computer",
                "inputIndices": [0]
            }
        ]
    }
}
```
0
},
"operation": "INSERT",
"relatedItems": [
{
  "operation": "INSERT",
  "warningCount": 0,
  "errors": [],
  ".sysId": "735c79151bb45010593876a61a4bcb67",
  "className": "cmdb_ci_network_adapter",
  "errorCount": 0,
  "markers": [],
  "inputIndices": [
    {
      "subIndex": 0,
      "mainIndex": 0
    }
  ],
  "mergedPayloadIds": []
},
"mergedPayloadIds": [
  "fe4c39151bb45010593876a61a4bcb6d"
],
"identificationAttempts": [
{
  "info": "sys_object_source NO_MATCH",
  "attemptResult": "NO_MATCH",
  "identifierName": "",
  "attributes": [],
  "hybridEntryCiAttributes": []
},
{
  "attemptResult": "SKIPPED",
  "identifierName": "Hardware Rule",
  "attributes": [
    "serial_number",
    "serial_number_type"
  ],
  "hybridEntryCiAttributes": [],
  "searchOnTable": "cmdb_serial_number"
},
{
  "attemptResult": "SKIPPED",
  "identifierName": "",
  "attributes": [],
  "hybridEntryCiAttributes": []
}
]
"identifierName": "Hardware Rule",
"attributes": [
    "serial_number"
],
"hybridEntryCiAttributes": [],
"searchOnTable": "cmdb_ci_hardware"
},
{
    "attemptResult": "SKIPPED",
    "identifierName": "Hardware Rule",
    "attributes": [
        "name"
    ],
    "hybridEntryCiAttributes": [],
    "searchOnTable": "cmdb_ci_hardware"
},
{
    "attemptResult": "NO_MATCH",
    "identifierName": "Hardware Rule",
    "attributes": [
        "mac_address",
        "name"
    ],
    "hybridEntryCiAttributes": [],
    "searchOnTable": "cmdb_ci_network_adapter"
}
],
"warningCount": 0
},
"additionalCommittedRelations": [],
"summary": {
    "cmdb_ci_network_adapter": {
        "mergedPartialPayloads": 0,
        "inserted": 1,
        "partial": 0,
        "warnings": 0,
        "incomplete": 0,
        "additionalInserted": 0,
        "unchanged": 0,
        "skipped": 0,
        "updated": 0,
        "errors": 0
    }
}
Example: Python request
Submit a payload containing entries for three Linux server CIs and generate a summary of operations performed for this payload.

```python
# Need to install requests package for python
import requests

# Set the API endpoint URL for the request
url = 'https://instance.servicenow.com/api/now/identifyreconcile/enhanced'

# Set credentials
user = 'username'
pwd = 'password'

# Set query parameters
params = {
    "options": "generate_summary:true"
}

# Set HTTP headers
headers = {
    "Accept": "application/json",
    "Content-Type": "application/json"
}

# Define JSON payload
payload = {
    "items": [
```
{  
  "className": "cmdb_ci_linux_server",
  "sys_object_source_info": {
    "source_name": "ServiceNow",
    "native_source_key": "linserver273"
  },
  "values": {
    "name": "linuxserver9423",
    "serial_number": "m51ced54747j89ed561n2bn674sa",
    "ram": "16384",
    "ip_address": "92.251.144.62",
    "company": "b7e831bdc0a80169015ae10f3c4d6cd",
    "discovery_source": "ServiceNow"
  }
},
{  
  "className": "cmdb_ci_linux_server",
  "sys_object_source_info": {
    "source_name": "ServiceNow",
    "native_source_key": "linserver913"
  },
  "values": {
    "name": "linuxserver9331",
    "serial_number": "2b1g6ds5ae48k4n2m4n32xc4c6s",
    "ram": "16384",
    "ip_address": "135.172.23.7",
    "company": "b7e831bdc0a80169015ae10f3c4d6cd",
    "discovery_source": "ServiceNow"
  }
},
{  
  "className": "cmdb_ci_linux_server",
  "sys_object_source_info": {
    "source_name": "ServiceNow",
    "native_source_key": "linserver972"
  },
  "values": {
    "name": "linuxserver9742",
    "serial_number": "54795s25c2h4ds8as5e10nca",
    "ram": "16384",
    "ip_address": "89.174.215.57",
    "company": "b7e831bdc0a80169015ae10f3c4d6cd",
    "discovery_source": "ServiceNow"
  }
}
# Issue the HTTP request
response = requests.post(url, auth=(user, pwd), params=params, headers=headers,
    json=payload)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.content)
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

All three CIs are updated, and the operation summary is generated.

{  "result": {  "relations": [],  "hasError": false,  "additionalCommittedItems": [],  "hasWarning": false,  "items": [  {  "operation": "UPDATE",  "identifierEntrySysId": "fb27f69cc3000200d8d4bea192d3ae67",  "mergedPayloadIds": [],  "sysId": "5472d2d11bb85010593876a61a4bcbeb",  "identificationAttempts": [  {  "info": "sys_object_source SKIPPED",  "attemptResult": "SKIPPED",  "identifierName": ",",  "attributes": [],  "hybridEntryCiAttributes": []  },  {  "attemptResult": "SKIPPED",  "identifierName": "Hardware Rule",  "attributes": [}
"serial_number",
"serial_number_type"
],
"hybridEntryCiAttributes": [],
"searchOnTable": "cmdb_serial_number"
},

{  
"attemptResult": "MATCHED",
"identifierName": "Hardware Rule",
"attributes": [  
"serial_number"
],
"hybridEntryCiAttributes": [],
"searchOnTable": "cmdb_ci_hardware"
}
],
"errorCount": 0,
"markers": [],
"warningCount": 0,
"className": "cmdb_ci_linux_server",
"inputIndices": [ 0 ]
},

{  
"operation": "UPDATE",
"identifierEntrySysId": "fb27f69cc3000200d8d4bea192d3ae67",
"mergedPayloadIds": [],
"sysId": "9472d2d1bb85010593876a61a4bcbee",
"identificationAttempts": [  
{  
"info": "sys_object_source SKIPPED",
"attemptResult": "SKIPPED",
"identifierName": "",
"attributes": [],
"hybridEntryCiAttributes": []
},
{  
"attemptResult": "SKIPPED",
"identifierName": "Hardware Rule",
"attributes": [  
"serial_number",
"serial_number_type"
]  
},
{  
"attemptResult": "SKIPPED",
"identifierName": "Hardware Rule",
"attributes": [  
"serial_number",
"serial_number_type"
]  
}  
],
"attemptResult": "SKIPPED",
"identifierName": "Hardware Rule",
"attributes": [  
"serial_number",
"serial_number_type"
]  
}
"hybridEntryCiAttributes": [],
"searchOnTable": "cmdb_serial_number"
},
{
"attemptResult": "MATCHED",
"identifierName": "Hardware Rule",
"attributes": [

"serial_number"
],
"hybridEntryCiAttributes": [],
"searchOnTable": "cmdb_ci_hardware"
},
"errorCount": 0,
"markers": [],
"warningCount": 0,
"className": "cmdb_ci_linux_server",
"inputIndices": [
1
]
},
{
"operation": "UPDATE",
"identifierEntrySysId": "fb27f69cc3000200d8d4bea192d3ae67",
"mergedPayloadIds": [],
"sysId": "5072d2d11bb85010593876a61a4bcbef",
"identificationAttempts": [

{
"info": "sys_object_source SKIPPED",
"attemptResult": "SKIPPED",
"identifierName": "",
"attributes": [],
"hybridEntryCiAttributes": []
},

{
"attemptResult": "SKIPPED",
"identifierName": "Hardware Rule",
"attributes": [

"serial_number",

"serial_number_type"
],
"hybridEntryCiAttributes": [],
"searchOnTable": "cmdb_serial_number"
},
Identification and Reconciliation - POST /now/identifyreconcile/query

Determines the CMDB operation (insert/update) to perform with the specified payload (request body), without committing the operations in the database.

Use this endpoint to simulate submission of a payload (request body) to the Identification and Reconciliation - POST /now/identifyreconcile endpoint.
Use the Identification and Reconciliation - POST /now/identifyreconcile/enhanced endpoint instead if you need to specify Enhanced IRE options. For more on Enhanced IRE options, see Identification and Reconciliation engine (IRE).

### URL format

Versioned URL: `/api/now/{api_version}/identifyreconcile/query`

Default URL: `/api/now/identifyreconcile/query`

### Supported request parameters

#### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

#### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_data_source</td>
<td>Identifies the source of the CI information. This must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table. Data type: String Default: Insert the API payload into the incomplete payloads table.</td>
</tr>
</tbody>
</table>

#### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>Array of objects in which each object describes a CI to create or update. Data type: Array</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>items.className</strong></td>
<td>Required. The <code>sys_class_name</code> (class/table name) of the CI to create or update. This value can be any CMDB table, such as <code>cmdb_ci_linux_server</code> or <code>cmdb_ci_win_server</code>. Data type: String</td>
</tr>
<tr>
<td><strong>items.internal_id</strong></td>
<td>Unique identifier for this item in this payload. Can be any value, but must be unique within the payload. Data type: String</td>
</tr>
<tr>
<td><strong>items.lookup</strong></td>
<td>Array of objects in which each object describes a lookup-based identification record. These records are used to identify the top-level configuration item based on a lookup table that has a reference back to the Configuration Item [cmdb_ci] table. Data type: Array</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.lookup.className</td>
<td>Required. The sys_class_name or class/table name of the CI to create or update. This value can be any CMDB class/table, such as cmdb_serial_number or cmdb_ci_network_adapter or a non-CMDB hierarchy class. Data type: String</td>
</tr>
<tr>
<td>items.lookup.internal_id</td>
<td>Unique identifier for this lookup record in this payload. Can be any value, but must be unique within the payload. Data type: String</td>
</tr>
<tr>
<td>items.lookup.sys_object_source_info</td>
<td>Object describing a unique CI identifier for a specified data source. Different sources may have different name-value pairs for the same CI. Data type: Object</td>
</tr>
</tbody>
</table>

```json
"lookup": [
  {
    "className": "String",
    "internal_id": "String",
    "sys_object_source_info": {
      "source_feed": "String",
      "source_name": "String"
    }
  }
]
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>items.lookup.sys_object_source_info.source_feed</code></td>
<td>Name that uniquely identifies the feed sending this CI. Use if the source can have multiple feeds. Data type: String</td>
</tr>
<tr>
<td><code>items.lookup.sys_object_source_info.source_name</code></td>
<td>Data source for the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td><code>items.lookup.sys_object_source_info.source_native_key</code></td>
<td>Unique key/id from the source for the CI. Can be any string that is unique to the item. Data type: String</td>
</tr>
<tr>
<td><code>items.lookup.sys_object_source_info.source_recency_timestamp</code></td>
<td>UTC date and time that CI was scanned. Data type: String Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td><code>items.lookup.values</code></td>
<td>Object describing fields this lookup item as name/value pairs, where the name is the field name. Data type: Object</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field names and types</td>
<td>Field names and types depend on the fields selected by the user, such as:</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&quot;values&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;ip_address&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;mac_address&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;serial_number&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;serial_number_type&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;valid&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>items.related</td>
<td>Array of objects in which each object describes an item record to create or update, based on a related table that has a reference to the top-level CI that is being identified. Rules in the Related Entry [cmdb_related_entry] table define what type of records can be in this array. The related table may or may not extend the Configuration Item [cmdb_ci] table. These records are not used to identify the configuration item. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&quot;related&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;className&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;internal_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_object_source_info&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.related.className</td>
<td>Required. The sys_class_name or class/table name of the related item to create or update. Data type: String</td>
</tr>
<tr>
<td>items.related.internal_id</td>
<td>Unique identifier for this related item in this payload. Can be any value, but must be unique within the payload. Data type: String</td>
</tr>
<tr>
<td>items.related.sys_object_source_info</td>
<td>Object describing a unique CI identifier for a specified data source. Different sources may have different name-value pairs for the same CI. Data type: Object</td>
</tr>
<tr>
<td>items.related.sys_object_source_info.source_feed</td>
<td>Name that uniquely identifies the feed sending this related item. Use if the source can have multiple feeds. Data type: String</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.related.sys_object_source_info.source_name</td>
<td>Data source for the related item information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table. Data type: String</td>
</tr>
<tr>
<td>items.related.sys_object_source_info.source_native_key</td>
<td>Unique key/id from the source for the related item. Can be any string that is unique to the item. Data type: String</td>
</tr>
<tr>
<td>items.related.sys_object_source_info.source_recency_timestamp</td>
<td>UTC date and time that the related item was scanned. Data type: String Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
<tr>
<td>items.related.values</td>
<td>Object describing fields to create or update for this related item as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced sys_id. Data type: Object</td>
</tr>
</tbody>
</table>

Field names and types depend on the fields selected by the user, such as:

```json
"values": {
    "host_name": "String",
    "ip_address": "String",
    "name": "String",
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.settings</td>
<td>Object containing parameters defining the types of updates that are permitted.</td>
<td>Object</td>
</tr>
</tbody>
</table>
| items.settings.skipReclassificationRestrictionRules | Flag that indicates whether IRE should skip running the Reclassification Restriction rule that matches the class for the payload item. Valid values:  
• true: Skip running the rule.  
• false: Run the rule.  
Default: false                                  | Boolean             |
| items.settings.updateWithoutDowngrade           | Indicates whether update and downgrade are both permitted for this CI. Valid values:  
• true: Update and downgrade are both permitted.  
• false: Update or downgrade is not permitted.  
Default: true                                   | Boolean             |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• true: Update the item but downgrade is not permitted.</td>
</tr>
<tr>
<td></td>
<td>• false: Both item update and downgrade are permitted.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td>Default: false</td>
</tr>
<tr>
<td>items.settings.updateWithoutSwitch</td>
<td>Indicates whether this CI can be updated and its class switched. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Update the item but class switching is not permitted.</td>
</tr>
<tr>
<td></td>
<td>• false: Both item update and class switching are permitted.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td>Default: false</td>
</tr>
<tr>
<td>items.settings.updateWithoutUpgrade</td>
<td>Indicates whether update and upgrade are both permitted for this CI. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Update the item upgrade is not permitted.</td>
</tr>
<tr>
<td></td>
<td>• false: Both item update and upgrade are permitted.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td>Default: false</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>items.sys_object_source_info</td>
<td>Object describing a unique CI identifier.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_object_source_info&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;source_feed&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;source_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;source_native_key&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;source_recency_timestamp&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>items.sys_object_source_info.source_feed</td>
<td>Name that uniquely identifies the feed sending this CI. Use if the source</td>
</tr>
<tr>
<td></td>
<td>can have multiple feeds.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.sys_object_source_info.source_name</td>
<td>Data source for the CI information. This value must be one of the choice</td>
</tr>
<tr>
<td></td>
<td>values defined for the discovery_source field of the Configuration Item</td>
</tr>
<tr>
<td></td>
<td>[cmdb_ci] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.sys_object_source_info.source_native_key</td>
<td>Unique key/id from the source for the CI. Can be any string that is unique</td>
</tr>
<tr>
<td></td>
<td>the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.sys_object_source_info.source_recency_timestamp</td>
<td>UTC date and time that CI was scanned.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Format: YYYY-MM-DD hh:mm:ss</td>
</tr>
</tbody>
</table>
**Request body parameters (XML or JSON) (continued)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.values</td>
<td>Object describing fields to create or update for this CI as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced sys_id. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>Field names and types depend on the fields selected by the user, such as:</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&quot;values&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;host_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;ip_address&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;os_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_class_name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>referenceItems</td>
<td>Array of objects in which each object describes a reference between two items in the payload. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&quot;referenceItems&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;referenced&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;referencedBy&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;referenceField&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>referenceItems.referenced</td>
<td>The <strong>internal_id</strong> defined for the item being referenced by another item. Data type: String</td>
</tr>
</tbody>
</table>
## Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>referenceItems.referencedBy</td>
<td>The <strong>internal_id</strong> defined for the item that references another item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>referenceItems.referenceField</td>
<td>Name of the reference field in the class/table for the <strong>referencedBy</strong> item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
| relations                                 | Array of objects in which each object describes a relationship between two items defined in the payload. An object in this array can use either of two formats:
|                                           | • The object can define a relationship between two top-level items using **parent** and **child** name-value pairs, with values representing item indexes from the payload **items** array. |
|                                           | • The object can define a relationship between any two items, including top-level, related, or lookup items, using **parent_id** and **child_id** key/value pairs, with values representing **internal_id** values defined for those items. |
|                                           | Data type: Array                                                           |

```json
"relations": [
]
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relations.child</td>
<td>The integer index of the object in the items array that represents the child in the relationship. Data type: Number</td>
</tr>
<tr>
<td>relations.child_id</td>
<td>The internal_id of the child item in the relationship. Data type: String</td>
</tr>
<tr>
<td>relations.parent</td>
<td>The integer index of the object in the items array that represents the parent in the relationship. Data type: Number</td>
</tr>
<tr>
<td>relations.parent_id</td>
<td>The internal_id of the parent item in the relationship. Data type: String</td>
</tr>
<tr>
<td>relations.sys_rel_source_info</td>
<td>Discovery source information for the</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>relationship. For non-dependency relationship, this information is in the Relationship Sources [sys_rel_source] table. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_rel_source_info&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;source_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;source_feed&quot;: &quot;String&quot; }</td>
</tr>
<tr>
<td>relations.sys_rel_source_info.source_name</td>
<td>Discovery source name. Default: Discovery source passed in the API method parameter.</td>
</tr>
<tr>
<td>relations.sys_rel_source_info.source_feed</td>
<td>Any string that is a sub-discovery/scan within the discovery source. Default: 'UNKNOWN' is stored in the source_feed column when creating a record in sys_rel_source table.</td>
</tr>
<tr>
<td>relations.type</td>
<td>The type of relationship existing between the parent and child items. This must be a name field value from the CI Relationship Type [cmdb_rel_type] table. Data type: String</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
## Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

## Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

## Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Array of results produced by the query. Data type: Array</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>result.additionalCommittedItems</code></td>
<td>Array of objects in which each object describes a CI not included in the request body to insert or update.</td>
<td>Array</td>
</tr>
<tr>
<td><code>result.additionalCommittedItems.className</code></td>
<td>The sys_class_name of this additional CI.</td>
<td>String</td>
</tr>
<tr>
<td><code>result.additionalCommittedItems.errorCount</code></td>
<td>The number of errors encountered while processing this additional CI.</td>
<td>Number</td>
</tr>
<tr>
<td><code>result.additionalCommittedItems.errors</code></td>
<td>Array of objects in which each object describes an error encountered while processing this additional CI.</td>
<td>Array</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.additionalCommittedItems.errors.error</td>
<td>The type of error encountered while processing this additional CI. Data type: String</td>
</tr>
<tr>
<td>result.additionalCommittedItems.errors.message</td>
<td>The error message encountered while processing this additional CI. Data type: String</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identificationAttempts</td>
<td>Array of objects describing an attempt made to identify this additional CI. Data type: Array</td>
</tr>
<tr>
<td>&quot;identificationAttempts&quot;: [</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;attemptResult&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;attributes&quot;: [Array],</td>
<td></td>
</tr>
<tr>
<td>&quot;hybridEntryCiAttributes&quot;: [Array],</td>
<td></td>
</tr>
<tr>
<td>&quot;identifierName&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;searchOnTable&quot;: [Array]</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>result.additionalCommittedItems.identificationAttempts.attemptResult</td>
<td>The outcome of this additional CI identification attempt. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• MATCHED: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• MULTI_MATCH: Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.</td>
</tr>
<tr>
<td></td>
<td>• NO_MATCH: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| result.additionalCommittedItems.identificationAttempts.attributes | Array of CI identifier entry attributes used during this additional CI identification attempt. Data type: Array | Attribute names and types depend on the request body data and the identifier in use, such as:

```
"attributes": [
  "serial_number": "String",
  "serial_number_type": "String"
]
```

| result.additionalCommittedItems.identificationAttempts.hybridEntryCiAttributes | Array of CI identifier entry attributes used during this additional CI identification attempt. Data type: Array | Attribute names and types depend on the request body data and the identifier in use, such as:

```
"hybridEntryCiAttributes": [
  "name": "String",
  "serial_number": "String"
]
```

| result.additionalCommittedItems.identificationAttempts.identifierName | The identifier rule used for this additional CI identification attempt. Data type: String |

<p>| result.additionalCommittedItems.identificationAttempts.searchOnTable | The name of the table searched for this additional CI identification attempt. |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| result.additionalCommittedItems.identifierEntrySysId     | The sys_id used to identify this additional CI. Notable values:  
* Unknown: Identification of this additional CI failed. See errors for details.                                               |
<p>| result.additionalCommittedItems.inputIndices             | Array of index values for CIs from the request body items array that correspond to this additional CI.                                                                                                      |
| result.additionalCommittedItems.markers                  | Array of marker values for internal use.                                                                                                                                                                    |
| result.additionalCommittedItems.mergedPayloads           | Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this additional CI.                                            |
| result.additionalCommittedItems.operation                | The operation performed for this additional CI. Possible values:                                                                                                                                           |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELETE</td>
<td>An existing CI is removed from the target table.</td>
</tr>
<tr>
<td>INSERT</td>
<td>The additional CI is inserted into the target table as a new record.</td>
</tr>
<tr>
<td>NO_CHANGE</td>
<td>No operation is performed for the additional CI.</td>
</tr>
<tr>
<td>UPDATE</td>
<td>An existing CI in the target table is updated.</td>
</tr>
<tr>
<td>UPDATE_WITH_DOWNGRADE</td>
<td>An existing CI in the target table is updated and its class is changed to a more generic class (ancestor class).</td>
</tr>
<tr>
<td>UPDATE_WITH_SWITCH</td>
<td>An existing CI in the target table is updated and its class is changed to another class which is not an ancestor or descendent class.</td>
</tr>
<tr>
<td>UPDATE_WITH_UPGRADE</td>
<td>An existing CI in the target table is updated and its class is changed to a more specialized class (descendent class).</td>
</tr>
</tbody>
</table>

**Data type:** String

result.additionalCommittedItems.sysId

The sys_id found for this additional CI through identification.

Notable values:
- Unknown: Identification of this additional CI failed. See errors for details.

**Data type:** String

result.additionalCommittedItems.warnings

Array of objects in which each object describes a warning encountered while processing this additional CI.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
<th>Additional Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.additionalCommittedItems.warnings.error</td>
<td>The type of warning encountered while processing this additional CI.</td>
<td>String</td>
<td>Description: The type of warning encountered while processing this additional CI.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.warnings.message</td>
<td>The warning message encountered while processing this additional CI.</td>
<td>String</td>
<td>Description: The warning message encountered while processing this additional CI.</td>
</tr>
<tr>
<td>result.additionalCommittedRelations</td>
<td>Array of objects describing dependent relationships CI not included in the request body to insert or update.</td>
<td>Array</td>
<td>Description: Array of object describing dependent relationships CI not included in the request body to insert or update.</td>
</tr>
<tr>
<td>result.additionalCommittedRelations.className</td>
<td>The sys_class_name of this additional dependent relationship CI. Only supported value: cmdb_rel_ci: The CI Relationship table.</td>
<td>String</td>
<td>Description: The sys_class_name of this additional dependent relationship CI. Only supported value: cmdb_rel_ci: The CI Relationship table.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>result.additionalCommittedRelations.errorCount</td>
<td>The number of errors encountered while processing this additional dependent relationship CI. Data type: Number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>result.additionalCommittedRelations.errors</td>
<td>Array of objects in which each object describes an error encountered while processing this additional dependent relationship CI. Data type: Array.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>result.additionalCommittedRelations.errors.error</td>
<td>The type of error encountered while processing this additional dependent relationship CI. Data type: String.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>result.additionalCommittedRelations.errors.message</td>
<td>The error message encountered while processing this additional dependent relationship CI. Data type: String.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>result.additionalCommittedRelations.inputIndices</td>
<td>Array of index values for dependent relationship CI objects in the request body relations array that correspond to this additional dependent relationship CI. Data type: Array.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>result.additionalCommittedRelations.markers</td>
<td>Array of markers for internal use.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>result.additionalCommittedRelations.mergedPayloadIds</td>
<td>Array of system-generated partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this additional dependent relationship CI.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;mergedPayloadIds&quot;: [&quot;String&quot;]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| result.additionalCommittedRelations.operation                      | The operation performed for this additional dependent relationship CI. Possible values:  
  - INSERT: The dependent relationship CI is inserted into the target table as a new record.  
  - INSERT_AS_INCOMPLETE: The dependent relationship CI had errors and is inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.  
  - INSERT_AS_PARTIAL: The dependent relationship CI had errors and is inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.  |

© 2021 ServiceNow, Inc. All rights reserved.  
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.  
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO_CHANGE</td>
<td>No operation is performed for the dependent relationship CI.</td>
</tr>
<tr>
<td>UPDATE</td>
<td>An existing dependent relationship CI in the target table is updated.</td>
</tr>
</tbody>
</table>

**Data type:** String

- **result.items**
  - Array of objects in which each object describes a CI included in the request body `items` array.
  - **Data type:** Array
    - "items": {
        "additionalRelatedItems": [Array],
        "className": "String",
        "duplicateLookupIndices": {Object},
        "errorCount": Number,
        "errors": [Array],
        "identificationAttempts": [Array],
        "identifierEntrySysId": "String",
        "info": [Array],
        "inputIndices": [Array],
        "markers": [Array],
        "mergedPayloadIds": [Array],
        "operation": "String",
        "relatedItems": [Array],
        "relatedSysIds": [Array],
        "sysId": "String",
        "warnings": [Array]
    }

- **result.items.additionalRelatedItems**
  - Array of objects in which each object describes a related CI (table lookup CI) from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.
  - **Data type:** Array
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.additionalRelatedItems.className</td>
<td>The sys_class_name of the related item. Data type: String</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.inputIndices</td>
<td>Array of index values for CIs from the request body items array that correspond to this related item. Data type: Array</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.markers</td>
<td>Array of marker values for internal use. Data type: Array</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.mergedPayloadIds</td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing. Data type: Array</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.sysId</td>
<td>The sys_id of this related CI. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.items.className</td>
<td>The <code>sys_class_name</code> of this CI. Data type: String</td>
</tr>
<tr>
<td>result.items.duplicateLookupIndices</td>
<td>Object containing arrays of index values for array elements that reference the same related CI. The array name indicates the first element from the request body <code>items.lookup</code> array that references a given related CI. Its values are the set of additional elements from the same array that reference the same related CI. Data type: Object. This example shows a case where the second element in the <code>items.lookup</code> array references the same related CI as a later element from that array:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.items.errorCount</td>
<td>The number of errors encountered while processing this CI. Data type: Number</td>
</tr>
<tr>
<td>result.items.errors</td>
<td>Array of objects in which each object describes an error encountered while processing this CI. Data type: Array.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>result.items.errors.error</td>
<td>The type of error encountered while processing this CI. Data type: String</td>
</tr>
<tr>
<td>result.items.errors.message</td>
<td>The error message encountered while processing this CI. Data type: String</td>
</tr>
<tr>
<td>result.items.identificationAttempts</td>
<td>Array of objects in which each object describes an attempt made to identify this CI. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"identificationAttempts": [
    {
      "attemptResult": "String",
      "attributes": [Array],
      "hybridEntryCiAttributes": [Array],
      "identifierName": "String",
      "searchOnTable": [Array]
    }
]
```

<table>
<thead>
<tr>
<th>result.items.identificationAttempts.attemptResult</th>
<th>The outcome of this CI identification attempt. Possible values:</th>
</tr>
</thead>
<tbody>
<tr>
<td>MATCHED</td>
<td>Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.</td>
</tr>
<tr>
<td>MULTI_MATCH</td>
<td>Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.</td>
</tr>
<tr>
<td>NO_MATCH</td>
<td>Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• SKIPPED: Identification not attempted. The attributes required for this identifier rule table search were not provided, so the rule was not applied.</td>
<td>-</td>
</tr>
</tbody>
</table>

**result.items.identificationAttempts.attributes**

Array of CI identifier entry attributes used during this CI identification attempt.

**result.items.identificationAttempts.hybridEntryCiAttributes**

Array of CI identifier entry attributes used during this CI identification attempt.

<table>
<thead>
<tr>
<th>Attribute names and types depend on the request body data and the identifier in use.</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;attributes&quot;:</td>
</tr>
<tr>
<td>&quot;serial_number&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;serial_number_type&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>]</td>
</tr>
</tbody>
</table>

**result.items.identificationAttempts.identifierName**

The identifier rule used for this CI identification attempt.

**result.items.identificationAttempts.searchOnTable**

The name of the table searched for this CI identification attempt.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.identifierEntrySysId</td>
<td>The sys_id to identify this CI. Notable values:</td>
</tr>
<tr>
<td></td>
<td>• Unknown: Identification of this CI failed. See errors for details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.items.info</td>
<td>List of objects that contains additional information about the processing of the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;info&quot;: [{</td>
</tr>
<tr>
<td></td>
<td>&quot;code&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;ruleSysId&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}]</td>
</tr>
<tr>
<td>result.items.info.code</td>
<td>Reclassification type that was skipped. Possible values: SKIPPED_CLASS_SWITCH, SKIPPED_CLASS_DOWNGRADE, SKIPPED_CLASS_UPGRADE</td>
</tr>
<tr>
<td>result.items.info.message</td>
<td>Message that provides additional insights into the reason for skipping the reclassification.</td>
</tr>
<tr>
<td>result.items.info.ruleSysId</td>
<td>Sys_id of the reclassification restriction rule that was matched.</td>
</tr>
<tr>
<td></td>
<td>Applicable only when IRE skips reclassification due to reclassification restriction rule. This value is empty if the reclassification is skipped due to a payload or global flag.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.items.inputIndices</td>
<td>Array of index values for CIs from the request body corresponding to this CI. Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.items.markers</td>
<td>Array of marker values for internal use.                                    Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.items.mergedPayloadIds</td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this CI. Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.items.operation</td>
<td>The operation performed for this CI. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• DELETE: An existing CI is removed from the target table.</td>
</tr>
<tr>
<td></td>
<td>• INSERT: The CI is inserted into the target table as a new record.</td>
</tr>
<tr>
<td></td>
<td>• NO_CHANGE: No operation is performed.</td>
</tr>
<tr>
<td></td>
<td>• UPDATE: An existing CI in the target table is updated.</td>
</tr>
</tbody>
</table>
NAME | DESCRIPTION
--- | ---
• UPDATE_WITH_DOWNGRADE: An existing CI in the target table is updated and its class is changed to a more generic class (ancestor class).
• UPDATE_WITH_SWITCH: An existing CI in the target table is updated and its class is changed to another class which is not an ancestor or descendent class.
• UPDATE_WITH_UPGRADE: An existing CI in the target table is updated and its class is changed to a more specialized class (descendent class).

result.items.relatedItems

Data type: Array

```
"relatedItems": [
  {
    "className": "String",
    "inputIndices": [Array],
    "markers": [Array],
    "mergedPayloadIds": [Array],
    "sysId": "String"
  }
]
```

result.items.relatedItems.className

The sys_class_name of the related item.

Data type: String

result.items.relatedItems.inputIndices

Array of index values for CIs and lookup items from the request body items and items.lookup arrays that correspond to this related item.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.relatedItems.inputIndices.mainIndex</td>
<td>Index value from the request body items array that corresponds to the CI parent of the related item.</td>
</tr>
<tr>
<td>result.items.relatedItems.inputIndices.subIndex</td>
<td>Index value from the request body items.lookup array that corresponds to the related item.</td>
</tr>
<tr>
<td>result.items.relatedItems.markers</td>
<td>Array of marker values for internal use.</td>
</tr>
<tr>
<td>result.items.relatedItems.mergedPayloadIds</td>
<td>Array of sys_id values for entries in the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged into this CI during processing.</td>
</tr>
<tr>
<td>result.items.relatedSysIds</td>
<td>Array of sys_id values for related items (table lookup items) from the request body items.lookup array.</td>
</tr>
<tr>
<td></td>
<td>Notable values:</td>
</tr>
<tr>
<td></td>
<td>• null: No sys_id was identified for this related item.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>result.items.sysId</code></td>
<td>The sys_id found for this CI through identification. Notable values:</td>
</tr>
<tr>
<td></td>
<td>• Unknown: Identification of this CI failed. See errors for details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>result.items.warnings</code></td>
<td>Array of objects in which each object describes a warning encountered CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td><code>result.items.warnings.error</code></td>
<td>The type of warning encountered while processing this CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>result.items.warnings.message</code></td>
<td>The warning message encountered while processing this CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>result.logContextId</code></td>
<td>Context ID reported for this payload.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>result.relations</code></td>
<td>Array of objects in which each object describes a dependent relationship CI</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.relations.className</td>
<td>The sys_class_name of this dependent relationship CI. Only supported values:</td>
</tr>
<tr>
<td></td>
<td>• cmdb_rel_ci: The CI Relationship table.</td>
</tr>
<tr>
<td>result.relations.errorCount</td>
<td>The number of errors encountered while processing this dependent relationship.</td>
</tr>
<tr>
<td>result.relations.errors</td>
<td>Array of objects, each object describes an error encountered while processing dependent relationship.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>result.relations.errors.error</td>
<td>The type of error encountered while processing this dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td>result.relations.errors.message</td>
<td>The error message encountered while processing this dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td>result.relations.inputIndices</td>
<td>Array of index values for the dependent relationship CI objects in the request body relations array that correspond to this dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td>result.relations.markers</td>
<td>Array of markers for internal use. Data type: Array</td>
</tr>
<tr>
<td>result.relations.mergedPayloadIds</td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td>result.relations.operation</td>
<td>The operation performed for this dependent relationship CI. Possible values:</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>INSERT</td>
<td>The dependent relationship CI was inserted into the target table as a new record.</td>
</tr>
<tr>
<td>UPDATE</td>
<td>An existing dependent relationship CI in the target table was updated.</td>
</tr>
<tr>
<td>INSERT_AS_INCOMPLETE</td>
<td>The dependent relationship CI had errors and was inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.</td>
</tr>
<tr>
<td>INSERT_AS_PARTIAL</td>
<td>The dependent relationship CI had errors and was inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.</td>
</tr>
<tr>
<td>NO_CHANGE</td>
<td>No operation was performed for the dependent relationship CI.</td>
</tr>
</tbody>
</table>

Data type: String

result.relations.warnings

Array of objects in which each object describes a warning encountered while processing this dependent relationship CI.

Data type:

```json
"warnings": [
    {
        "error": "String",
        "message": "String"
    }
]
```

result.relations.warnings.error

The type of warning encountered while processing this dependent relationship CI.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.relations.warnings.message</td>
<td>The warning message encountered while processing this dependent relationship CI.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

Simulate submission of a payload containing a Linux server (with serial number and network adapter related-item lookups) and an IP switch, with an IP Connection relationship between them.

```bash
curl "https://instance.servicenow.com/api/now/identifyreconcile/query" \
  --request POST  \
  --header "Accept: application/json"  \
  --header "Content-Type: application/json"  \
  --user "username":"password"  \
  --data '{ \
    "items": [{ \
      "className": "cmdb_ci_linux_server", \n      "values": { \
        "sys_class_name": "cmdb_ci_linux_server", \n        "name": "lnux101", \n        "host_name": "lnux101.servicenow.com", \n        "ip_address": "10.11.12.173", \n        "os_name": "Linux Red Hat" \n      }, \n      "lookup": [{ \
        "className": "cmdb_serial_number", \n        "values": { \n          "serial_number": "lnux101", \n          "serial_number_type": "uuid", \n          "valid": "true" \n        } \n      }, { \n        "className": "cmdb_ci_network_adapter", \n        "values": { \n          "name": "eth0", \n          "ip_address": "10.11.12.173", \n          "mac_address": "00:55:51:21:26:2f" \n        } \n      }] \n    } 
  }
```
The Linux server CI is identified via a Hardware Rule search for its name on the Hardware [cmdb_ci_hardware] table and is processed as an update to an existing CI in the Linux Server [cmdb_ci_linux_server] table. The IP switch is similarly identified via a Hardware Rule search for its name on the Hardware table and updates an existing CI in the IP Switch [cmdb_ci_ip_switch] table. Finally, the IP Connection relationship between the server and the switch is inserted as a new record in the CI Relationship [cmdb_rel_ci] table.
"sysId": "539747cac0a801640163e60735fbbf6e",
"errorCount": 0,
"markers": [],
"identificationAttempts": [
  {
    "attemptResult": "NO_MATCH",
    "identifierName": "Hardware Rule",
    "attributes": [
      "serial_number",
      "serial_number_type"
    ],
    "hybridEntryCiAttributes": [],
    "searchOnTable": "cmdb_serial_number"
  },
  {
    "attemptResult": "SKIPPED",
    "identifierName": "Hardware Rule",
    "attributes": ["serial_number"],
    "hybridEntryCiAttributes": [],
    "searchOnTable": "cmdb_ci_hardware"
  },
  {
    "attemptResult": "MATCHED",
    "identifierName": "Hardware Rule",
    "attributes": ["name"],
    "hybridEntryCiAttributes": [],
    "searchOnTable": "cmdb_ci_hardware"
  }
],
"inputIndices": [0],
"additionalRelatedItems": [],
"operation": "UPDATE",
"relatedItems": [
  {
    "markers": [],
    "mergedPayloadIds": [],
    "inputIndices": [
    ]
]
Example: Python request

Simulate submission of a payload containing entries for two load-balancer pool member CIs, two load-balancer pool CIs, a BigIP load balancer CI, and two load-balancer service instance CI. Each pool owns one pool member and is owned by the BigIP load balancer, and both service instances run on the load balancer.

```python
# Need to install requests package for python
import requests

# Set the API endpoint URL for the request
url = 'https://instance.servicenow.com/api/now/identifyreconcile/query'

# Set credentials
user = 'username'
pwd = 'password'

# Set HTTP headers
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Basic ' + base64.b64encode(user + ':' + pwd).decode(),
}
```
"Accept": "application/json",
"Content-Type": "application/json"
}

# Define JSON payload
payload = {
    "items": [
        {
            "className": "cmdb_ci_lb_pool_member",
            "values": {
                "name": "TestCI_name_0__0",
                "ip_address": "ip_address_0__0",
                "server_id": "server_id_0__0",
                "object_id": "object_id_0__0",
                "service_port": "0"
            }
        },
        {
            "className": "cmdb_ci_lb_pool_member",
            "values": {
                "name": "TestCI_name_1__0",
                "ip_address": "ip_address_1__0",
                "server_id": "server_id_1__0",
                "object_id": "object_id_1__0",
                "service_port": "1"
            }
        },
        {
            "className": "cmdb_ci_lb_pool",
            "values": {
                "name": "TestCI_name_0__0",
                "server_id": "server_id_0__0",
                "object_id": "object_id_0__0"
            }
        },
        {
            "className": "cmdb_ci_lb_pool",
            "values": {
                "name": "TestCI_name_1__0",
                "server_id": "server_id_1__0",
                "object_id": "object_id_1__0"
            }
        },
        {
            "className": "cmdb_ci_lb_bigip",
            "values": {
                "name": "TestCI_name_0__0",
                "serial_number": "serial_number_0__0"
            }
        }
    ]
}
{ "className": "cmdb_ci_lb_service", "values": {
  "fqdn": "fqdn_0__0", "port": "0", "name": "TestCI_name_0__0", "ip_address": "ip_address_0__0", "object_id": "object_id_0__0"
}
},
{ "className": "cmdb_ci_lb_service", "values": {
  "fqdn": "fqdn_1__0", "port": "1", "name": "TestCI_name_1__0", "ip_address": "ip_address_1__0", "object_id": "object_id_1__0"
}
}]
"relations": [{
  "parent": 2, "child": 0, "type": "Owns::Owned by"
}],
{ "parent": 3, "child": 1, "type": "Owns::Owned by"
},
{ "parent": 4, "child": 2, "type": "Owns::Owned by"
},
{ "parent": 4, "child": 3, "type": "Owns::Owned by"
},
{ "parent": 5, "child": 4, "type": "Runs on::Runs"
},
{ "parent": 6, "child": 4, "type": "Runs on::Runs"
}]

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
```python
# Issue the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, json=payload)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.content)
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

All CIs and relationship CIs are inserted as new records into their respective tables.

{
  "result": {
    "relations": [
      {
        "operation": "INSERT",
        "mergedPayloadIds": [],
        "className": "cmdb_rel_ci",
        "errorCount": 0,
        "markers": [],
        "inputIndices": [0]
      },
      {
        "operation": "INSERT",
        "mergedPayloadIds": [],
        "className": "cmdb_rel_ci",
        "errorCount": 0,
        "markers": [],
        "inputIndices": [1]
      },
      {
        "operation": "INSERT",
        "mergedPayloadIds": []
      }
    ]
  }
}```
"className": "cmdb_rel_ci",
"errorCount": 0,
"markers": [],
"inputIndices": [
  2
]
},
{
    "operation": "INSERT",
    "mergedPayloadIds": [],
    "className": "cmdb_rel_ci",
    "errorCount": 0,
    "markers": [],
    "inputIndices": [
        3
    ]
},
{
    "operation": "INSERT",
    "mergedPayloadIds": [],
    "className": "cmdb_rel_ci",
    "errorCount": 0,
    "markers": [],
    "inputIndices": [
        4
    ]
},
{
    "operation": "INSERT",
    "mergedPayloadIds": [],
    "className": "cmdb_rel_ci",
    "errorCount": 0,
    "markers": [],
    "inputIndices": [
        5
    ]
}
],
"items": [
    
    "operation": "INSERT",
    "mergedPayloadIds": [],
    "identificationAttempts": [],
    "errorCount": 0,

"markers": [],
"className": "cmdb_ci_lb_pool_member",
"inputIndices": [
  0
],

{
  "operation": "INSERT",
  "mergedPayloadIds": [],
  "identificationAttempts": [],
  "errorCount": 0,
  "markers": [],
  "className": "cmdb_ci_lb_pool_member",
  "inputIndices": [
    1
  ]
},

{
  "operation": "INSERT",
  "mergedPayloadIds": [],
  "identificationAttempts": [],
  "errorCount": 0,
  "markers": [],
  "className": "cmdb_ci_lb_pool",
  "inputIndices": [
    2
  ]
},

{
  "operation": "INSERT",
  "mergedPayloadIds": [],
  "identificationAttempts": [],
  "errorCount": 0,
  "markers": [],
  "className": "cmdb_ci_lb_pool",
  "inputIndices": [
    3
  ]
},

{
  "operation": "INSERT",
  "mergedPayloadIds": [],
  "identificationAttempts": [
"attemptResult": "SKIPPED",
"identifierName": "Load Balancer",
"attributes": [
  "serial_number",
  "serial_number_type"
],
"hybridEntryCiAttributes": [
  "name",
  "serial_number"
],
"searchOnTable": "cmdb_serial_number"
},
{
  "attemptResult": "NO_MATCH",
  "identifierName": "Load Balancer",
  "attributes": [
    "name",
    "serial_number"
  ],
  "hybridEntryCiAttributes": [],
  "searchOnTable": "cmdb_ci_lb"
}
],
"errorCount": 0,
"markers": []
"className": "cmdb_ci_lb_bigip",
"inputIndices": [4]
],
{
  "operation": "INSERT",
  "mergedPayloadIds": [],
  "identificationAttempts": [],
  "errorCount": 0,
  "markers": [],
  "className": "cmdb_ci_lb_service",
  "inputIndices": [5]
},
{
  "operation": "INSERT",
  "mergedPayloadIds": []
}
Identification and Reconciliation - POST /now/identifyreconcile/queryEnhanced

Determines the CMDB operation (insert/update) to perform with the specified payload (request body), without committing the operations in the database.

Use this endpoint to simulate submission of a payload (request body) to the Identification and Reconciliation - POST /now/identifyreconcile/enhanced endpoint.

This method is similar to the Identification and Reconciliation - POST /now/identifyreconcile/query method, but adds the following functionality:

• Partial payloads
  ◦ In case of an item having a warning or error, indicates if an item operation is INSERT_AS_PARTIAL or INSERT_INCOMPLETE.
  ◦ Returns the sys_ids of partial payloads that were merged with existing partial payloads.
• Supports payload deduplicate feature.
• Generates a summary.

URL format

Versioned URL: /api/now/{api_version}/identifyreconcile/queryEnhanced

Default URL: /api/now/identifyreconcile/queryEnhanced
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| options             | Comma-separated list of name-value pairs representing Enhanced IRE options. As an example, to override the default values for the partial payloads, partial commits, deduplicate payloads, and generate summary Enhanced IRE options: partial_payloads:false,partial_commits:false,deduplicate_payloads:

**Note:** By default or if `partial_payloads` is set to true, both `partial_commits` and `deduplicate_payloads` are enabled, even if they are set to false, as those features are essential for partial payloads functionality.

For more on Enhanced IRE options, see [Identification and Reconciliation engine (IRE)](https://go.to/snowdoc).

Valid values for each name-value pair:

- **true**: Enable this Enhanced IRE option.
- **false**: Disable this Enhanced IRE option.

Data type: String |
| sysparm_data_source | Identifies the source of the CI information. This must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table. Data type: String Default: Insert the API payload into the incomplete payloads table. |
**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>Array of objects in which each object describes a CI to create or update.</td>
<td>Array</td>
</tr>
<tr>
<td>items.className</td>
<td>Required. The sys_class_name (class/table name) of the CI to create or update. This value can be any CMDB table, such as cmdb_ci_linux_server or cmdb_ci_win_server.</td>
<td>String</td>
</tr>
<tr>
<td>items.internal_id</td>
<td>Unique identifier for this item in this payload. Can be any value, but must be unique within the payload.</td>
<td>String</td>
</tr>
<tr>
<td>items.lookup</td>
<td>Array of objects in which each object describes a lookup-based identification record. These records are used to identify the top-level configuration item based on a lookup table.</td>
<td>Array</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>that has a reference back to the Configuration Item [cmdb_ci] table. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;lookup&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;className&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;internal_id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_object_source_info&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;values&quot;: {Object}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>items.lookup.className</td>
<td>Required. The sys_class_name or class/table name of the CI to create or update. This value can be any CMDB class/table, such as cmdb_serial_number, cmdb_ci_network_adapter, or a non-CMDB hierarchy class. Data type: String</td>
</tr>
<tr>
<td>items.lookup.internal_id</td>
<td>Unique identifier for this lookup record in this payload. Can be any value, but must be unique within the payload. Data type: String</td>
</tr>
<tr>
<td>items.lookup.sys_object_source_info</td>
<td>Object describing a unique CI identifier for a specified data source. Different sources may have different name-value pairs for the same CI.</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>items.lookup.sys_object_source_info.source_feed</strong></td>
<td>Name that uniquely identifies the feed sending this CI. Use if the source can have multiple feeds.</td>
</tr>
<tr>
<td><strong>items.lookup.sys_object_source_info.source_name</strong></td>
<td>Data source for the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td><strong>items.lookup.sys_object_source_info.source_native_key</strong></td>
<td>Unique key/id from the source for the CI. Can be any string that is unique to the item.</td>
</tr>
<tr>
<td><strong>items.lookup.sys_object_source_info.source_recency_timestamp</strong></td>
<td>UTC date and time that the CI was scanned.</td>
</tr>
<tr>
<td><strong>items.lookup.values</strong></td>
<td>Object describing fields for this lookup item as name/data type.</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| value pairs, where the name is the field name. | Data type: Object  
Field names and types depend on the fields selected by the user, such as: |

```
"values": {
  "ip_address": "String",
  "mac_address": "String",
  "serial_number": "String",
  "serial_number_type": "String",
  "valid": "String"
}
```

| items.related         | Array of objects in which each object describes an item record to create or update, based on a related table that has a reference to the top-level CI that is being identified. Rules in the Related Entries [cmdb_related_entry] table define what type of record can be in this array.  
The related table may or may not extend the Configuration Item [cmdb_ci] table. These records are not used to identify the configuration item. |

```
"related": [
  {
    "className": "String",
  }
]
```
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.related.className</td>
<td>Required. The sys_class_name or class/table name of the related item to create or update. Data type: String</td>
</tr>
<tr>
<td>items.related.internal_id</td>
<td>Unique identifier for this related item in this payload. Can be any value, but must be unique within the payload. Data type: String</td>
</tr>
<tr>
<td>items.related.sys_object_source_info</td>
<td>Object describing a unique CI identifier for a specified data source. Different sources may have different name-value pairs for the same CI. Data type: Object</td>
</tr>
<tr>
<td>items.related.sys_object_source_info.source_feed</td>
<td>Name that uniquely identifies the feed sending this related item. Use if the sys_object_source_info contains a feed-related identifier.</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>source can have multiple feeds.</strong></td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>items.related.sys_object_source_info.source_name</td>
<td>Data source for the related item information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>items.related.sys_object_source_info.source_native_key</td>
<td>Unique key/id from the source for the related item. Can be any string that is unique to the item.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>items.related.sys_object_source_info.source_recency_timestamp</td>
<td>UTC date and time that related item was scanned.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Format: YYYY-MM-DD hh:mm:ss</td>
<td></td>
</tr>
<tr>
<td>items.related.values</td>
<td>Object describing fields to create or update for this related item as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced sys_id.</td>
</tr>
<tr>
<td>Data type: Object</td>
<td></td>
</tr>
<tr>
<td>Field names and types depend on the fields selected by the user, such as:</td>
<td></td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requests.items.name</td>
<td>Name</td>
</tr>
<tr>
<td>requests.items.description</td>
<td>Description</td>
</tr>
<tr>
<td>requests.items.values</td>
<td>&quot;values&quot;: { &quot;host_name&quot;: &quot;String&quot;, &quot;ip_address&quot;: &quot;String&quot;, &quot;name&quot;: &quot;String&quot;, &quot;os_name&quot;: &quot;String&quot;, &quot;sys_class_name&quot;: &quot;String&quot; }</td>
</tr>
<tr>
<td>requests.items.settings</td>
<td>Object containing parameters defining the types of updates that are permitted.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;settings&quot;: { &quot;skipReclassificationRestrictionRules&quot;: Boolean, &quot;updateWithoutDowngrade&quot;: Boolean, &quot;updateWithoutSwitch&quot;: Boolean, &quot;updateWithoutUpgrade&quot;: Boolean }</td>
</tr>
</tbody>
</table>
| requests.items.settings.skipReclassificationRestrictionRules | Flag that indicates whether IRE should skip running the Reclassification Restriction rule that matches the class for the payload item. Valid values:  
  - true: Skip running the rule.  
  - false: Run the rule. Default: false |
<p>| requests.items.settings.updateWithoutDowngrade | Indicates whether update and downgrade are both permitted for this CI. |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.settings.updateWithoutSwitch</td>
<td>Indicates whether this CI can be updated and its class switched. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Update the item but class switching is not permitted.</td>
</tr>
<tr>
<td></td>
<td>• false: Both item update and class switching are permitted.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>items.settings.updateWithoutUpgrade</td>
<td>Indicates whether update and upgrade are both permitted for this CI. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Update the item upgrade is not permitted.</td>
</tr>
<tr>
<td></td>
<td>• false: Both item update and upgrade are permitted.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| `items.sys_object_source_info` | Object describing a unique CI identifier.  
Data type: Object  

```json
"sys_object_source_info": {
  "source_feed": "String",
  "source_name": "String",
  "source_native_key": "String",
  "source_recency_timestamp": "String"
}
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| `items.sys_object_source_info.source_feed` | Name that uniquely identifies the feed sending this CI. Use if the source can have multiple feeds.  
Data type: String |

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| `items.sys_object_source_info.source_name` | Data source for the CI information. This value must be one of the choice values defined for the discovery_source field of the Configuration Item [cmdb_ci] table.  
Data type: String |

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| `items.sys_object_source_info.source_native_key` | Unique key/id from the source for the CI. Can be any string that is unique to the item.  
Data type: String |

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| `items.sys_object_source_info.source_recency_timestamp` | UTC date and time that CI was scanned.  
Data type: String  
Format: YYYY-MM-DD hh:mm:ss |
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.values</td>
<td>Object describing fields to create or update for this CI as name/value pairs, where the name is the field name. For a reference field, the value must be the referenced sys_id.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Object</td>
</tr>
<tr>
<td></td>
<td>Field names and types depend on the fields selected by the user, such as:</td>
</tr>
<tr>
<td></td>
<td>```json</td>
</tr>
<tr>
<td></td>
<td>&quot;values&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;host_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;ip_address&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;os_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_class_name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>referenceItems</td>
<td>Array of objects in which each object describes a reference between two items in the payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Array</td>
</tr>
<tr>
<td></td>
<td>```json</td>
</tr>
<tr>
<td></td>
<td>&quot;referenceItems&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;referenced&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;referencedBy&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;referenceField&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>referenceItems.referenced</td>
<td>The <strong>internal_id</strong> defined for the item being referenced by another item.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> String</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>referenceItems.referencedBy</td>
<td>The <code>internal_id</code> defined for the item that references another item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>referenceItems.referenceField</td>
<td>Name of the reference field in the class/table for the <code>referencedBy</code> item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>relations</td>
<td>Array of objects in which each object describes a relationship between two items defined in the payload. An object in this array can use either of two formats:</td>
</tr>
<tr>
<td></td>
<td>1. The object can define a relationship between two top-level items (only) using <code>parent</code> and <code>child</code> name-value pairs, with values representing item indexes from the payload <code>items</code> array.</td>
</tr>
<tr>
<td></td>
<td>2. The object can define a relationship between any two items, including top-level, related, or lookup items, using <code>parent_id</code> and <code>child_id</code> key-value pairs, with values representing <code>internal_id</code> values defined for those items.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>

```json
"relations": [
]
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relations.child</td>
<td>The integer index of the object in the items array that represents the child in the relationship. Data type: Number</td>
</tr>
<tr>
<td>relations.child_id</td>
<td>The internal_id of the child item in the relationship. Data type: String</td>
</tr>
<tr>
<td>relations.parent</td>
<td>The integer index of the object in the items array that represents the parent in the relationship. Data type: Number</td>
</tr>
<tr>
<td>relations.parent_id</td>
<td>The internal_id of the parent item in the relationship. Data type: String</td>
</tr>
<tr>
<td>relations.sys_rel_source_info</td>
<td>Discovery source information for the</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>relationship. For non-dependency relationships, this information is in the Relationship Sources [sys_rel_source] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td><em>sys_rel_source_info</em>: {</td>
</tr>
<tr>
<td></td>
<td><em>source_name</em>: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td><em>source_feed</em>: &quot;String&quot;</td>
</tr>
<tr>
<td>relations.sys_rel_source_info.source_name</td>
<td>Discovery source name.</td>
</tr>
<tr>
<td></td>
<td>Default: Discovery source passed in the API method parameter.</td>
</tr>
<tr>
<td>relations.sys_rel_source_info.source_feed</td>
<td>Any string that is a sub-discovery/scan within the discovery source.</td>
</tr>
<tr>
<td></td>
<td>Default: ‘UNKNOWN’ is stored in the source_feed column when creating a record in sys_rel_source table.</td>
</tr>
<tr>
<td>relations.type</td>
<td>The type of relationship existing between the parent and child items. The type must be a name field value from the CI Relationship Type [cmdb_rel_type] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

Error message detail "Chunk [...] is not a valid entry" indicates that the `options` query parameter is not in the correct name-value pair format. For example, specifying an `options` parameter of `partial` produces a 500 status code with "Chunk [partial] is not a valid entry" in the message detail.

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Array of results produced by the query.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>result.additionalCommittedItems</td>
<td>Array of objects in which each object describes a CI not included in the request body to insert or update.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.className</td>
<td>The sys_class_name of this additional CI.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.errorCount</td>
<td>The number of errors encountered while processing this additional CI.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.errors</td>
<td>Array of objects in which each object describes an error encountered while processing this additional CI.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>result.additionalCommittedItems.errors.error</td>
<td>The type of error encountered while processing this additional CI. Data type: String</td>
</tr>
<tr>
<td>result.additionalCommittedItems.errors.message</td>
<td>The error message encountered while processing this additional CI. Data type: String</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identificationAttempts</td>
<td>Array of objects in which each object describes an attempt made to identify this additional CI. Data type: Array</td>
</tr>
</tbody>
</table>
| result.additionalCommittedItems.identificationAttempts.attemptResult | The outcome of the identification attempt. Possible values:
- MATCHED: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.
- MULTI_MATCH: Identification failed with an error. Duplicate CIs were found. |

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO_MATCH</td>
<td>Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
</tr>
<tr>
<td>SKIPPED</td>
<td>Identification not attempted. The attributes required for this identifier rule table search were not provided, so the rule was not applied.</td>
</tr>
</tbody>
</table>

Data type: String

result.additionalCommittedItems.identificationAttempts.attributes

Array of CI identifier entry attributes used during this additional CI identification attempt. Data type: Array

Attribute names and types depend on the request body data and the identifier in use, such as:

```json
"attributes": {
  "serial_number": "String",
  "serial_number_type": "String"
}
```

result.additionalCommittedItems.identificationAttempts.hybridEntryCiAttributes

Array of CI identifier entry attributes used during this additional CI identification attempt. Data type: Array

Attribute names and types depend on the request body data and the identifier in use, such as:

```json
"hybridEntryCiAttributes": {
  "name": "String",
  "serial_number": "String"
}
```

result.additionalCommittedItems.identificationAttempts.identifierName

The identifier rule used for this additional CI identification attempt. The identifier rule used for this additional CI identification attempt.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.additionalCommittedItems.identificationAttempts.searchOnTable</td>
<td>The name of the table searched for this additional CI identification attempt.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.identifierEntrySysId</td>
<td>The sys_id for the identifier rule used to identify this additional CI. Notable values: • Unknown: Identification of this additional CI failed. See errors for details.</td>
</tr>
<tr>
<td>result.additionalCommittedItems.inputIndices</td>
<td>Array of index values for CIs from the request body items array that correspond to this additional CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.additionalCommittedItems.markers</td>
<td>Array of marker values for internal use.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>result.additionalCommittedItems.mergedPayloads</td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this additional CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>
### Name

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;mergedPayloadIds&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;String&quot;</td>
</tr>
</tbody>
</table>

#### result.additionalCommittedItems.operation

- **DELETE**: An existing CI is removed from the target table.
- **INSERT**: The additional CI is inserted into the target table as a new record.
- **NO_CHANGE**: No operation is performed for the additional CI.
- **UPDATE**: An existing CI in the target table is updated.
- **UPDATE_WITH_DOWNGRADE**: An existing CI in the target table is updated and its class is changed to a more generic class (ancestor class).
- **UPDATE_WITH_SWITCH**: An existing CI in the target table is updated and its class is changed to another class which is not an ancestor or descendent class.
- **UPDATE_WITH_UPGRADE**: An existing CI in the target table is updated and its class is changed to a more specialized class (descendent class).

#### Data type: String

#### The operation performed for this additional CI.

Possible values:

- **DELETE**: An existing CI is removed from the target table.
- **INSERT**: The additional CI is inserted into the target table as a new record.
- **NO_CHANGE**: No operation is performed for the additional CI.
- **UPDATE**: An existing CI in the target table is updated.
- **UPDATE_WITH_DOWNGRADE**: An existing CI in the target table is updated and its class is changed to a more generic class (ancestor class).
- **UPDATE_WITH_SWITCH**: An existing CI in the target table is updated and its class is changed to another class which is not an ancestor or descendent class.
- **UPDATE_WITH_UPGRADE**: An existing CI in the target table is updated and its class is changed to a more specialized class (descendent class).

#### result.additionalCommittedItems.sysId

- **Unknown**: Identification of this additional CI failed. See errors for details.

#### Data type: String

The sys_id found for this additional CI through identification.

Notable values:

- **Unknown**: Identification of this additional CI failed. See errors for details.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| result.additionalCommittedItems.warnings                  | Array of objects in which each object describes a warning encountered while processing this additional CI. | Data type: Array

```
"warnings": [
    {
        "error": "String",
        "message": "String"
    }
]
```

| result.additionalCommittedItems.warnings.error            | The type of warning encountered while processing this additional CI.         | Data type: String |
|-----------------------------------------------------------|-----------------------------------------------------------------------------|
| result.additionalCommittedItems.warnings.message          | The warning message encountered while processing this additional CI.         | Data type: String |
| result.additionalCommittedRelations                      | Array of objects in which each object describes a dependent relationship CI not included in the request body relations list to insert or update. | Data type: Array

```
"additionalCommittedRelations": [
    {
        "className": "String",
        "operation": "String",
        "markers": [],
        "inputIndices": [],
        "mergedPayloadIds": []
    }
]
```

<p>| result.additionalCommittedRelations.className            | The sys_class_name of this additional dependent relationship CI.             | Data type: String |
|-----------------------------------------------------------|-----------------------------------------------------------------------------|</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>result.additionalCommittedRelations.errorCount</strong></td>
<td>The number of errors encountered while processing this additional dependent CI. Data type: Number</td>
</tr>
<tr>
<td><strong>result.additionalCommittedRelations.errors</strong></td>
<td>Array of objects, each object describing an error encountered while processing this additional dependent CI. Data type: Array</td>
</tr>
<tr>
<td><strong>result.additionalCommittedRelations.errors.error</strong></td>
<td>The type of error encountered while processing this additional dependent CI. Data type: String</td>
</tr>
<tr>
<td><strong>result.additionalCommittedRelations.errors.message</strong></td>
<td>The error message encountered while processing this additional dependent CI. Data type: String</td>
</tr>
<tr>
<td><strong>result.additionalCommittedRelations.inputIndices</strong></td>
<td>Array of input index values for dependent relationship CI objects in the request body that correspond to this additional dependent relation. Data type: Array</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>result.additionalCommittedRelations.markers</code></td>
<td>Array of marker values for internal use.</td>
</tr>
<tr>
<td><code>result.additionalCommittedRelations.mergedPayloadIds</code></td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this additional dependent relationship CI.</td>
</tr>
</tbody>
</table>
| `result.additionalCommittedRelations.operation`                   | The operation performed for this additional dependent relationship CI. Possible values:  
  - INSERT: The dependent relationship CI is inserted into the target table.  
  - INSERT_AS_INCOMPLETE: The dependent relationship CI had errors and is inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.  
  - INSERT_AS_PARTIAL: The dependent relationship CI had errors and is inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table. |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items</td>
<td>Array of objects describing each CI included in the request body. Data type: Array.</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems</td>
<td>Array of objects describing each related CI (table lookup CI) from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table. Data type: Array.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.className</td>
<td>The sys_class_name of the related item. Data type: String</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.inputIndices</td>
<td>Array of index values for CIs from the request body items array that correspond to this related item. Data type: Array</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.markers</td>
<td>Array of marker values for internal use. Data type: Array</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.mergedPayloadIds</td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this related item. Data type: Array</td>
</tr>
<tr>
<td>result.items.additionalRelatedItems.sysId</td>
<td>The sys_id of this related CI. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.items.className</td>
<td>The sys_class_name of this CI.</td>
</tr>
<tr>
<td>result.items.duplicateLookupIndices</td>
<td>Object containing arrays of index values for items.lookup array elements that reference the same related CI. The array name indicates the first element from the request body items.lookup array that references a given related CI. Its values are the set of additional elements from the same array that reference the same related CI. Data type: Object. This example shows a case where the second element in the items.lookup array references the same related CI as a later element from that array:</td>
</tr>
<tr>
<td>result.items.errorCount</td>
<td>The number of errors encountered while processing this CI. Data type: Number.</td>
</tr>
<tr>
<td>result.items.errors</td>
<td>Array of objects in which each object describes an error encountered while processing this CI. Data type: Array.</td>
</tr>
</tbody>
</table>

```
"duplicateLookupIndices": [%2: [Array]
```

"errors": [
{
"error": "String",
"message": "String"
}
]
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.errors.error</td>
<td>The type of error encountered while processing this CI. Data type: String</td>
</tr>
<tr>
<td>result.items.errors.message</td>
<td>The error message encountered while processing this CI. Data type: String</td>
</tr>
<tr>
<td>result.items.identificationAttempts</td>
<td>Array of objects in which each object describes an attempt made to identify this CI. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"identificationAttempts": [
    {
      "attemptResult": "String",
      "attributes": [Array],
      "hybridEntryCiAttributes": [Array],
      "identifierName": "String",
      "searchOnTable": [Array]
    }
  ]
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.identificationAttempts.attemptResult</td>
<td>The outcome of this CI identification attempt. Possible values:</td>
</tr>
<tr>
<td>• MATCHED: Identification succeeded. A unique CI was found in the identifier rule table which exactly matched the specified attributes.</td>
<td></td>
</tr>
<tr>
<td>• MULTI_MATCH: Identification failed with an error. Duplicate CIs were found in the identifier rule table when matching against the specified attributes.</td>
<td></td>
</tr>
<tr>
<td>• NO_MATCH: Identification failed. No CI was found in the identifier rule table which matched the specified attributes.</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>• SKIPPED: Identification not attempted. The attributes required for this identifier rule table search were not provided, so the rule was not applied.</td>
<td></td>
</tr>
<tr>
<td>data_type: string</td>
<td></td>
</tr>
<tr>
<td>result.items.identificationAttempts.attributes</td>
<td>Array of CI identifier entry attributes used during this CI identification attempt.</td>
</tr>
<tr>
<td>data_type: array</td>
<td></td>
</tr>
<tr>
<td>Attribute names and types depend on the request body data and the identifier in use, such as:</td>
<td></td>
</tr>
<tr>
<td>&quot;attributes&quot;: [&quot;serial_number&quot;: &quot;string&quot;, &quot;serial_number_type&quot;: &quot;string&quot;]</td>
<td></td>
</tr>
<tr>
<td>result.items.identificationAttempts.hybridEntryCiAttributes</td>
<td>Array of CI identifier entry attributes used during this CI identification attempt.</td>
</tr>
<tr>
<td>data_type: array</td>
<td></td>
</tr>
<tr>
<td>Attribute names and types depend on the request body data and the identifier in use, such as:</td>
<td></td>
</tr>
<tr>
<td>&quot;hybridEntryCiAttributes&quot;: [&quot;name&quot;: &quot;string&quot;, &quot;serial_number&quot;: &quot;string&quot;]</td>
<td></td>
</tr>
<tr>
<td>result.items.identificationAttempts.identifierName</td>
<td>The identifier rule used for this CI identification attempt.</td>
</tr>
<tr>
<td>data_type: string</td>
<td></td>
</tr>
<tr>
<td>result.items.identificationAttempts.searchOnTable</td>
<td>The name of the table searched for this CI identification attempt.</td>
</tr>
<tr>
<td>data_type: string</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| result.items.identifierEntrySysId | The sys_id to identify this CI. Notable values: - Unknown: Identification of this CI failed. See errors for details.  
Data type: String |
| result.items.info | List of objects that contains additional information about the processing of the item.  
Data type: Array |
| result.items.info.code | Reclassification type that was skipped. Possible values: - SKIPPED_CLASS_SWITCH  
- SKIPPED_CLASS_DOWNGRADE  
- SKIPPED_CLASS_UPGRADE |
| result.items.info.message | Message that provides additional insights into the reason for skipping the reclassification.  
Data type: String |
<p>| result.items.info.ruleSysId | Sys_id of the reclassification restriction rule that was matched. Applicable when IRE skips reclassification due to reclassification restriction. This value is empty if the reclassification is skipped due to a payload or global flag. |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.inputIndices</td>
<td>Array of index values for CIs from the request body corresponding to this CI. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;inputIndices&quot;: [ Number ],</td>
</tr>
<tr>
<td>result.items.markers</td>
<td>Array of marker values for internal use. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;markers&quot;: [</td>
</tr>
<tr>
<td>result.items.mergedPayloadIds</td>
<td>Array of sys_id values for partial payloads from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this CI. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;mergedPayloadIds&quot;: [ String ],</td>
</tr>
<tr>
<td>result.items.operation</td>
<td>The operation performed for this CI. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• DELETE: An existing CI is removed from the target table.</td>
</tr>
<tr>
<td></td>
<td>• INSERT: The CI is inserted into the target table as a new record.</td>
</tr>
<tr>
<td></td>
<td>• NO_CHANGE: No operation is performed.</td>
</tr>
<tr>
<td></td>
<td>• UPDATE: An existing CI in the target table is updated.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.items.relatedItems.className</td>
<td>The sys_class_name of the related item.</td>
</tr>
<tr>
<td>result.items.relatedItems.inputIndices</td>
<td>Array of index values for CIs and lookup items from the request body items and items.lookup arrays that correspond to this related item.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>result.items.relatedItems.inputIndices.mainIndex</code></td>
<td>Index value from the request body <code>items</code> array that corresponds to the CI parent of the related item. Data type: Number</td>
</tr>
<tr>
<td><code>result.items.relatedItems.inputIndices.subIndex</code></td>
<td>Index value from the request body <code>items.lookup</code> array that corresponds to the related item. Data type: Number</td>
</tr>
<tr>
<td><code>result.items.relatedItems.markers</code></td>
<td>Array of marker values for internal use. Data type: Array</td>
</tr>
<tr>
<td><code>result.items.relatedItems.mergedPayloadIds</code></td>
<td>Array of sys_id values for entries in the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged into this CI during processing. Data type: Array</td>
</tr>
<tr>
<td><code>result.items.relatedSysIds</code></td>
<td>Array of sys_id values for related items (table lookup items) from the request body <code>items.lookup</code> array. Data type: Array</td>
</tr>
</tbody>
</table>

Notable values: 
- **null**: No sys_id was identified for this related item.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.items.sysId</td>
<td>The sys_id found for this CI through identification. Notable values:</td>
</tr>
<tr>
<td></td>
<td>• Unknown: Identification of this CI failed. See errors for details.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.items.warnings</td>
<td>Array of objects in which each object describes a warning encountered while</td>
</tr>
<tr>
<td></td>
<td>processing this CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.items.warnings.error</td>
<td>The type of warning encountered while processing this CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.items.warnings.message</td>
<td>The warning message encountered while processing this CI.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.logContextId</td>
<td>Context ID reported for this payload.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.relations</td>
<td>Array of objects in which each object describes a dependent relationship CI</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.relations.className</td>
<td>The sys_class_name of this dependent relationship CI. Only supported value: cmdb_rel_ci: The CI Relationship table.</td>
</tr>
<tr>
<td>result.relations.errorCount</td>
<td>The number of errors encountered while processing this dependent relationship CI.</td>
</tr>
<tr>
<td>result.relations.errors</td>
<td>Array of objects, each object describing an error encountered while processing this dependent relationship CI.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>result.relations.errors.error</code></td>
<td>The type of error encountered while processing this dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td><code>result.relations.errors.message</code></td>
<td>The error message encountered while processing this dependent relationship CI. Data type: String</td>
</tr>
<tr>
<td><code>result.relations.inputIndices</code></td>
<td>Array of index values for the dependent relationship CI objects in the request body that correspond to this dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td><code>result.relations.markers</code></td>
<td>Array of marker values for internal use. Data type: Array</td>
</tr>
<tr>
<td><code>result.relations.mergedPayloadIds</code></td>
<td>Array of system partial payload sys_id values from the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table that were merged during processing of this dependent relationship CI. Data type: Array</td>
</tr>
<tr>
<td><code>result.relations.operation</code></td>
<td>The operation performed for this dependent relationship CI. Possible values: © 2021 ServiceNow, Inc. All rights reserved.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>INSERT:</td>
<td>The dependent relationship CI was inserted into the target table as a new record.</td>
</tr>
<tr>
<td>INSERT_AS_INCOMPLETE:</td>
<td>The dependent relationship CI had errors and was inserted into the CMDB IRE Incomplete Payloads [cmdb_ire_incomplete_payloads] table.</td>
</tr>
<tr>
<td>INSERT_AS_PARTIAL:</td>
<td>The dependent relationship CI had errors and was inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.</td>
</tr>
<tr>
<td>NO_CHANGE:</td>
<td>No operation was performed for the dependent relationship CI.</td>
</tr>
<tr>
<td>UPDATE:</td>
<td>An existing dependent relationship CI in the target table was updated.</td>
</tr>
</tbody>
</table>

Data type: String
result.relations.warnings

Array of objects in which each object describes a warning encountered while processing this dependent relationship CI.

Data type: Array

```json
"warnings": [
  {
    "error": "String",
    "message": "String"
  }
]
```

result.relations.warnings.error

The type of warning encountered while processing this dependent relationship CI.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.relations.warnings.message</td>
<td>The warning message encountered while processing this dependent relationship.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

Simulate submission of a partial payload for a computer CI.

```bash
curl "https://instance.servicenow.com/api/now/identifyreconcile/queryEnhanced" \
  --request POST \n  --header "Accept: application/json" \n  --header "Content-Type: application/json" \n  --user "username":"password" \n  --data '{ \n    "items": [ \n      { \n        "className": "cmdb_ci_computer", \n        "sys_object_source_info": { \n          "source_name": "ServiceNow", \n          "source_native_key": "ncomputer3367" \n        }, \n        "values": { \n          "cpu_core_count": "6", \n          "ram": "8192", \n          "manufacturer": "DELL", \n          "company": "My Company" \n        } \n      } \n    ] \n  }'
```

The response shows that the payload would be inserted into the CMDB IRE Partial Payloads [cmdb_ire_partial_payloads] table.

```json
{
  "result": {
    "relations": [],
    "items": [
      {
        "identifierEntrySysId": "Unknown",
        "partialSysIds": [
```
"1aec79151bb45010593876a61a4bcb28",
"sysId": "Unknown",
"errorCount": 0,
"markers": [],
"className": "cmdb_ci_computer",
"inputIndices": [0],
"operation": "INSERT_AS_PARTIAL",
"errors": [],
"identificationAttempts": [
{
"info": "sys_object_source NO_MATCH",
"attemptResult": "NO_MATCH",
"identifierName": "",
"attributes": [],
"hybridEntryCiAttributes": []
},
{
"attemptResult": "SKIPPED",
"identifierName": "Hardware Rule",
"attributes": [
    "serial_number",
    "serial_number_type"
],
"hybridEntryCiAttributes": [],
"searchOnTable": "cmdb_serial_number"
},
{
"attemptResult": "SKIPPED",
"identifierName": "Hardware Rule",
"attributes": [
    "serial_number"
],
"hybridEntryCiAttributes": [],
"searchOnTable": "cmdb_ci_hardware"
},
{
"attemptResult": "SKIPPED",
"identifierName": "Hardware Rule",
"attributes": [
    "name"
]
}
Simulate submission of a second partial payload that would complete the description of the computer CI, with the summary generation Enhanced IRE option enabled.
The response shows that the computer CI would be inserted, and displays a summary of operations that would be performed.

```json
{
  "result": {
    "relations": [],
    "hasError": false,
    "additionalCommittedItems": [],
    "hasWarning": false,
    "items": [
      {
        "relatedSysIds": [
          "735c79151bb45010593876a61a4bcb67"
        ],
        "identifierEntrySysId": "Unknown",
        "sysId": "ff5c79151bb45010593876a61a4bcb64",
        "errorCount": 0,
        "markers": [],
        "className": "cmdb_ci_computer",
        "inputIndices": [
          0
        ],
        "operation": "INSERT",
        "relatedItems": [
          {
            "operation": "INSERT",
            "warningCount": 0,
            "errors": [],
            "sysId": "735c79151bb45010593876a61a4bcb67",
            "className": "cmdb_ci_network_adapter",
            "errorCount": 0,
            "markers": [],
            "inputIndices": []
          }
        ]
      }
    ]
  }
}
```
"subIndex": 0,
"mainIndex": 0
},
"mergedPayloadIds": [],
},
"mergedPayloadIds": [
"fe4c39151bb45010593876a4bcbd3"
],
"identificationAttempts": [
{
"info": "sys_object_source NO_MATCH",
"attemptResult": "NO_MATCH",
"identifierName": "",
"attributes": [],
"hybridEntryCiAttributes": []
},
{
"attemptResult": "SKIPPED",
"identifierName": "Hardware Rule",
"attributes": [
"serial_number",
"serial_number_type"
],
"hybridEntryCiAttributes": [],
"searchOnTable": "cmdb_serial_number"
},
{
"attemptResult": "SKIPPED",
"identifierName": "Hardware Rule",
"attributes": [
"serial_number"
],
"hybridEntryCiAttributes": [],
"searchOnTable": "cmdb_ci_hardware"
},
{
"attemptResult": "SKIPPED",
"identifierName": "Hardware Rule",
"attributes": [
"name"
],
"hybridEntryCiAttributes": []
}
"searchOnTable": "cmdb_ci_hardware"
},
{
"attemptResult": "NO_MATCH",
"identifierName": "Hardware Rule",
"attributes": [
  "mac_address",
  "name"
],
"hybridEntryCiAttributes": [],
"searchOnTable": "cmdb_ci_network_adapter"
},
"warningCount": 0
},
"additionalCommittedRelations": [],
"summary": {
"cmdb_ci_network_adapter": {
  "mergedPartialPayloads": 0,
  "inserted": 1,
  "partial": 0,
  "warnings": 0,
  "incomplete": 0,
  "additionalInserted": 0,
  "unchanged": 0,
  "skipped": 0,
  "updated": 0,
  "errors": 0
},
"cmdb_ci_computer": {
  "mergedPartialPayloads": 1,
  "inserted": 1,
  "partial": 0,
  "warnings": 0,
  "incomplete": 0,
  "additionalInserted": 0,
  "unchanged": 0,
  "skipped": 0,
  "updated": 0,
  "errors": 0
}
}
Example: Python request

Simulate submission of a payload containing entries for three Linux server CIs and generate a summary of operations performed for this payload.

```python
# Need to install requests package for python
import requests

# Set the API endpoint URL for the request
data_url = 'https://instance.servicenow.com/api/now/identifyreconcile/queryEnhanced'

# Set credentials
data_user = 'username'
data_pwd = 'password'

# Set query parameters
query_params = {
    "options": "generate_summary:true"
}

# Set HTTP headers
headers = {
    "Accept": "application/json",
    "Content-Type": "application/json"
}

# Define JSON payload
payload = {
    "items": [
        {
            "className": "cmdb_ci_linux_server",
            "sys_object_source_info": {
                "source_name": "ServiceNow",
                "native_source_key": "linserver273"
            },
            "values": {
                "name": "linuxserver9423",
                "serial_number": "m51ced54747j89ed561n2bn674sa",
                "ram": "16384",
                "ip_address": "92.251.144.62",
                "company": "b7e831bd0a80169015ae101f3c4d6cd",
                "discovery_source": "ServiceNow"
            }
        }
    ]
}
```

# Issue the HTTP request
response = requests.post(url, auth=(user, pwd), params=params, headers=headers,
                        json=payload)

# Check for HTTP status codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.content)
    exit()
All three CIs would be updated, and the operation summary is generated.

```json
{
    "result": {
        "items": [
            {
                "className": "cmdb_ci_linux_server",
                "operation": "UPDATE",
                "sysId": "5472d2d1bb85010593876a61a4bcbe6",
                "identifierEntrySysId": "fb27f69cc3000200d8d4bea192d3ae67",
                "identificationAttempts": [
                    {
                        "info": "sys_object_source SKIPPED",
                        "identifierName": "",
                        "attemptResult": "SKIPPED",
                        "attributes": [],
                        "hybridEntryCiAttributes": []
                    },
                    {
                        "identifierName": "Hardware Rule",
                        "attemptResult": "SKIPPED",
                        "attributes": [
                            "serial_number",
                            "serial_number_type"
                        ],
                        "searchOnTable": "cmdb_serial_number",
                        "hybridEntryCiAttributes": []
                    },
                    {
                        "identifierName": "Hardware Rule",
                        "attemptResult": "MATCHED",
                        "attributes": [
                            "serial_number"
                        ],
                        "searchOnTable": "cmdb_ci_hardware",
                        "hybridEntryCiAttributes": []
                    }
                ],
                "errorCount": 0,
            }]
        }
    }
}``
"markers": [],
"warningCount": 0,
"mergedPayloadIds": [],
"inputIndices": [
  0
],

"
"className": "cmdb_ci_linux_server",
"operation": "UPDATE",
"sysId": "9472d2d1bb85010593876a61a4bcbee",
"identifierEntrySysId": "f927f69cc3000200d8d4bea192d3ae67",
"identificationAttempts": [
  {
    "info": "sys_object_source SKIPPED",
    "identifierName": "",
    "attemptResult": "SKIPPED",
    "attributes": [],
    "hybridEntryCiAttributes": []
  },
  {
    "identifierName": "Hardware Rule",
    "attemptResult": "SKIPPED",
    "attributes": [
      "serial_number",
      "serial_number_type"
    ],
    "searchOnTable": "cmdb_serial_number",
    "hybridEntryCiAttributes": []
  },
  {
    "identifierName": "Hardware Rule",
    "attemptResult": "MATCHED",
    "attributes": [
      "serial_number"
    ],
    "searchOnTable": "cmdb_ci_hardware",
    "hybridEntryCiAttributes": []
  }
],
"errorCount": 0,
"markers": [],
"warningCount": 0,
"mergedPayloadIds": []

Import Set API

The Import Set API allows you to interact with import set tables.

Overview

The API transforms incoming data based on associated transform maps. The import set API supports synchronous transforms. The Import Set API mirrors the existing SOAP interface.

Security

Access to tables via the REST API is restricted by BasicAuth. To allow access to tables without any authentication or authorization, add the table name to sys_public.list. ACLs defined on tables are still enforced, and it is the administrator's responsibility to deactivate ACLs.

Import Set - GET /now/import/{stagingTableName}/{sys_id}

Retrieves the specified import staging record and resulting transformation result.
**URL format**

Versioned URL: `/api/now/{api_version}/import/{stagingTableName}/{sys_id}`

Default URL: `/api/now/import/{stagingTableName}/{sys_id}`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>stagingTableName</td>
<td>Name of the table from which to obtain the import data. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the record that contains the data. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Indicates the specified resource was not available. As import set tables are deleted frequently based on a schedule, GET requests may return 404 NotFound responses if the transformation result no longer exists.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>import_set</td>
<td>Name of the import set. Data type: String</td>
</tr>
<tr>
<td>result</td>
<td>List of objects that contain information on the data sets that were imported.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Data type: Array</td>
<td>&quot;result&quot;: [</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>result.display_name</td>
<td>Display name of the import set.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.display_value</td>
<td>Value of the import set.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.record_link</td>
<td>Table API GET request for the imported record.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.status</td>
<td>Status of the import.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.sys_id</td>
<td>Sys_id of the import record.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.table</td>
<td>Name of the table in which the data was imported.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.transform_map</td>
<td>Name of the transform map.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>staging_table</td>
<td>Name of the import staging table.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

```
curl
"https://instance.servicenow.com/api/now/import/imp_user/e2928be64f411200adf9f8e18110c777"
```
Example: Sample Python request

```python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/import/imp_user/e2928be64f411200adf9f8e18110c777'

# Eg. User name='username', Password='password' for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept': 'application/xml'}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    # Handle error
```
print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<?xml version="1.0" encoding="UTF-8"?>
<response>
  <import_set>ISET0010001</import_set>
  <staging_table>imp_user</staging_table>
  <result>
    <display_name>name</display_name>
    <display_value>John Public</display_value>
    <status>inserted</status>
    <sys_id>ea928be64f411200adf9e18110c777</sys_id>
    <record_link>https://instance.service-now.com/api/now/table/sys_user/ea928be64f411200adf9e18110c777</record_link>
    <table>sys_user</table>
    <transform_map>User</transform_map>
  </result>
</response>

**Import Set - POST /now/import/{stagingTableName}**

Inserts incoming data into a specified staging table and triggers transformation based on predefined transform maps in the import set table.

Transformation occurs synchronously. For each transform map that you define, the responses include transformation results such as information on the target records.

⚠️ **Note:** The status_message and error_message fields on transformation scripts are processed and returned in response, along with any custom response fields.

**URL format**

Versioned URL: /api/now/{api_version}/import/{stagingTableName}
Default URL: /api/now/import/{stagingTableName}
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>stagingTableName</td>
<td>Name of the table from which to import the data. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call specific</td>
<td>Name-value pairs to insert in the import fields. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>import_set</td>
<td>Name of the import set. Data type: String</td>
</tr>
<tr>
<td>result</td>
<td>List of objects that contain information on the data sets that were imported. Data type: Array</td>
</tr>
</tbody>
</table>

```
"result": [  
  
  "display_name": "String",  
  "display_value": "String",  
  "record_link": "String",  
  "status": "String",  
  "sys_id": "String",  
  "table": "String",  
  "transform_map": "String"
```

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.display_name</td>
<td>Display name of the import set.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.display_value</td>
<td>Value of the import set.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.record_link</td>
<td>Table API GET request for the imported record.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.status</td>
<td>Status of the import.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.sys_id</td>
<td>Sys_id of the import record.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.table</td>
<td>Name of the table in which the data was imported.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.transform_map</td>
<td>Name of the transform map.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>staging_table</td>
<td>Name of the import staging table.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

```bash
curl "https://instance.servicenow.com/api/now/import/imp_user" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --data "{"first_name":"John","last_name":"Public","user_id":"john.public","email":"john.public@company.com"}" \ 
  --user "username":"password"
```

```json
{
  "import_set": "ISET0010001",
  "staging_table": "imp_user",
  "result": [ 
```
Example: Sample Python request

```python
# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.service-now.com/api/now/import/imp_user'
user = 'username'
pwd = 'password'
headers = {'Content-Type': 'application/xml', 'Accept': 'application/xml'}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers,
                          data='<request><entry><first_name>John</first_name><last_name>Public</last_name><user_id>john.public</user_id><email>john.public@company.com</email></entry></request>"

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```
<?xml version="1.0" encoding="UTF-8"?>
<response>
    <import_set>ISET0010001</import_set>
</response>
```
Import Set - POST /now/import/{stagingTableName}/insertMultiple

Inserts multiple records into a specified staging table and triggers transformation based on predefined transform maps or Robust Transform Engine (RTE) configurations in a single request.

Transformation is asynchronous by default. To set synchronous transformation, create a new record in the Rest Insert Multiples [sys_rest_insert_multiple] table, select the source table, and set the transformation to synchronous.

This endpoint can send a request body in two possible formats.

**Data source file format**

If you generate a staging table from a JSON data source, match the JSON format of the source file.

**Staging table column format**

Default. Matches the staging table column request body format in key value pairs.

**URL format**

Versioned URL: /api/now/{api_version}/import/{stagingTableName}/insertMultiple

Default URL: /api/now/import/{stagingTableName}/insertMultiple

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>stagingTableName</td>
<td>Name of the import set table from which to import the data. Refer to Import sets key concepts. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>multi_import_set_id</td>
<td>Sys_id of an entry in the Multi Import Sets [sys_multi_import_set] table. If specified, adds the current import to this multiple import set instead of adding to a new multiple import set. Data type: String</td>
</tr>
<tr>
<td>run_after</td>
<td>Sys_id of an entry in the Import Sets [sys_import_set] table. Enables running the current import set after the specified import set is complete. You can use this parameter to enforce the sequential order of imports. This parameter is only valid in asynchronous transformations. Data type: String</td>
</tr>
</tbody>
</table>

### Request body (JSON)

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data source file</td>
<td>This request body format matches the JSON file format used to create the data source. Provide the request body in the same format as the JSON in the data source. JSON input varies depending on the properties in your data source. Refer to JSON information in File type data source.</td>
</tr>
</tbody>
</table>
Request body (JSON) (continued)

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• This option is only available if the staging table was created using a JSON data source. Refer to <a href="#">Create a File type data source</a>.</td>
<td></td>
</tr>
<tr>
<td>• You must define the path of the data source in the Data Source [sys_data_source] table in the Path for each row field.</td>
<td></td>
</tr>
<tr>
<td>• To change the default behavior for the REST Insert Multiple user, create an entry in the REST Insert Multiples [sys_rest_insert_multiple] table.</td>
<td></td>
</tr>
<tr>
<td>• Enable the <strong>Use data source format</strong> in the REST insert multiple entry.</td>
<td></td>
</tr>
</tbody>
</table>

**Data type:** Object

**Staging table column (default)**

This request body format matches staging table columns. Use the `records` array of key-value pairs matching the staging table column to insert in the import fields. Each JSON key is maps the table column to a JSON value representing the value to be inserted. JSON input varies depending on what fields are in your staging table.

The column mapping default key value is to the column table.

```
{
  "records": [  
    {  
      "<ColumnLabel1>":"<value>",
      "<ColumnLabel2>":"<value>"
    },  
    {  
      "<ColumnLabel1>":"<value>",
      "<ColumnLabel2>":"<value>"
    }  
  ]
}
```

You can modify mapping settings by adding an entry in the Rest Insert Multiples [sys_rest_insert_multiple] table and changing the **Column mapping** from Label to Column name.

```
{
  "records": [  
    {  
      "<column_name1>":"<value>",
      "<column_name2>":"<value>"
    }  
  ]
}
```
Request body (JSON) (continued)

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;&lt;column_name1&gt;&quot;:&quot;&lt;value&gt;&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;&lt;column_name2&gt;&quot;:&quot;&lt;value&gt;&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

The data dictionary provides details on table fields in the system.

Data type: Array

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>import_set_id</td>
<td>Sys_id of the record added to the Import Sets [sys_import_set] table. For asynchronous requests, you can use this value to run another import set after this import set process is complete. Data type: String</td>
</tr>
<tr>
<td>multi_import_set_id</td>
<td>Sys_id of the record added to the Multi Import Sets [sys_multi_import_set] table. This value can be used to group multiple import sets into one set. Data type: String</td>
</tr>
</tbody>
</table>

### Example: Sample cURL request

The following example shows how to run a transform on an import table called `u_employee_import_set_table` using the staging table column format.

```cURL
curl "https://instance.servicenow.com/api/now/import/u_employee_import_set_table/insertMultiple" \
--request POST \
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--data "{
  "records": [
    {
      "Address": "Hollywood",
      "Name": "Tom",
      "ID": "123"
    }
  ]
"}
```
Results include sys_ids for new records in the Import Sets [sys_import_set] and Multi Import Sets [sys_multi_import_set] tables.

```
}

"import_set_id": "<import_set_sys_id>",
"multi_import_set_id": "<multi_import_set_sys_id>"
```  

**Interaction Management API**

Use the interaction management API to create interactions.

This class requires the Interaction Logging, Routing, and Queueing plugin (com.glide.interaction).

**Interaction - POST /now/interaction**

Inserts or updates a record in the interaction table.

**URL format**

- Versioned URL: /api/now/{api_version}/interaction
- Default URL: /api/now/interaction

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>
### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel</td>
<td>Required. Sys_id for the interaction connector you want to create an interaction for. Data type: String</td>
</tr>
<tr>
<td>context</td>
<td>A Map or a table/sys_id pair. By default, interactions have a record in the interaction_json_blob table that stores context data as a JSON object. You can point an interaction to a different record for context. Use context or context_table/context_id to define the context for an interaction. Data type: String</td>
</tr>
<tr>
<td>context_id</td>
<td>Document ID for a record you want to use to store context. The default is a new record in the interaction_json_blob table. If you define the context ID, you also need to define the table. Data type: String</td>
</tr>
<tr>
<td>context_table</td>
<td>Context table you want to use to store interaction context for. The default is interaction_json_blob. If you define the context table, you also need to define the context ID. Data type: String</td>
</tr>
<tr>
<td>channel_metadata</td>
<td>Map or a table/sys_id pair. Channel metadata contains information that a connector can use to communicate through the associated channel. Data type: String</td>
</tr>
<tr>
<td>channel_metadata_id</td>
<td>Document ID for a record you want to use to store channel metadata. The default is a new record in the live_group_profile table. If you define the</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel_metadata_ID</td>
<td>channel metadata ID, you also need to define the table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>channel_metadata_table</td>
<td>Table you want to use to store channel metadata. The default is the live_group_profile table. If you define the channel metadata table, you also need to define the channel metadata ID.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>queue</td>
<td>Sys_id for the queue you want to associate with the interaction.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json Or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json Or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Indicates the query ran successfully.</td>
</tr>
<tr>
<td>400</td>
<td>Indicates that one or more mandatory parameters were missing from the request.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assigned_to</td>
<td>The sys_id of the user the interaction is assigned to. Data type: String</td>
</tr>
<tr>
<td>channelMetadataDocument</td>
<td>The sys_id of the channel metadata document associated with the interaction. Data type: String</td>
</tr>
<tr>
<td>channelMetadataTable</td>
<td>The table name of the channel metadata table associated with the interaction. Data type: String</td>
</tr>
<tr>
<td>channel_metadata</td>
<td>Any channel metadata included for the interaction. Data type: String</td>
</tr>
<tr>
<td>closed_by</td>
<td>The sys_id of the user who closed the interaction. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>contextDocument</td>
<td>The sys_id for the context document associated with the interaction. Data type: String</td>
</tr>
<tr>
<td>contextTable</td>
<td>The name of the table associated with the interaction. Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>The number of the new interaction. Data type: String</td>
</tr>
<tr>
<td>opened_for</td>
<td>The sys_id of the user who the interaction was opened for. Data type: String</td>
</tr>
<tr>
<td>queue</td>
<td>The sys_id for the queue you create the interaction for. Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>The state the interaction is in. If you assign a queue, the state automatically changes to <strong>Queued</strong>, otherwise the default is <strong>New</strong>. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>The sys_id of the interaction. Data type: String</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

```bash
curl "http://instance.servicenow.com/api/now/interaction" \
   --request POST "
   --header "Accept:application/json" "
   --header "Content-Type:application/json" "
   --data "{"channel":"28a5989387310302ae97e2526cb0b5d","queue":"f3a50867b30303002186a72256a8dcb7"}" "
   --user "username":"password"
```

```json
{
   "result": {
      "channel": "28a5989387310302ae97e2526cb0b5d",
      "context": {
```
Example: Sample Python request

```python
import requests

# Set the request parameters
url = 'http://instance.servicenow.com/api/now/interaction'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {"Content-Type":"application/json","Accept":"application/json"}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers,
data="{"channel":"28a59893873103002ae97e2526cb0b5d","queue":"f3a50867b3030302186a72256a8dcb7"}")

# Check for HTTP codes other than 200
if response.status_code != 200:
    exit()
```
Interaction - POST /now/interaction/{interaction_id}/close
Changes the state of an interaction to closed or closed complete.

URL format
Versioned URL: /api/now/{api_version}interaction/{interaction_id}/close
Default URL: /api/now/interaction/{interaction_id}/close
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>interaction_id</td>
<td>Sys_id for the interaction you want to close. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assigned_to</td>
<td>The sys_id of the user the interaction is assigned to. Data type: String</td>
</tr>
<tr>
<td>channelMetadataDocument</td>
<td>The sys_id of the channel metadata document associated with the interaction. Data type: String</td>
</tr>
<tr>
<td>channelMetadataTable</td>
<td>The table name of the channel metadata table associated with the interaction. Data type: String</td>
</tr>
<tr>
<td>channel_metadata</td>
<td>Any channel metadata included for the interaction. Data type: String</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>closed_by</td>
<td>The sys_id of the user who closed the interaction.</td>
</tr>
<tr>
<td>contextDocument</td>
<td>The sys_id for the context document associated with the interaction.</td>
</tr>
<tr>
<td>contextTable</td>
<td>The name of the table associated with the interaction.</td>
</tr>
<tr>
<td>number</td>
<td>The number of the new interaction.</td>
</tr>
<tr>
<td>opened_for</td>
<td>The sys_id of the user who the interaction was opened for.</td>
</tr>
<tr>
<td>queue</td>
<td>The sys_id for the queue you create the interaction for.</td>
</tr>
<tr>
<td>state</td>
<td>The state the interaction is in. If you assign a queue, the state automatically changes to <strong>Queued</strong>, otherwise the default is <strong>New</strong>.</td>
</tr>
<tr>
<td>sys_id</td>
<td>The sys_id of the interaction.</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

```
curl "http://instance.servicenow.com/api/now/interaction/b243cde4208f1700964f959e0488dee8/close" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --user "username":"password"

{
  "result": {
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: Sample Python request

```python
# Need to install requests package for python
import requests

# Set the request parameters
url = 'http://instance.servicenow.com/api/now/interaction/b243cde4208f1700964f959e0488dee8/close'

# Eg. User name="username", Password="password" for this code sample.
user = "username"
pwd = "password"

# Set proper headers
headers = {"Content-Type":"application/json","Accept":"application/json"}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    pass
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Knowledge Management REST API

The Knowledge Management REST API enables searching, viewing, and fetching lists of most-viewed and featured knowledge articles.

This API can only be used when the Knowledge API (sn_km_api) plugin is activated. The Knowledge Management REST API was originally released in Orlando using the Knowledge API app available in the ServiceNow Store.

To enable other domains to use Knowledge Management REST API endpoints, define a Cross-Origin Resource Sharing (CORS) rule. For more information, see Define a CORS rule.

To view an article from the scoped knowledge base using this REST API, allow the sn_km_api scope read access from the requesting scope in the Restricted Caller
Access Privileges [sys_restricted_caller_access] table. For more information, see Define cross-scope access to an application resource.

By default, this API has a rate limit of 500 per hour for unauthenticated and snc_external users. For more information about rate limiting, see Inbound REST API rate limiting.

**Knowledge Management - GET /knowledge/articles**

Returns a list of knowledge base (KB) articles which can be searched and filtered using various parameters.

**URL format**

Versioned URL: `/api/sn_km_api/{api_version}/knowledge/articles`

Default URL: `/api/sn_km_api/knowledge/articles`

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| filter | Encoded query to use to filter the result set. Syntax: `filter=<attr><operator><value>`.

- `<attr>`: Name of the table column.
- `<operator>`: Valid values:
  - `=`: Exactly matches `<value>`.
  - `!`: Does not match `<value>`.
  - `∧`: Enables you to specify more than one condition and logically AND them.
**Query parameters (continued)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>^OR:</td>
<td>Enables you to specify more than one condition and logically OR them.</td>
</tr>
<tr>
<td>LIKE:</td>
<td>&lt;attr&gt; contains the specified string. Only works for &lt;attr&gt; fields whose data type is string.</td>
</tr>
<tr>
<td>STARTSWITH:</td>
<td>&lt;attr&gt; starts with the specified string. Only works for &lt;attr&gt; fields whose data type is string.</td>
</tr>
<tr>
<td>ENDSWITH:</td>
<td>&lt;attr&gt; ends with the specified string. Only works for &lt;attr&gt; fields whose data type is string.</td>
</tr>
<tr>
<td>&lt;value&gt;:</td>
<td>Value to match against.</td>
</tr>
</tbody>
</table>

All parameters are case-sensitive. Query can contain more than one entry, such as `filter=<attr><operator><value>[<operator><attr><operator><value>]`.  

**Data type:** String  
**Default:** empty

| fields | Comma-separated list of fields from the Knowledge [kb_knowledge] table to show details in results.  
**Data type:** String  
**Default:** None

| kb | Comma-separated list of knowledge base sys_ids from the Knowledge Bases [kb_knowledge_base] table to restrict results to.  
**Data type:** String

| language | List of comma-separated languages in two-letter ISO 639-1 language code format to restrict results to. Alternatively type 'all' to search in all valid installed languages on an instance.  
**Data type:** String  
**Default:** User's session language or en

| limit | Maximum number of records to return. Unusually large limit values can impact system performance. For requests that exceed this number of records, use the offset parameter to paginate record retrieval.  
**Data type:** Number
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks. For example, the first time this endpoint is called, offset is set to &quot;0&quot;. To page through all available records, use offset=offset+limit until the end of the all records is reached. Data type: Number Default: 0</td>
</tr>
<tr>
<td>query</td>
<td>Text to search for, can be empty. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
## Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>articles</td>
<td>List of articles returned in response. Data type: Array</td>
</tr>
<tr>
<td>articles.fields</td>
<td>Values of requested fields, if any. Data type: Object</td>
</tr>
</tbody>
</table>

```json
"articles": [
  {
    "fields": {Object},
    "link": "String",
    "id": "String",
    "number": "String",
    "rank": Number,
    "score": Number,
    "snippet": "String",
    "title": "String"
  }
]
```

```json
"fields": {
  "<field_name>": {Object}
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>articles.fields.&lt;field_name&gt;</td>
<td>Lists each field requested using the fields parameter, if any. Data type: Object</td>
</tr>
</tbody>
</table>
|   | ```
   "<field_name>": {
   "display_value": "String",
   "label": "String",
   "name": "String",
   "type": "String",
   "value": "String"
   }
   ``` |
<p>| articles.fields.&lt;field_name&gt;.display_value | Display value of the requested field. Data type: String |
| articles.fields.&lt;field_name&gt;.label | Label representing the requested field. For example, Knowledge. Data type: String |
| articles.fields.&lt;field_name&gt;.name | Name of the requested field. Matches <code>&lt;field_name&gt;</code>. Data type: String |
| articles.fields.&lt;field_name&gt;.type | Data type of requested field. Data type: String |
| articles.fields.&lt;field_name&gt;.value | Value of the requested field. Data type: String |
| articles.id | Knowledge article sys_id from the Knowledge [kb_knowledge] table. Data type: String |
| articles.link | Link to the article. Data type: String |
| articles.number | Knowledge article number. Data type: String |
| articles.rank | Search rank of article specific to this search. Data type: Number (Float) |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>articles.snippet</td>
<td>Text showing a small portion of the knowledge article.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>articles.score</td>
<td>Relevancy score, results sorted in descending order by score.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>articles.title</td>
<td>Short description or title of the knowledge article.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>meta</td>
<td>Meta information of the results and request parameters.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>&quot;meta&quot;: {</td>
<td></td>
</tr>
<tr>
<td>&quot;count&quot;: Number,</td>
<td></td>
</tr>
<tr>
<td>&quot;end&quot;: Number,</td>
<td></td>
</tr>
<tr>
<td>&quot;fields&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;filter&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;kb&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;language&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;query&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;start&quot;: Number,</td>
<td></td>
</tr>
<tr>
<td>&quot;status&quot;: {Object},</td>
<td></td>
</tr>
<tr>
<td>&quot;ts_query_id&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>meta.count</td>
<td>Number of available KB articles.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>meta.end</td>
<td>Ending index of the result set.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>meta.fields</td>
<td>Fields in the article.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>meta.filter</td>
<td>Filter used to acquire the data.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>meta.kb</td>
<td>List of knowledge base article sys_ids.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>meta.language</td>
<td>List of comma-separated languages of the KB articles that were requested.</td>
</tr>
<tr>
<td>meta.query</td>
<td>Specified request query.</td>
</tr>
<tr>
<td>meta.start</td>
<td>Starting index of result set.</td>
</tr>
<tr>
<td>meta.status</td>
<td>Status of the call.</td>
</tr>
<tr>
<td>meta.ts_query_id</td>
<td>Sys_id of the query.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl 
  "https://instance.servicenow.com/api/sn_km_api/knowledge/articles?query=Windows&limit=2&fields=short_description&fields=sys_class_name" 
  --request GET 
  --header "Accept:application/xml" 
  --user "username":"password"
```

```json
{
  "result": {
    "meta": {
      "start": 0,
      "end": 2,
      "fields": "short_description,sys_class_name",
      "query": "Windows",
      "filter": ",",
      "kb": ",",
      "language": "en",
      "count": 19,
      "ts_query_id": "7976f36129c30410f877796e70786991",
      "status": {
        "code": 200
      }
    }
  }
}
```
Should I upgrade to Windows 8.x?

Windows 8.x is designed for using touch, mouse, and keyboard the Windows Store and access apps such as Calendar, Mail, and Messaging. By most accounts, Windows boot times, smaller memory footprint, and more free memory for the programs you run.

What is the Windows key?

The Windows key is a standard key on most keyboards on computers built to use a Windows operating system. It is labeled with a Windows logo, and is usually placed between on the right side as well. Pressing Win (the Windows key) on its own will do the following:

Windows: Toggle
Knowledge Management - GET /knowledge/articles/{article_sys_id}/attachments/{attachment_sys_id}

Returns a knowledge article attachment as a file.

**URL format**

**Versioned URL:** `/api/sn_km_api/{api_version}/knowledge/articles/{article_sys_id}/attachments/{attachment_sys_id}`

**Default URL:** `/api/sn_km_api/knowledge/articles/{article_sys_id}/attachments/{attachment_sys_id}`

**Supported request parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>article_sys_id</td>
<td>Sys_id of the knowledge article with the attachment you intend to retrieve. Located in the Knowledge Bases [kb_knowledge] table.</td>
</tr>
<tr>
<td>attachment_sys_id</td>
<td>Sys_id of record to which the attachment belongs.</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>The content type of the response, for example, image/gif or <em>/</em>.</td>
</tr>
</tbody>
</table>
**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

**Response body parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File is returned as a response.</td>
<td></td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

```bash
curl "https://instance.service-now.com/api/sn_km_api/knowledge/articles/0b48fd75474321009db4b5b08b9a71c2/attachments/fedf5614294f4010f877796e70786956" \
--request GET \
--header "Accept:*/*" \
--user "username":"password"
```

Binary response not shown (file is returned as a response).

**Knowledge Management - GET /knowledge/articles/featured**

Returns a list of the most-viewed knowledge articles and featured knowledge articles.

**URL format**

*Versioned URL*: `/api/sn_km_api/{api_version}/knowledge/articles/featured`

*Default URL*: `/api/sn_km_api/knowledge/articles/featured`
### Supported request parameters

#### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

#### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>Comma-separated list of fields from the Knowledge [kb_knowledge] table to show details in results.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: None</td>
</tr>
<tr>
<td>kb</td>
<td>Comma-separated list of knowledge base sys_ids from the Knowledge Bases [kb_knowledge_base] table to restrict results to.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: None</td>
</tr>
<tr>
<td>language</td>
<td>List of comma-separated languages in two-letter ISO 639-1 language code format to restrict results to. Alternatively type 'all' to search in all valid installed languages on an instance.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: User’s session language or en</td>
</tr>
<tr>
<td>limit</td>
<td>Maximum number of records to return. Unusually large limit values can impact system performance. For requests that exceed this number of records, use the offset parameter to paginate record retrieval.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Default: 30</td>
</tr>
<tr>
<td>offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks.</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>For example, the first time this endpoint is called, <code>offset</code> is set to &quot;0&quot;. To page through all available records, use <code>offset=offset+limit</code> until the end of the all records is reached.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Default: 0</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| articles        | List of articles returned in response. Data type: Array  
"articles":[
  {
    "fields": {Object},
    "id": "String",
    "link": "String",
    "number": "String",
    "rank": Number,
    "score": Number,
    "snippet": "String",
    "title": "String"
  }
] |
| articles.fields | Values of requested fields, if any. Data type: Object  
"fields": {
"<field_name>": {Object}
} |
<p>| articles.fields.&lt;field_name&gt; | Lists each field requested using the fields parameter, if any. Data type: Object |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>articles.fields.&lt;field_name&gt;.display_value</td>
<td>Display value of the requested field.</td>
</tr>
<tr>
<td>articles.fields.&lt;field_name&gt;.name</td>
<td>Name of the requested field.</td>
</tr>
<tr>
<td>articles.fields.&lt;field_name&gt;.label</td>
<td>Label representing the requested field. For example, Knowledge.</td>
</tr>
<tr>
<td>articles.fields.&lt;field_name&gt;.type</td>
<td>Data type of requested field.</td>
</tr>
<tr>
<td>articles.fields.&lt;field_name&gt;.value</td>
<td>Value of the requested field.</td>
</tr>
<tr>
<td>articles.id</td>
<td>Knowledge article sys_id from the Knowledge [kb_knowledge] table.</td>
</tr>
<tr>
<td>articles.link</td>
<td>Link to the article.</td>
</tr>
<tr>
<td>articles.number</td>
<td>Knowledge article number.</td>
</tr>
<tr>
<td>articles.rank</td>
<td>Search rank of article specific to this search.</td>
</tr>
<tr>
<td>articles.score</td>
<td>Relevancy score, results sorted in descending order by score.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>articles.snippet</td>
<td>Text showing a small portion of the knowledge article. Data type: String</td>
</tr>
<tr>
<td>articles.title</td>
<td>Short description or title of the knowledge article. Data type: String</td>
</tr>
<tr>
<td>meta</td>
<td>Meta information of the results and request parameters. Data type: Object</td>
</tr>
<tr>
<td>meta.count</td>
<td>Number of available KB articles. Data type: Number</td>
</tr>
<tr>
<td>meta.end</td>
<td>Ending index of the result set. Data type: Number</td>
</tr>
<tr>
<td>meta.fields</td>
<td>Fields in the article. Data type: String</td>
</tr>
<tr>
<td>meta.filter</td>
<td>Filter used to acquire the data. Data type: String</td>
</tr>
<tr>
<td>meta.kb</td>
<td>List of knowledge base article sys_ids. Data type: String</td>
</tr>
</tbody>
</table>

```javascript
"meta": {
    "count": Number,
    "end": Number,
    "fields": "String",
    "filter": "String",
    "kb": "String",
    "language": "String",
    "query": "String",
    "start": Number,
    "status": {Object},
    "ts_query_id": "String"
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meta.language</td>
<td>List of comma-separated languages of the KB articles that were requested.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>meta.query</td>
<td>Specified request query.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>meta.start</td>
<td>Starting index of result set.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>meta.status</td>
<td>HTTP status of the call.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>meta.ts_query_id</td>
<td>Sys_id of the query.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
  "https://instance.servicenow.com/api/sn_km_api/knowledge/articles/featured?fields=short_description&limit=3" \
  --request GET \
  --header "Accept:application/json" \
  --user "username":"password"

{
  "result": {
    "meta": {
      "start": 0,
      "end": 3,
      "fields": "short_description",
      "query": "homepage",
      "filter": "",
      "kb": "",
      "language": "en",
      "status": {
        "code": 200
      },
      "count": 2
    },
    "articles": []
  }
```
If the site is UP but you can't access the page, try one of the below solutions: Browser Related Problems Force a full refresh for the site. This can be achieved by pressing CTRL + F5 keys at the same time on your favourite browser (Firefox, Chrome, Explorer, etc.) Try alternative urls such as m.outlook.com Clear the temporary cache and cookies ,

On Friday, January 20th, we experienced a widespread outage that affected all Zoho services. The outage started around 8:13 am Pacific Time. Zoho services started coming back online for customer use at 3:49 pm, and all services were fully restored at 6:22 pm PST. We absolutely realize how important our services are for businesses and users who,
Knowledge Management - GET /knowledge/articles/{id}
Returns specific knowledge article content and its field values.

**URL format**

Versioned URL: /api/sn_km_api/{api_version}/knowledge/articles/{id}
Default URL: /api/sn_km_api/knowledge/articles/{id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>Sys_id or knowledge base (KB) number of a knowledge article in the Knowledge [kb_knowledge] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>Comma-separated list of fields from the Knowledge [kb_knowledge] table to show details in results.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: None</td>
</tr>
<tr>
<td>language</td>
<td>Two letter ISO 639-1 language code; for example, &quot;fr&quot; for French. Results display only when searches use the knowledge article KB number as the id and a translated version of the article is available in the language specified.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>✪ <strong>Note:</strong> Only valid when setting the id parameter as a KB number (not sys_id).</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>search_id</td>
<td>Optional unless using search_rank. Unique identifier of search that returned this article. You can retrieve search_id using one of the following APIs that returns the articles.id element:</td>
</tr>
</tbody>
</table>
|            | • Knowledge Management - GET /knowledge/articles  
|            | • Knowledge Management - GET /knowledge/articles/featured  
|            | • Knowledge Management - GET knowledge/articles/most_viewed  
|            | Passing the search_id and search_rank parameter increments the article view count and records an entry for the article in the Knowledge Use [kb_use] table. You can also verify incremented view counts in Knowledge Base [kb_view2] page. Data type: String |
| search_rank| Optional unless using search_id. Article search rank by click-rate that you can retrieve using one of the following APIs that returns the articles.rank element:                                           |
|            | • Knowledge Management - GET /knowledge/articles  
|            | • Knowledge Management - GET /knowledge/articles/featured  
|            | • Knowledge Management - GET knowledge/articles/most_viewed  
|            | Data type: Number                                                                                                                                                                                            |
| update_view| Update view count and record an entry for the article in the Knowledge Use [kb_use] table. True whether present as a standalone parameter or set to true.                                                        |
|            | Note: If you pass update_view with search_id and search_rank, update_view is ignored because the view count will already be incremented.                                                                     |
|            | Data type: Boolean that is always handled as true when passed whether set to "true", "false", or not set at all.                                                                                      |
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attachments</td>
<td>Provides attachment details for each instance if attachment exists.</td>
</tr>
<tr>
<td></td>
<td>Only displays if <code>display_attachments = true</code>.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;attachments&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;file_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;size_bytes&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;state&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>attachments.file_name</td>
<td>File name of attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>attachments.size_bytes</td>
<td>File size.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Unit: Bytes</td>
</tr>
<tr>
<td>attachments.state</td>
<td>State.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• available</td>
</tr>
<tr>
<td></td>
<td>• available_conditionally</td>
</tr>
<tr>
<td></td>
<td>• not_available</td>
</tr>
<tr>
<td></td>
<td>• pending</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>attachments.sys_id</td>
<td>Sys_id of the attachment.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>content</td>
<td>Entire HTML content of the article.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>display_attachments</td>
<td>Flag that indicates whether the <code>display_attachments</code> flag is active for that</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>display_attachments</td>
<td>Article. Attachments are returned only if <code>display_attachments</code> is true (active) in the knowledge article record.</td>
</tr>
<tr>
<td>• true: <code>display_attachments</code></td>
<td>is active.</td>
</tr>
<tr>
<td>• false: <code>display_attachments</code></td>
<td>is inactive.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>embedded_content</td>
<td>Lists each attachment containing embedded content by sys_id and includes relevant attachment information. Only displays if <code>display_attachments</code> = true.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td>`&quot;attachments&quot;: [</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;file_name&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;size_bytes&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;state&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_id&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>]</td>
<td></td>
</tr>
<tr>
<td>embedded_content.file_name</td>
<td>File name of the attachment.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>embedded_content.size_bytes</td>
<td>Size of the attachment.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Unit: Bytes</td>
<td></td>
</tr>
<tr>
<td>embedded_content.state</td>
<td>State of the attachment. Possible values:</td>
</tr>
<tr>
<td>Data type: String</td>
<td>• available</td>
</tr>
<tr>
<td></td>
<td>• available_conditionally</td>
</tr>
<tr>
<td></td>
<td>• not_available</td>
</tr>
<tr>
<td></td>
<td>• pending</td>
</tr>
<tr>
<td>embedded_content.sys_id</td>
<td>Sys_id of the attachment.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>fields</td>
<td>Values of requested fields (if any). Data type: Object</td>
</tr>
<tr>
<td>fields.&lt;field_name&gt;</td>
<td>Lists each field requested using the fields parameter, if any. Data type: Object</td>
</tr>
<tr>
<td>fields.&lt;field_name&gt;.display_value</td>
<td>Display value of the requested field. Data type: String</td>
</tr>
<tr>
<td>fields.&lt;field_name&gt;.label</td>
<td>Label representing the requested field. For example, Knowledge. Data type: String</td>
</tr>
<tr>
<td>fields.&lt;field_name&gt;.name</td>
<td>Name of the requested field. Matches &lt;field_name&gt;. Data type: String</td>
</tr>
<tr>
<td>fields.&lt;field_name&gt;.type</td>
<td>Data type of requested field. Data type: String</td>
</tr>
<tr>
<td>fields.&lt;field_name&gt;.value</td>
<td>Value of the requested field. Data type: String</td>
</tr>
<tr>
<td>language</td>
<td>Two-letter ISO 639-1 language code for the current article (if translation is available). Data type: String</td>
</tr>
<tr>
<td>languages</td>
<td>For each translated version of a knowledge article (if translated):</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;languages&quot;: [</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;label&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;language&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;sys_id&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>]</td>
<td></td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>languages.label</td>
<td>String representation for language.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>languages.language</td>
<td>Two-letter ISO 639-1 code language.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>languages.sys_id</td>
<td>Unique identifier for translated version of knowledge article.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>number</td>
<td>Article number.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>short_description</td>
<td>Short description or title of the knowledge article.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>sys_id</td>
<td>Knowledge article sys_id from the Knowledge [kb_knowledge] table.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>template</td>
<td>Flag that indicates whether a returned article is a template.</td>
</tr>
<tr>
<td>Possible values:</td>
<td></td>
</tr>
<tr>
<td>• true: Article is a template.</td>
<td></td>
</tr>
<tr>
<td>• false: Article is not a template.</td>
<td></td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>template_table</td>
<td>Name of template table, only returns if knowledge article is a template.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>
Example: cURL request

curl
  "https://instance.servicenow.com/api/sn_km_api/knowledge/articles/0b48fd75474321009db4b5b08b9a71c2?search_id=spam&search_rank=26.426" \ 
  --request GET \ 
  --header "Accept:application/json" \ 
  --user "username":"password"

{
  "result": {
    "content": "<p><span style="font-size: 18pt;""><strong>How to Deal with Spam</strong></span></p>
    <p>Spam has increasingly become a problem on the Internet. While every Internet user receives some spam, email addresses posted to web sites or in newsgroups and chat rooms attract the most spam.</p>
    <p>To reduce the amount of spam you receive:</p>
    "template": false,
    "number": "KB0000011",
    "sys_id": "0b48fd75474321009db4b5b08b9a71c2",
    "short_description": "How to Deal with Spam",
    "display_attachments": true,
    "attachments": [
      {
        "sys_id": "dc27ae18294f4010f877796e707869c8",
        "file_name": "image.jpg",
        "size_bytes": "66792",
        "state": "available_conditionally"
      },
      {
        "sys_id": "fedf5614294f4010f877796e70786956",
        "file_name": "attachment.txt",
        "size_bytes": "75",
        "state": "available_conditionally"
      }
    ],
    "embedded_content": []
  }
}

Example: Sample cURL request (update_view)

curl "https://instance.servicenow.com/api/sn_km_api/knowledge/KB0000020?update_view=' \ 
  --request GET \ 
  --header "Accept:application/json" \ 
  --user "username":"password"
Should I upgrade to Windows 8.x?

Windows 8.x is designed for using touch, mouse, and keyboard together, on hardware ranging from touch-enabled tablets and laptops to PCs and all-in-one computers...(intentionally truncated)

Knowledge Management - GET knowledge/articles/most_viewed

Returns a list of knowledge articles prioritized by most-viewed.

URL format

Versioned URL: /api/sn_km_api/{api_version}/knowledge/articles/most_viewed

Default URL: /api/sn_km_api/knowledge/articles/most_viewed

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>Comma-separated list of fields from the Knowledge [kb_knowledge] table to show details in results.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kb</td>
<td>Comma-separated list of knowledge base sys_ids from the Knowledge Bases [kb_knowledge_base] table to restrict results to. Data type: String</td>
</tr>
<tr>
<td>language</td>
<td>List of comma-separated languages in two-letter ISO 639-1 language code format to restrict results to. Alternatively type 'all' to search in all valid installed languages on an instance. Data type: String Default: User's session language or en</td>
</tr>
<tr>
<td>limit</td>
<td>Maximum number of records to return. Unusually large limit values can impact system performance. For requests that exceed this number of records, use the offset parameter to paginate record retrieval. Data type: Number Default: 30</td>
</tr>
<tr>
<td>offset</td>
<td>Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks. For example, the first time this endpoint is called, offset is set to &quot;0&quot;. To page through all available records, use offset=offset+limit until the end of the all records is reached. Data type: Number Default: 0</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes
The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>articles</td>
<td>List of articles returned in response. Data type: Array</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>articles.fields</td>
<td>Values of requested fields (if any). Data type: Object</td>
</tr>
<tr>
<td>articles.fields.&lt;field_name&gt;</td>
<td>Lists each field requested using the fields parameter, if any. Data type: Object</td>
</tr>
<tr>
<td>articles.fields.&lt;field_name&gt;.display_value</td>
<td>Display value of the requested field. Data type: String</td>
</tr>
<tr>
<td>articles.fields.&lt;field_name&gt;.label</td>
<td>Label representing the requested field. For example, Knowledge. Data type: String</td>
</tr>
<tr>
<td>articles.fields.&lt;field_name&gt;.name</td>
<td>Name of the requested field. Matches &lt;field_name&gt;. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>articles.&lt;field_name&gt;.type</td>
<td>Data type of requested field. Data type: String</td>
</tr>
<tr>
<td>articles.&lt;field_name&gt;.value</td>
<td>Value of the requested field. Data type: String</td>
</tr>
<tr>
<td>articles.id</td>
<td>Knowledge article sys_id from the Knowledge [kb_knowledge] table. Data type: String</td>
</tr>
<tr>
<td>articles.link</td>
<td>Link to the article. Data type: String</td>
</tr>
<tr>
<td>articles.number</td>
<td>Knowledge article number. Data type: String</td>
</tr>
<tr>
<td>articles.rank</td>
<td>Search rank of article specific to this search. Data type: Float</td>
</tr>
<tr>
<td>articles.score</td>
<td>Relevancy score, results sorted in descending order by score. Data type: String</td>
</tr>
<tr>
<td>articles.snippet</td>
<td>Text showing a small portion of the knowledge article. Data type: String</td>
</tr>
<tr>
<td>articles.title</td>
<td>Short description or title of the knowledge article. Data type: String</td>
</tr>
<tr>
<td>meta</td>
<td>Meta information of the results and request parameters. Data type: Object</td>
</tr>
</tbody>
</table>

```
"meta": {  
"count": Number,  
"end": Number,  
"fields": "String",  
"filter": "String",  
"kb": "String",}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meta.count</td>
<td>Number of available KB articles. Data type: Number</td>
</tr>
<tr>
<td>meta.end</td>
<td>Ending index of the result set. Data type: Number</td>
</tr>
<tr>
<td>meta.fields</td>
<td>Fields in the article. Data type: String</td>
</tr>
<tr>
<td>meta.filter</td>
<td>Filter used to acquire the data. Data type: String</td>
</tr>
<tr>
<td>meta.kb</td>
<td>List of knowledge base article sys_ids. Data type: String</td>
</tr>
<tr>
<td>meta.language</td>
<td>List of comma-separated languages of the KB articles that were requested. Data type: String</td>
</tr>
<tr>
<td>meta.query</td>
<td>Specified request query. Data type: String</td>
</tr>
<tr>
<td>meta.start</td>
<td>Starting index of result set. Data type: Number</td>
</tr>
<tr>
<td>meta.status</td>
<td>HTTP status of the call. Data type: String</td>
</tr>
<tr>
<td>meta.ts_query_id</td>
<td>Sys_id of the query. Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

curl "https://instance.servicenow.com/api/sn_km_api/knowledge/articles/most_viewed?limit=5"
\--request GET \--header "Accept:application/json" \--user "username":"password"

{
  "result": {
    "meta": {
      "start": 0,
      "end": 5,
      "fields": ",",
      "query": ",",
      "filter": "workflow_state=published^valid_to>=javascript:gs.beginningOfToday()^active=true^sys_class_name!=kb_knowledge_block^sys_view_count>0^ORDERBYDESCsys_view_count^ORDERBYshort_description",
      "kb": ",",
      "count": 2,
      "status": {
        "code": 200
      },
      "language": "en"
    },
    "articles": [
      {
        "link": "?id=kb_article_view&sys_kb_id=0b48fd7547432109db4b5b08b9a71c2",
        "id": "kb_knowledge:0b48fd7547432109db4b5b08b9a71c2",
        "title": "How to Deal with Spam",
        "snippet": "How to Deal with Spam Spam has increasingly become a problem on the Internet. While every Internet user receives some spam, email addresses posted to web sites or in newsgroups and chat rooms attract the most spam. To reduce the amount of spam you receive: Don't reply to spam Be careful releasing your email address, and know how it will be used ",
        "score": 7,
        "tags": [],
        "number": "KB0000011"
      },
      {
        "link": "?id=kb_article_view&sys_kb_id=c85cd2519f77230088aebde8132e70c2",
        "id": "kb_knowledge:c85cd2519f77230088aebde8132e70c2",
        "title": "Microsoft Outlook Issues",
      }
    ]
  }
}
Microsoft Outlook Issues This article explains how to use automatic replies in Outlook 2010 for Exchange accounts. Setting Up Automatic Replies Click the File tab. Click Automatic Replies. Select Send automatic replies. If desired, select the Only send during this time range check box to schedule when your out of office replies are active. If yo",

MetricBase Time Series API

Use the MetricBase Time Series API to insert data into, retrieve information from, and to run transforms against a MetricBase database.

These APIs can only be used when the MetricBase plugin (com.snc.clotho) is installed and activated.

Role required to write to this API: clotho_rest_put

The examples in this section were created using data in the MetricBase Demo (com.snc.clotho.demo) plugin.

MetricBase Time Series - GET /now/clotho/table/{table}/{subject}/{metric}

Retrieves specified time series data from the MetricBase database.

**URL format**

Versioned URL: /api/now/{api_version}/clotho/table/{table}/{subject}/{metric}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>metric</td>
<td>Name of the column in the table identified in the subject parameter to use as the metric.</td>
</tr>
</tbody>
</table>
### Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject</td>
<td>Sys_id of the GlideRecord associated with this series.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>table</td>
<td>Name of the table containing the GlideRecord associated with this series.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_display_value</td>
<td>Flag that indicates whether the result data is labeled with the subject record display value if no other label is specified. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Result data is labeled with the subject record display value.</td>
</tr>
<tr>
<td></td>
<td>• false: Result data is not labeled with the subject record display value.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>sysparm_end</td>
<td>Required. End time of the evaluation period. An empty or missing value is treated as the current time.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Format: ISO 8601 (UTC), either:</td>
</tr>
<tr>
<td></td>
<td>• Absolute date format [YYYY-MM-DDThh:mm:ss], such as 2019-03-20T17:04:55.</td>
</tr>
<tr>
<td></td>
<td>• Relative to current time duration format [P(n)Y(n)M(n)DT(n)H(n)M(n)S], such as P1M.</td>
</tr>
<tr>
<td>sysparm_start</td>
<td>Required. Start time of the evaluation period. The special value all can be used to set the start time as the current time minus the maximum retention period</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>for the specified metrics. An empty or missing value is treated as an implicit all.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Format: ISO 8601 (UTC), either:</td>
</tr>
<tr>
<td></td>
<td>• Absolute date format [YYYY-MM-DDThh:mm:ss], such as 2019-03-20T17:04:55</td>
</tr>
<tr>
<td></td>
<td>• Relative to current time duration format [P(n)Y(n)M(n)DT(n)H(n)M(n)S], such as P1M.</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>The label used to identify this metric’s result set. Data type: String</td>
</tr>
<tr>
<td>seriesRef</td>
<td>Data series results. Data type: Array</td>
</tr>
<tr>
<td>seriesRef.metric</td>
<td>The name of the metric from which the data series was obtained. Only appears if the metrics object is passed in the request. Data type: String</td>
</tr>
<tr>
<td>seriesRef.subject</td>
<td>The sys_id of the data series record. Located in the table specified in seriesRef.table. Data type: String</td>
</tr>
<tr>
<td>seriesRef.table</td>
<td>The name of the table from where the data series was obtained. Data type: String</td>
</tr>
<tr>
<td>values</td>
<td>The transformed series values.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>values:</td>
<td>The ISO 8601 timestamp of the value.</td>
</tr>
<tr>
<td>values.timestamp</td>
<td>The ISO 8601 timestamp of the value.</td>
</tr>
<tr>
<td>values.value</td>
<td>The metric values.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl "https://instance.servicenow.com/api/now/v1/clotho/table/mb_demo_drone/626b051787333200a328c5b836cb0b99/mb_demo_mt_altitude?sysparm_start=2019-03-20T17%3A04%3A55&sysparm_end=2019-03-20T17%3A09%3A55" \
   --request GET \
   --header "Accept:application/json" \
   --user "username":"password"
```

```json
{
   "seriesRef": {
      "subject": "626b051787333200a328c5b836cb0b99",
      "table": "mb_demo_drone",
      "metric": "mb_demo_mt_altitude"
   },
   "label": "626b051787333200a328c5b836cb0b99:mb_demo_drone|mb_demo_mt_altitude",
   "values": [
      {
         "timestamp": "2019-03-20T17:05:00Z",
         "value": 83.150185
      },
      {
         "timestamp": "2019-03-20T17:06:00Z",
         "value": 83.46074
      }
   ]
}```
Example: Python request

import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/v1/clotho/table/mb_demo_drone/626b051787333200a328c5b836cb0b99/mb_demo_mt_altitude?sysparm_start=2019-03-20T17%3A04%3A55&sysparm_end=2019-03-20T17%3A09%3A55'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {"Accept":"application/json"}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "seriesRef": {
        "subject": "626b051787333200a328c5b836cb0b99",
        "table": "mb_demo_drone",
        "metric": "mb_demo_mt_altitude"
    },
    "label": "626b051787333200a328c5b836cb0b99:mb_demo_drone|mb_demo_mt_altitude",
    "values": [
        {
            "timestamp": "2019-03-20T17:05:00Z",
            "value": 83.150185
        },
        {
            "timestamp": "2019-03-20T17:06:00Z",
            "value": 83.46074
        },
        {
            "timestamp": "2019-03-20T17:07:00Z",
            "value": 83.83104
        },
        {
            "timestamp": "2019-03-20T17:08:00Z",
            "value": 84.260635
        },
        {
            "timestamp": "2019-03-20T17:09:00Z",
            "value": 84.749
        }
    ]
}
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>metric</td>
<td>Name of the column in the table identified in the subject parameter to use as the metric. Data type: String</td>
</tr>
<tr>
<td>table</td>
<td>Name of the table containing the GlideRecord associated with this series. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| sysparm_display_value | Flag that indicates whether the result data is labeled with the subject record display value if no other label is specified. Valid values:  
  • true: Result data is labeled with the subject record display value.  
  • false: Result data is not labeled with the subject record display value.  
  Data type: Boolean  
  Default: false                                                                 |
| sysparm_end           | Required. End time of the evaluation period. An empty or missing value is treated as the current time.  
  Data type: String  
  Format: ISO 8601 (UTC), either:                                                                                     |
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_query</td>
<td>Encoded query string for finding the subject records. Data type: String Default: None</td>
</tr>
</tbody>
</table>
| sysparm_start      | Required. Start time of the evaluation period. The special value all can be used to set the start time as the current time minus the maximum retention period for the specified metrics. An empty or missing value is treated as an implicit all. Data type: String Format: ISO 8601 (UTC), either:  
  - Absolute date format [YYYY-MM-DDThh:mm:ss], such as 2019-03-20T17:04:55  
  - Relative to current time duration format [P(n)Y(n)M(n)DT(n)H(n)M(n)S], such as P1M.                                                                                                           |
| sysparm_subject_limit | Limit size of the subject query result. Data type: Number Default: 10,000                                                                                                                                     |
| sysparm_transforms | Comma separated list of transforms. Valid transforms:  
  - add  
  - avg  
  - label  
  - mul  
  - resample  
  - sum  
  - top |
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: None</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>The label used to identify this metric’s result set. Data type: String</td>
</tr>
<tr>
<td>seriesRef</td>
<td>Data series results. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;seriesRef&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;metric&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;subject&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;table&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>seriesRef.metric</td>
<td>The name of the metric from which the data series was obtained. Only appears if the metrics object is passed in the request. Data type: String</td>
</tr>
<tr>
<td>seriesRef.subject</td>
<td>The sys_id of the data series record. Located in the table specified in seriesRef.table. Data type: String</td>
</tr>
<tr>
<td>seriesRef.table</td>
<td>The name of the table from where the data series was obtained. Data type: String</td>
</tr>
<tr>
<td>values</td>
<td>The transformed series values. Data type: Array</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;values&quot;: [</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;timestamp&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;value&quot;: Number }</td>
<td></td>
</tr>
<tr>
<td>]</td>
<td></td>
</tr>
</tbody>
</table>

values.timestamp

The ISO 8601 timestamp of the value.
Data type: String

values.value

The metric values.
Data type: Number

Example: cURL request


[                        |
|  |
| {                        |
|   "seriesRef": {        |
|     "subject": "2a6b051787333200a328c5b836cb0b92", |
|     "table": "mb_demo_drone", |
|     "metric": "mb_demo_mt_speed" } |
| }, |
| "label": "2a6b051787333200a328c5b836cb0b92:mb_demo_drone |mb_demo_mt_speed", |
| "values": [            |
| {                       |
|   "timestamp": "2019-03-25T17:05:00Z", |
|   "value": 33.67892     |
| }                       |
| ]                        |
| },                          |
| {                          |
|   "seriesRef": {          |
| },                        |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
"subject": "666b051787333200a328c5b836cb0b92",
"table": "mb_demo_drone",
"metric": "mb_demo_mt_speed"
},
"label": "666b051787333200a328c5b836cb0b92:mb_demo_drone
|mb_demo_mt_speed",
"values": [
{
"timestamp": "2019-03-25T17:05:00Z",
"value": 41.94985
}
],
"seriesRef": {
"subject": "a26b051787333200a328c5b836cb0b92",
"table": "mb_demo_drone",
"metric": "mb_demo_mt_speed"
},
"label": "a26b051787333200a328c5b836cb0b92:mb_demo_drone
|mb_demo_mt_speed",
"values": [
{
"timestamp": "2019-03-25T17:05:00Z",
"value": 37.74187
}
],
"seriesRef": {
"subject": "ea6b051787333200a328c5b836cb0b92",
"table": "mb_demo_drone",
"metric": "mb_demo_mt_speed"
},
"label": "ea6b051787333200a328c5b836cb0b92:mb_demo_drone
|mb_demo_mt_speed",
"values": [
{
"timestamp": "2019-03-25T17:05:00Z",
"value": 34.914192
}
]
Example: Python request

```python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/v1/
clotho/transform/mb_demo_drone/mb_demo_mt_speed?
sysparm_query=model%3DKingfisher%20Phantom&
sysparm_start=2019-03-25T17%3A04%3A55&
sysparm_end=2019-03-25T17%3A05%3A10'

# Eg. User name="username", Password="password" for code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept':"application/json"}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd),
                        headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:',
          response.headers, 'Error Response:', response.json())
    exit()
```
# Decode the JSON response into a dictionary and use the data

data = response.json()
print(data)

[
{
  "seriesRef": {
    "subject": "2a6b051787333200a328c5b836cb0b92",
    "table": "mb_demo_drone",
    "metric": "mb_demo_mt_speed"
  },
  "label": "2a6b051787333200a328c5b836cb0b92:mb_demo_drone
  |mb_demo_mt_speed",
  "values": [
    {
      "timestamp": "2019-03-25T17:05:00Z",
      "value": 33.67892
    }
  ]
},
...
{
  "seriesRef": {
    "subject": "ee6b051787333200a328c5b836cb0b91",
    "table": "mb_demo_drone",
    "metric": "mb_demo_mt_speed"
  },
  "label": "ee6b051787333200a328c5b836cb0b91:mb_demo_drone
  |mb_demo_mt_speed",
  "values": [
    {
      "timestamp": "2019-03-25T17:05:00Z",
      "value": 44.170887
    }
  ]
}
MetricBase Time Series - POST /now/clotho/transform

Retrieves and optionally groups time series data for one or more metrics after applying a specified list of transforms that form a linear pipeline.

**URL format**

Versioned URL: `/api/now/{api_version}/clotho/transform`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| sysparm_display_value           | Flag that indicates whether the result data is labeled with the subject record display value if no other label is specified. Valid values:  
• true: Result data is labeled with the subject record display value.  
• false: Result data is not labeled with the subject record display value.  
Data type: Boolean  
Default: false |

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>end</td>
<td>Required. End time of the evaluation period. An empty or missing value is treated as the current time. Data type: String</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Description</strong></td>
</tr>
</tbody>
</table>
| limit             | Maximum number of records to return. Unusually large values can impact system performance.  
Data type: Number  
Default: 10,000    |
| metrics           | List of metrics objects to use in the transform. For more information on metrics, see [Metrics](#).  
Data type: Array   |
| metrics.groupBy   | Comma-separated list of fields, contained in the table specified in the **table** parameter, to group the series by before collecting or applying aggregated transformations.  
  The order of the items in the list is the order in which the data is grouped by. For example, if there are two items in the list, "state, zip code", then the series is first grouped by state and then by zip code within the state.  
Data type: String  
Default: Data is not grouped. |
| metrics.label     | Label to use for this metric's result set. It replaces any labels generated by the transform chain.  
Data type: String  |

Format: ISO 8601 (UTC), either:
- Absolute date format [YYYY-MM-DDThh:mm:ss], such as 2019-03-20T17:04:55.
- Relative to current time duration format [P(n)Y(n)M(n)DT(n)H(n)M(n)S], such as P1M.
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metrics.metric</td>
<td>Required if <code>metrics</code> object is passed. Metric field to use in the transform. This field must be in the table specified in the <code>table</code> parameter. Data type: String</td>
</tr>
<tr>
<td>metrics.transforms</td>
<td>List of transforms (transform chain) to apply to the retrieved time series data. Each transform builds on the results of the previous transform. For a list of available transforms, see Supported Transforms below. Data type: Array</td>
</tr>
</tbody>
</table>
| metrics.transforms.arg | Dependent on the transform. Parameter or parameters to pass into the transform. General guideline:  
  - Do not use the `arg` parameter when specifying transformations that do not take a parameter.  
  - Use Number, String, or Boolean for transforms that take a single parameter.  
  - Use a JSON object, with the appropriate name-value pairs, for transforms that take more than one parameter. Data type: Number, String, Boolean, or JSON object, depending on transform. (For a list of available transforms, refer to the table Supported transforms below.) |
| metrics.transforms.name | Required if a `transforms` object is specified. Name of the transform. |
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>query</strong></td>
<td>Encoded query to use to filter the result set. You can compose the query using the specified table's filter editor. Once created, select Copy URL from the filter's breadcrumbs context menu. Data type: String Default: None</td>
</tr>
<tr>
<td><strong>start</strong></td>
<td>Required. Start time of the evaluation period. The special value <code>all</code> can be used to set the start time as the current time minus the maximum retention period for the specified metrics. An empty or missing value is treated as an implicit <code>all</code>. Data type: String Format: ISO 8601 (UTC), either: • Absolute date format <code>[YYYY-MM-DDThh:mm:ss]</code>, such as <code>2019-03-20T17:04:55</code> • Relative to current time duration format <code>[P(n)Y(n)M(n)DT(n)H(n)M(n)S]</code>, such as <code>P1M</code>.</td>
</tr>
<tr>
<td><strong>table</strong></td>
<td>Required. Name of the table that contains the GlideRecord associated with this series. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results</td>
<td>Zero or more return result objects that satisfy the query.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Contains either a <strong>series</strong> array or a <strong>grouped</strong> array.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;results&quot;:{</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;grouped&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;marker&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;series&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>results.grouped</td>
<td>List of grouped result objects where each <strong>grouped</strong> result object corresponds to a specific group. This format is returned when the <strong>metrics.groupBy</strong> parameter is passed in the request body.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;grouped&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;groupingBy&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;groups&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>results.grouped.groupingBy</td>
<td>Value passed in the <strong>metrics.groupBy</strong> parameter in the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>&quot;groupingBy&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;groups&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>results.grouped.groups</td>
<td>List of generated groups based on the content of the <strong>metrics.groupBy</strong> parameter in the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;groups&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;group&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;label&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>results.grouped.groups.group</td>
<td>Name of the group.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>results.grouped.groups.label</td>
<td>Label of the group.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>results.grouped.groups.series</td>
<td>Transformed values for the group's result.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;series&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;label&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;seriesRef&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;values&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>results.grouped.groups.series.label</td>
<td>Label of the series.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>results.grouped.groups.series.seriesRef</td>
<td>Results of a single data series. Does not appear if there are multiple data series in the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;seriesRef&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;metric&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;subject&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;table&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>results.grouped.groups.series.seriesRef.metric</td>
<td>Name of the metric from which the data series was obtained. Only appears if the metrics object is passed in the request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>results.grouped.groups.series.seriesRef.subject</td>
<td>Sys_id of the data series record. Located in the table specified in <code>results.series.seriesRef.table</code>. Data type: String</td>
</tr>
<tr>
<td>results.grouped.groups.series.seriesRef.table</td>
<td>Name of the table from where the data series was obtained. Data type: String</td>
</tr>
<tr>
<td>results.grouped.groups.series.values</td>
<td>List of result values for the group. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&quot;values&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;timestamp&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: Number</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>results.grouped.groups.series.values.timestamp</td>
<td>ISO 8601 timestamp of the value. Data type: String</td>
</tr>
<tr>
<td>results.grouped.groups.series.values.value</td>
<td>Transformed value. Data type: Number</td>
</tr>
<tr>
<td>results.marker</td>
<td>Unique identifier for the corresponding results. Data type: String</td>
</tr>
<tr>
<td>results.series</td>
<td>List of ungrouped series of result objects. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&quot;series&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;label&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;seriesRef&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;values&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>results.series.label</td>
<td>Data series label. This value is either generated by the endpoint or is the value passed in the <code>metrics.label</code> parameter. Data type: String</td>
</tr>
<tr>
<td>results.series.seriesRef</td>
<td>Results of a single data series. Does not appear if there are multiple data series in the response. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;seriesRef&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;metric&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;subject&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;table&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>results.series.seriesRef.metric</td>
<td>Name of the metric from which the data series was obtained. Only appears if the <code>metrics</code> object is passed in the request. Data type: String</td>
</tr>
<tr>
<td>results.series.seriesRef.subject</td>
<td>Sys_id of the data series record. Located in the table specified in <code>results.series.seriesRef.table</code>. Data type: String</td>
</tr>
<tr>
<td>results.series.seriesRef.table</td>
<td>Name of the table from where the data series was obtained. Data type: String</td>
</tr>
<tr>
<td>results.series.values</td>
<td>Transformed series values. Data type: Array</td>
</tr>
</tbody>
</table>
### Supported transforms

<table>
<thead>
<tr>
<th>Transform</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>add</td>
<td>Adds the specified number to each timestamp value. Arguments:</td>
</tr>
<tr>
<td></td>
<td>• arg (Decimal): The number to add.</td>
</tr>
<tr>
<td>autocorrelate</td>
<td>Calculates the correlation between timestamp values separated by an increasing number of periods starting at 1.</td>
</tr>
<tr>
<td>avg</td>
<td>Aggregates the time series into one series containing the average value for each corresponding timestamp across the input.</td>
</tr>
<tr>
<td>bottom</td>
<td>Returns the specified number of series that have the lowest values. Arguments:</td>
</tr>
<tr>
<td></td>
<td>• arg (Integer): The number of series to return.</td>
</tr>
</tbody>
</table>

**results.series.values.timestamp**

ISO 8601 timestamp of the value.
Data type: String

**results.series.values.value**

Transformed value.
Data type: Number
<table>
<thead>
<tr>
<th>Transform</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>For example, the following returns the two time series that contain the lowest sets of values.</td>
</tr>
<tr>
<td></td>
<td>&quot;transforms&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;bottom&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;arg&quot;: 2</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>ceil</td>
<td>Rounds the value in each timestamp up to the specified precision: <code>ceil(value / &lt;arg&gt;) * &lt;arg&gt;)</code></td>
</tr>
<tr>
<td></td>
<td>Arguments:</td>
</tr>
<tr>
<td></td>
<td>• arg: <strong>Decimal</strong>: The decimal precision to round up to.</td>
</tr>
<tr>
<td>collect</td>
<td>Displays the transform results of the transform chain up to the point of the collect transform call. Collect transform results contain a unique marker, but you may want to also define a label.</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
</tbody>
</table>
|           | ```
|           |   { |
|           |     "start": "PT1H", |
|           |     "end": "", |
|           |     "table": "mb_demo_drone", |
|           |     "limit": 5000, |
|           |     "metrics": [ |
|           |       { |
|           |         "metric": "mb_demo_dr_altitude", |
|           |         "transforms": [ |
|           |           { |
|           |             "name": "label", |
|           |             "arg": "Series Timestamp Values" |
|           |           }, |
|           |           { |
|           |             "name": "collect" |
|           |           }, |
|           |           { |
|           |             "name": "avg" |
|           |           }, |
|           |           { |
|           |             "name": "label", |
|           |             "arg": "Average Timestamp Values" |
|           |       } |
|           |   } |
```
<table>
<thead>
<tr>
<th>Transform</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>constrainValues</td>
<td>Replaces any value outside the specified range with the corresponding maximum or minimum value. For example, if you specified the following, the transform replaces any value that is less than 0 with 0, and any value that is more than 100 with 100.</td>
</tr>
</tbody>
</table>

```
"transforms": [
  {
    "name": "constrainValues",
    "val1": 0,
    "val2": 100
  }
]
```

Arguments:
- `val1` (**Decimal**): Minimum or maximum value.
- `val2` (**Decimal**): Minimum or maximum value.

| count         | Aggregates the time series into one series. The new series contains the number of values that are not NaN (Not a Number) for each corresponding timestamp across the series. |

| derivative    | Determines the rate of change between timestamps. Divides the difference between the value in each timestamp and the value in the next timestamp by the timestamp’s period.  

**Note:** This transform returns one less value than the number of values in the series.

| div           | Divides the value in each timestamp by the specified number (arg).  
Arguments:
- `arg` (**Decimal**): The number by which to divide the value of each timestamp. |

<p>| envelope      | Returns two time series, where, at any point in time, one contains the largest value and the other contains the smallest value. |</p>
<table>
<thead>
<tr>
<th>Transform</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note</strong>: NaNs are ignored, but are returned if there are no other return values.</td>
<td></td>
</tr>
</tbody>
</table>
| **exp** | Raises the value of the specified base to the power of the value in each timestamp. Arguments:  
- arg (Decimal): The base value. |
| **filter** | Applies an aggregator to the contents of a sliding window, such as producing a moving average. Arguments:  
- aggregator (Aggregator): Type of aggregation to perform.  
- window (Duration): The duration of the sliding window. |
| **floor** | Rounds the value in each timestamp down to the specified precision: floor(value / <arg>) * <arg>  
Arguments:  
- arg (Decimal): The decimal precision to round down to. |
| **fractiles** | Returns a time series for each fraction in the specified array. Each timestamp value is the value at which the specified fraction of values, for the corresponding timestamp across the input series, is below the specified percentage. For example, if the fraction is 0.5, then the value in the timestamp is the value where half the values in the input series are below 0.5. Arguments:  
- arg: (Array of Decimals): The fractions to use on the input series. For example:  
```json
"transforms": [  
  {  
    "name": "fractiles",  
    "arg": [0.25, 0.5, 0.75, 1]  
  }  
]
```
<p>| <strong>groupBy</strong> | Groups data by the specified fields before collecting or applying aggregated transformations. Arguments: |</p>
<table>
<thead>
<tr>
<th>Transform</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>arg</strong> (String): A comma separated list of fields in the table to use to group the transform results.</td>
<td></td>
</tr>
<tr>
<td>For example:</td>
<td></td>
</tr>
<tr>
<td>&quot;transforms&quot;: [</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;name&quot;: &quot;groupBy&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;arg&quot;: &quot;model&quot;</td>
<td></td>
</tr>
<tr>
<td>},</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;name&quot;: &quot;avg&quot;</td>
<td></td>
</tr>
<tr>
<td>},</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;name&quot;: &quot;label&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;arg&quot;: &quot;Model: %g&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>]</td>
<td></td>
</tr>
<tr>
<td>integrate</td>
<td>Multiplies the value in each timestamp by its period.</td>
</tr>
<tr>
<td>interpolate</td>
<td>Creates a data value for an NaN data item by interpolating from adjacent data values. Arguments:</td>
</tr>
<tr>
<td>• <strong>arg</strong> (Integer): Number of data samples in each direction to check for a non-NaN value. If a non-NaN value is not found, then NaN is used.</td>
<td></td>
</tr>
<tr>
<td>inverse</td>
<td>Computes the inverse of each timestamp value.</td>
</tr>
<tr>
<td>iqr</td>
<td>Performs an interquartile range transform and creates a result set that contains four series:</td>
</tr>
<tr>
<td>1. <strong>-IQR</strong>: The median of all entries below ( Q1 - (1.5 \times IQR) ).</td>
<td></td>
</tr>
<tr>
<td>2. <strong>Q1</strong>: The median of the smallest half of entries.</td>
<td></td>
</tr>
<tr>
<td>3. <strong>Q3</strong>: The median of the largest half of entries.</td>
<td></td>
</tr>
<tr>
<td>4. <strong>+IQR</strong>: The median of all entries above ( Q3 + (1.5 \times IQR) ).</td>
<td></td>
</tr>
<tr>
<td><strong>Note</strong>: ( IQR = Q3 - Q1 )</td>
<td></td>
</tr>
<tr>
<td>label</td>
<td>Labels a transformation chain.</td>
</tr>
<tr>
<td><strong>Note</strong>: Subsequent transformations may modify or replace the label.</td>
<td></td>
</tr>
<tr>
<td>Transform</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Arguments:</strong></td>
<td></td>
</tr>
<tr>
<td>• arg (String): Text of the label. Can contain the following formatting expressions:</td>
<td></td>
</tr>
<tr>
<td>◦ %%%: Escape a &quot;%%&quot; literal.</td>
<td></td>
</tr>
<tr>
<td>◦ %l: Current label that is being replaced.</td>
<td></td>
</tr>
<tr>
<td>◦ %%s: Series subject.</td>
<td></td>
</tr>
<tr>
<td>◦ %g: Value of the group by field. If this is a referenced record, then the value of the record’s name field. If multiple groups are specified in the <code>groupBy</code> transform, labels are comma separated.</td>
<td></td>
</tr>
<tr>
<td>◦ %G: Value of the group by field. If this is a referenced record, then the record's sys_id. If multiple groups are selected, values are comma separated.</td>
<td></td>
</tr>
<tr>
<td>For example:</td>
<td></td>
</tr>
<tr>
<td>&quot;transforms&quot;: [</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;name&quot;: &quot;label&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;arg&quot;: &quot;Series Timestamp Values&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>]</td>
<td></td>
</tr>
</tbody>
</table>

"transforms": [ |
| { | |
| "name": "groupBy", | |
| "arg": "model" | |
| } | |
| , | |
| { | |
| "name": "avg" | |
| } | |
| , | |
| { | |
| "name": "label", | |
| "arg": "Model: %g" | |
| } | |
| ] | |

**limit**

Returns, at most, the specified duration or number of values, starting with the most recently saved value.

**Arguments:**
<table>
<thead>
<tr>
<th>Transform</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• arg (Integer): Number of timestamp values to return for each time series. OR • arg (Duration): Duration to limit each time series to.</td>
</tr>
<tr>
<td>log</td>
<td>Runs a logarithm on the value in each timestamp where the result is the log of the specified base for the timestamp value. Arguments:</td>
</tr>
<tr>
<td></td>
<td>• arg (Decimal): Base for the logarithm calculation.</td>
</tr>
<tr>
<td>mapValues</td>
<td>Replaces any values within the specified range (inclusive) with the specified value. If both lowerBound and upperBound are specified as NaN, then it replaces any NaN value with the targetValue.</td>
</tr>
<tr>
<td></td>
<td>Arguments:</td>
</tr>
<tr>
<td></td>
<td>• lowerBound (Decimal): The lowest value in the range.</td>
</tr>
<tr>
<td></td>
<td>• upperBound (Decimal): The highest value in the range.</td>
</tr>
<tr>
<td></td>
<td>• targetValue (Decimal): Replacement value.</td>
</tr>
<tr>
<td></td>
<td>For example, the following changes all values in the time series that are between .1 and .9 to 1:</td>
</tr>
<tr>
<td></td>
<td>&quot;transforms&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;mapValues&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;lowerBound&quot;: .1,</td>
</tr>
<tr>
<td></td>
<td>&quot;upperBound&quot;: .9,</td>
</tr>
<tr>
<td></td>
<td>&quot;targetValue&quot;: 1</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>max</td>
<td>Returns a series that contains the maximum value for each corresponding timestamp across the input.</td>
</tr>
<tr>
<td>median</td>
<td>Creates a series that contains the median of values for each timestamp across a set of series. If there are n series:</td>
</tr>
<tr>
<td>Transform</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Transform</strong></td>
<td><strong>Description</strong></td>
</tr>
</tbody>
</table>
| • If \( n \) is odd, the median is the \( \frac{n}{2} + 1 \) value for a timestamp.  
  • If \( n \) is even, the median is the average of the \( \frac{n}{2} \) and \( \frac{n}{2} + 1 \) values for a timestamp. |
| **min** | Returns a series that contains the minimum value for each corresponding timestamp across the input. |
| **mul** | Multiplies the value in each timestamp by the specified number.  
Arguments:  
• arg (Decimal): Number by which to multiply the value of each timestamp. |
| **partition** | Produces a new series with values filtered by applying a specified aggregator to a non-overlapping window.  
Arguments:  
• aggregator (Aggregator): Type of aggregation to perform.  
• window (Duration): The duration of the non-overlapping window to apply the aggregator.  
• base (DateTime): The zero offset to use for partitioning.  
  For example, to partition by day (24h), set this value to Monday at midnight in your time zone. To partition by a 30-day period, set this value to the first day of the most recent month.  
  Default: Beginning of the EPOCH. |
| **pow** | Raises each timestamp value to the specified power.  
Arguments:  
• arg: (Decimal): Power to which to raise each value. |
| **product** | Aggregates the selected metric series into a single series that contains the product of all values for each timestamp. NaNs are excluded. If all numbers in the series are NaN, the output is also NaN. |
| **resample** | Resamples a time series to either a fixed number of points or from one frequency to another.  
Use the resample transformation to reduce the number of samples in the result set to more closely match the number of samples that you want to display. |
<table>
<thead>
<tr>
<th>Transform</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transform</td>
<td>You can pass various arguments for this transform. Below is a list of each of the possible argument groups. Only one of these argument groups can be passed within a single transform. The following argument groups are delineated by AND/OR. AND meaning the argument is part of the current group; OR meaning it is the start of a new argument group.</td>
</tr>
<tr>
<td></td>
<td>• arg (Integer): Number of samples to include in the result set. If available samples are less than this number, the endpoint uses interpolation to create additional samples. If there are more samples than this number, they are reduced by averaging them. OR</td>
</tr>
<tr>
<td></td>
<td>• arg (Duration): The frequency at which to resample. If available samples are less than this number, the endpoint uses interpolation to create additional samples. If there are more samples than this number, they are reduced by averaging them. OR</td>
</tr>
<tr>
<td></td>
<td>• minValues (Integer): The minimum number of samples to include in the result set. If available samples are less than this number, the endpoint uses interpolation to create additional samples. AND</td>
</tr>
<tr>
<td></td>
<td>• maxValues (Integer): The maximum number of samples to include in the result set. If there are more samples than this number, they are reduced by averaging them. OR</td>
</tr>
<tr>
<td></td>
<td>• aggregator (Aggregator): The aggregator to use when resampling. AND</td>
</tr>
<tr>
<td></td>
<td>• values (Integer): The number of samples to include in the result set. If available samples are less than this number, the endpoint uses interpolation to create additional samples. If there are more samples than this number, they are reduced by applying the specified aggregator to them. OR</td>
</tr>
<tr>
<td>Transform</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>• aggregator (<em>Aggregator</em>): The aggregator to use when resampling. AND • period (<em>Duration</em>): The frequency at which to resample. If available samples are less than this number, the endpoint uses interpolation to create additional samples. If there are more samples than this number, they are reduced by applying the specified <em>aggregator</em> to them. OR • aggregator (<em>Aggregator</em>): The aggregator to use when resampling. AND • minValues (Integer): The minimum number of samples to include in the result set. If available samples are less than this number, the endpoint uses interpolation to create additional samples. AND • maxValues (Integer): The maximum number of samples to include in the result set. If there are more samples than this number, they are reduced by averaging them.</td>
<td></td>
</tr>
</tbody>
</table>

**round**

Rounds the value in each timestamp up to the specified precision: `(value / <arg>) * <arg>`

Arguments:

• arg (*Decimal*): The decimal precision to round up to.

**root**

Calculates the root of each timestamp value using the specified index.

Arguments:

• arg (*Decimal*): The index of the root.

For example, the following returns the square root of each timestamp:

```json
"transforms": [
  {
    "name": "root",
    "arg": "2"
  }
]"transforms": [  
  {
    "name": "root",
    "arg": "2"
  }
]```
<table>
<thead>
<tr>
<th>Transform</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stddev</td>
<td>Creates a series that contains the standard deviation of values for each timestamp across a set of series.</td>
</tr>
</tbody>
</table>
| sub       | Subtracts the specified number from the value in each timestamp. Arguments:  
|           | • arg (Number): The number to subtract from the value in each timestamp. |
| sum       | Aggregates the selected metric series into one series that contains the sum of all values for each timestamp, excluding any NaNs. If all numbers in the series are NaN, the output is NaN. |
| timeshift | Shifts the time range by adding the specified offset to a value’s timestamp without modifying the value. Use this transform to shift timestamps to another timezone. Arguments:  
|           | • arg: (Duration): The amount of time to shift by. |
| top       | Returns the specified number of series that have the highest values. Arguments:  
|           | • arg (Integer): The number of series to return.  
|           | For example, the following returns the two time series that contain the highest sets of values.  
|           | "transforms": [  
|           |   {  
|           |     "name": "top",  
|           |     "arg": 2  
|           |   }  
|           | ] |

### Transform data types

<table>
<thead>
<tr>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregator</td>
<td>Available aggregators to use within the transform.</td>
</tr>
</tbody>
</table>
### Transform data types (continued)

<table>
<thead>
<tr>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• AVG: Calculates the arithmetic mean of all currently selected series.</td>
<td></td>
</tr>
<tr>
<td>• CHISQUARE: Shows how well a statistical model fits the metric dataset.</td>
<td></td>
</tr>
<tr>
<td>• LAST: Returns the last defined value in the period window.</td>
<td></td>
</tr>
<tr>
<td>• MAX: Shows the largest value for the metric dataset, at each point in time.</td>
<td></td>
</tr>
<tr>
<td>• MEDIAN: Shows the median of the metric dataset. The median separates the higher values of the metric dataset from the lower values.</td>
<td></td>
</tr>
<tr>
<td>• MIN: Shows the smallest value for the metric dataset, at each point in time.</td>
<td></td>
</tr>
<tr>
<td>• STDDEV: Calculates the standard deviation across the underlying data. Used to quantify the variation or dispersion of a set of data values in the metric dataset.</td>
<td></td>
</tr>
<tr>
<td>• SUM: Calculates the sum across all currently selected series.</td>
<td></td>
</tr>
</tbody>
</table>

#### DateTime

Absolute date/time values are specified by the ISO 8601 date and time format: `YYYY-MM-DDThh:mm:ss`. For example: `2020-02-28T13:10:42`.

Where:

- `[YYYY]`: Four-digit year.
- `[MM]`: Zero-padded month between 01 and 12.
- `[DD]`: Zero-padded day between 01 and 31.
- `T`: Represents time and is mandatory before any of the time components are specified.
- `[hh]`: Zero-padded hour between 00 and 23.
- `[mm]`: Zero-padded minutes between 00 and 59.
- `[ss]`: Zero-padded seconds between 00 and 59. Note that leap seconds are spread out evenly across the previous 1,000 seconds.

Relative date/time values are specified using a `duration`, which is subtracted from the current time.
### Transform data types (continued)

<table>
<thead>
<tr>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Decimal** | Decimal value that can either be:  
- A signed decimal number of arbitrary precision that can use exponential notation.  
  Such as:  
  - 2  
  - 1.7  
  - -3.47  
  - 1.0E+2  
- A string representation of:  
  - A signed decimal number that may use exponential notation and is guaranteed to be a double-precision 64-bit IEEE 754 floating-point number.  
  - "NaN"  
  - "-Infinity"  
  - "+Infinity" |
  Where:  
  - P: Duration designator, referred to as "period", and is always placed at the beginning of the duration.  
  - [n]Y: Number of years.  
  - [n]M: Number of months.  
  - [n]D: Number of days.  
  - T: Time designator and is mandatory before any of the time components are specified.  
  - [n]H: Number of hours.  
  - [n]M: Number of minutes.  
  - [n]S: Number of seconds. |
Transform data types (continued)

<table>
<thead>
<tr>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>For example:</td>
<td></td>
</tr>
<tr>
<td>• P7D:</td>
<td>Period of seven days.</td>
</tr>
<tr>
<td>• P1M:</td>
<td>Period of one month.</td>
</tr>
<tr>
<td>• PT15M:</td>
<td>Period of 15 minutes.</td>
</tr>
<tr>
<td>• P1DT12H:</td>
<td>Period of one day and twelve hours.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

curl "https://instance.servicenow.com/api/now/v1/clotho/transform \
--request POST \
--header "Accept:application/json", "Content-Type:application/json" \
--user "username":"password" \
-d '{"start": "PT15M",
"end": "",
"table": "mb_demo_drone",
"query": "fleet.nameSTARTSWITHB",
"limit": 5000,
"metrics": [
  {
    "metric": "mb_demo_mt_altitude",
    "label": "Series - Avg",
    "transforms": [
      {
        "name": "partition",
        "arg": {
          "aggregator": "AVG",
          "window": "PT5M"
        }
      },
      {
        "name": "avg"
      }
    ]
  },
  {
    "metric": "mb_demo_mt_altitude",
    "groupBy": "fleet"
  }
]'}
"label": "Fleet - AVG",
"transforms": [
  
  "name": "partition",
  "arg": {
    "aggregator": "AVG",
    "window": "PT5M"
  }
],

"name": "avg"

"results": [
  
  "marker": "674d86ba-a810-4065-942b-0b7ca2f95db2",
  "series": [
    
    "label": "Series - Avg",
    "values": [
      
      "timestamp": "2020-05-01T21:05:00Z",
      "value": 157.43086
    ],
    
    "timestamp": "2020-05-01T21:10:00Z",
    "value": 162.92278
  ]
],

"marker": "846aa334-232a-4015-b033-d18ebc4b1d23",
"grouped": [
  
  "groupingBy": "fleet",
  "groups": [
  
  ]
]
MetricBase Time Series - POST /now/clotho/put

Adds time series data to the MetricBase database.
**URL format**

Versioned URL: /api/now/{api_version}/clotho/put

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| sysparm_ignore_unknown_series              | Flag that indicates whether to ignore unknown series and continue the transaction without returning an error. Valid values:  
                                        | • true: Ignore unknown series.  
                                        | • false: Do not ignore unknown series.  
                                        | Default: true                                                                          |

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>seriesRef</td>
<td>Required. Information to update. Data type: Array</td>
</tr>
</tbody>
</table>
|           | "seriesRef": [  
|           | {  
|           | "metric": "String",  
|           | "subject": "String",  
|           | "table": "String"  
|           | }  
|           | ]  
| seriesRef.metric | Required. The name of the metric to update. |
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>seriesRef.subject</td>
<td>Required. The sys_id of the record in which to update the data. Located in the table specified in <code>seriesRef.table</code>. Data type: String</td>
</tr>
<tr>
<td>seriesRef.table</td>
<td>Required. The name of the table in which to save the data. Data type: String</td>
</tr>
<tr>
<td>values</td>
<td>Required. The series values to store. Data type: Array</td>
</tr>
<tr>
<td>values.timestamp</td>
<td>Required. The ISO 8601 timestamp of the value. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Format: <code>YYYY-MM-ddTHH:mm:ssZ</code></td>
</tr>
<tr>
<td></td>
<td>The ending ‘Z’, which denotes the UTC time zone in an ISO-formatted timestamp, is optional.</td>
</tr>
<tr>
<td>values.value</td>
<td>Required. The metric value. Data type: Number</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>
Request headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>Message that indicates the status of the request, such as ok if the request processed successfully.</td>
</tr>
</tbody>
</table>

Example: cURL request

```
curl "https://instance.servicenow.com/api/now/v1/clotho/put" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --data "{
```
Example: Python request

```python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/v1/clotho/put'
```

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
# Eg. User name="username", Password="password" for this sample
user = 'username'
pwd = 'password'

# Set proper headers
headers = {
    "Content-Type": "application/json",
    "Accept": "application/json"
}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd),
                          headers=headers, data="{
\"seriesRef\": {
    \"subject\": "3D666b051787333200a328c5b836cb0b92\",
    \"table\": "mb_demo_drone\",
    \"metric\": "mb_demo_mt_altitude"
},
\"values\": [
    {
      \"timestamp\": \"2019-03-21T17:05:00Z\",
      \"value\": 0.150185
    },
    {
      \"timestamp\": \"2019-03-21T17:06:00Z\",
      \"value\": 0.46074
    },
    {
      \"timestamp\": \"2019-03-21T17:07:00Z\",
      \"value\": 0.83104
    },
    {
      \"timestamp\": \"2019-03-21T17:08:00Z\",
      \"value\": 1.260635
    },
    {
      \"timestamp\": \"2019-03-21T17:09:00Z\",
      \"value\": 1.749
    }
  ]
"
})

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
exit()

# Decode JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "message": "ok"
    }
}

Predictive Intelligence API

Use the Predictive Intelligence API to predict a field value based on one or more input fields and a trained solution.

This API can only be used when the Predictive Intelligence (com.glide.platform_ml) plugin is activated.

Predictive Intelligence - GET /agent_intelligence/solution/{solution_name}/prediction

Predicts an output field value using a specific solution.

URL format

Default URL: /api/now/agent_intelligence/solution/{solution_name}/prediction

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>solution_name</td>
<td>Name of solution to use for predictions. For example, ml_incident_categorization.</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solution definition input field key-value pair</td>
<td>Name-value pair of the solution input field. For example, enter the name: short_description and the value: Unable to connect to VPN. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>Name-value pairs that were specified as input to the call. Data type: Object</td>
</tr>
<tr>
<td>output</td>
<td>Predicted outcome values based on the specified solution. Data type: Object</td>
</tr>
<tr>
<td>output.confidence</td>
<td>Estimated precision of the prediction as a percentage. For example, <strong>53.84615375762915</strong>. Data type: Number</td>
</tr>
<tr>
<td>output.outcome</td>
<td>Prediction output field value. For example, an incident categorization solution would return an incident category such as <strong>inquiry</strong>. Data type: String</td>
</tr>
<tr>
<td>output.threshold</td>
<td>Value of the configured threshold associated with the prediction. Data type: Number</td>
</tr>
</tbody>
</table>
Example: Sample cURL request

```bash
curl "https://instance.service-now.com/api/now/predictive_intelligence/solution/ml_incident_category/prediction?short_description=unable%20to%20connect%20to%20VPN" \
--request GET \
--header "Accept:application/json" \
--user "username":"password"
{
  "result": {
    "input": {
      "short_description": "unable to connect to VPN",
      "api": "api"
    },
    "output": {
      "outcome": "inquiry",
      "confidence": 53.84615375762915,
      "threshold": 5
    }
  }
}
```

Example: Sample Python request

```python
# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.service-now.com/api/now/agent_intelligence/solution/ml_incident_category/prediction?short_description=unable%20to%20connect%20to%20VPN'

# Eg. User name= "username", Password= "password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept': 'application/json'}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers )

# Check for HTTP codes other than 200
if response.status_code != 200:
```
print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',response.json())
exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "input": {
            "short_description": "unable to connect to VPN",
            "api": "api"
        },
        "output": {
            "outcome": "inquiry",
            "confidence": 53.84615375762915,
            "threshold": 5
        }
    }
}

Predictive Intelligence - GET /agent_intelligence/solution/prediction

Returns predictions for multiple solutions.

⚠️ Note: Outcome result objects are grouped by solution name and sys_id in the format result.<solutionname>.<sys_id>.[{ <result1> },{ <result2> }].

For customization information, see MLSolutionFactory scriptable objects.

URL format

Versioned URL: /now/{api_version}/agent_intelligence/solution/prediction

Default URL: /now/agent_intelligence/solution/prediction

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input_filter</td>
<td>Required parameter if <code>input_table</code> is used, do not use this parameter with <code>input_maps</code> parameter. Filter to select records on which to run predictions. For example: sys_id 0ef47232db801300864adfea5e961912</td>
</tr>
</tbody>
</table>
| input_maps  | Required unless using `input_table` parameter. Array of input name-value pairs. For example: `[["short_description": "my email is not working"], ["short_description": "need help with password"]]

Data type: Array |
| input_table  | Required unless using `input_maps` parameter. Table name on which you want to run predictions. For example: `incident` |
| options      | JSON object with optional arguments. For example: `{"top_n" : 5, "apply_threshold":false}` |

Valid options:
- `top_n`: Number. If provided, returns the top results, up to the specified number of predictions.
- `apply_threshold`: Boolean. Checks the threshold value for the solution and applies it to the result set. The threshold value is solution threshold for similarity or class-level threshold for classification. Default value is true.
- `custom_results_filter`: String. Similarity solutions only. Specifies the allowed set from which results are returned using an encoded query.
Query parameters (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>solution_names</td>
<td>Required. Comma-separated list of solution names for which you want to run predictions. For example: ml_incident_categorization,ml_incident_assignment</td>
</tr>
<tr>
<td>Data type:</td>
<td>String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>confidence</td>
<td>Value of the confidence associated with the prediction. For example, 53.84.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>detailedResults</td>
<td>Similarity solutions only. JSON key-value pair containing details about the matching text indices.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>predictedSysId</td>
<td>The sys_id of the predicted value. Results can be from any table on which information is being predicted.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>predictedValue</td>
<td>Value representing the prediction result.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>threshold</td>
<td>Value of the configured threshold associated with the prediction.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

```plaintext
curl "http://instance.servicenow.com/api/now/agent_intelligence/solution/prediction?input_table=incident&input_filter=sys_id%3D0ef47232db801300864adfea5e961912&solution_names=ml_incident_categorization%2Cml_incident_assignment&options=%7B%22top_n%22%3A2%2C%22apply_threshold%22%3Afalse%7D" 
```
--request GET \
--header "Accept:application/json" \
--user "username":"password"

{
"result": {
"ml_incident_categorization": {
"0ef47232db801300864adfea5e961912": [
{
"confidence": 29.12211732875455,
"threshold": 15,
"predictedValue": "Email",
"predictedSysId": ""
},
{
"confidence": 19.08583525847071,
"threshold": 14,
"predictedValue": "Platform Performance",
"predictedSysId": ""
}
]
},
"ml_incident_assignment": {
"0ef47232db801300864adfea5e961912": [
{
"confidence": 5.782322543467415,
"threshold": 5,
"predictedValue": "IT Finance CAB",
"predictedSysId": "5f63e48fc0a8010e00eaad81cd4dd37"
},
{
"confidence": 5.303589009246953,
"threshold": -1,
"predictedValue": "NY DB",
"predictedSysId": "5f74727dc0a8010e01efe33a251993f9"
}
]
}
}
Example: Sample Python request

```python
# Need to install requests package for python
import requests

# Set the request parameters
url =
'http://instance.servicenow.com/api/now/agent_intelligence/solution/prediction?input_table=incident&input_filter=sys_id%3D0ef47232db801300864adfe5e961912%26solution_names=ml_incident_categorization%2Cml_incident_assignment%26options%3D%7B%22top_n%22%3A%22%20%22%2C%22%20%22apply_threshold%22%3A%false%7D'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept': 'application/json'}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)
```

```json
{
    "result": {
        "ml_incident_categorization": {
            "0ef47232db801300864adfe5e961912": [
                {
                    "confidence": 29.12211732875455,
                    "threshold": 15,
                    "predictedValue": "Email",
                    "predictedSysId": ""
                },
                {
                    "confidence": 19.08583525847071,
                    "threshold": 14,
                    "predictedValue": "Platform Performance",
                    "predictedSysId": ""
                }
            ]
        }
    }
}
```
Product Order Open API

The Product Order Open API provides a standardized mechanism for placing a product order with all of the necessary order parameters. This API is a ServiceNow® implementation of TM Forum TMF622 Product Ordering Management API REST Specification.

A product order is created based on a product offering that is defined and published in a product catalog. The product offering identifies the product or set of products that are available to a customer and includes the relevant product characteristics that capture the unique options of a product, and other relevant attributes such as pricing, contract terms, and availability.

To access this API, the Order Management for Telecommunications (sn_ind_tmt_orm) plugin must be activated.

This API is provided within the sn_ind_tmt_orm namespace.

The calling user must have the sn_ind_tmt_orm.order_integrator role.
Extending the Product Order Open API

The Product Order Open API can be extended using the following script includes. The API should only be extended with an understanding of the consequences of the changes.

- **TMFOrderAPIConstants**: Contains constants and required parameter information.
- **TMFProductOrderAPIUtil**: Contains functions to handle POST requests.
- **TMFProductOrderGetAPIUtil**: Contains functions to handle GET requests.
- **ProductOrderExtensionOOB**: Contains helper functions that support functions in TMFProductOrderAPIUtil and TMFProductOrderGetAPIUtil.
- **ProductOrderProcessor**: An empty script include file. Use this file to define any functions that you want to override from ProductOrderExtensionOOB.

Extend the Product Order Open API to make the following customizations.

**Required parameters**

To change which request body parameters are required or not required, update the TMFOrderAPIConstants script include.

- **TMFOrderAPIConstants.SCHEMA.CREATE_PRODUCT_ORDER**: Sets the required request body parameters to create a product order.

**Request body validation**

To perform additional validation on the request body, override ProductOrderExtensionOOB functions. ProductOrderExtensionOOB contains the following four helper functions that return `true` by default.

- **validatePostRequest()** - Called by processCreateOrder() in TMFProductOrderAPIUtil.
- **validateProductObj()** - Called by processCreateOrder() in TMFProductOrderAPIUtil.
- **validateRelatedPartyObj()** - Called by processCreateOrder() in TMFProductOrderAPIUtil.
- **validateGetRequest()** - Called by processGetOrder() in TMFProductOrderGetAPIUtil.

If a helper function returns `false`, it stops the API operation. To apply custom validation, override ProductOrderExtensionOOB helper functions by creating functions with identical names and parameters in ProductOrderProcessor. These new ProductOrderProcessor functions will be called by
TMFProductOrderAPIUtil and TMFProductOrderGetAPIUtil to replace the default ProductOrderExtensionOOB helper functions. In this example, a function in ProductOrderProcessor overrides a default function in ProductOrderExtensionOOB to perform validation on the productOrderItem attribute.

```javascript
// ProductOrderProcessor
var ProductOrderProcessor = Class.create();
ProductOrderProcessor.prototype =
    Object.extendsObject(ProductOrderExtensionOOB, {
        // Define overriding functions here
        // Function name and parameters must be identical to the function it overrides
        validatePostRequest: function(orderObject, details) {
            // Returning false terminates the POST request
            // Make sure to push error message in details array in case of error
            if (gs.nil(orderObject.productOrderItem)) {
                details.push(new TMFCommonOrderAPIUtil().getErrorDetailsObj(TMFOrderAPIConstants.MESSAGES.MISSING_ORDER_ITEM, '/'));
                return false;
            }
            return true;
        },
        type: 'ProductOrderProcessor'
    });
```

**Additional REST operations**

To create additional operations beyond the existing GET and POST operations, create additional scripted REST resources for the Product Order Open API. The logic of the new scripted REST resources should be consistent with the existing operations. Define functions for the new operations in a new script include.

**Field mapping**

When creating records, the API maps request body parameters to record fields. When retrieving records, the API maps record fields to response object attributes. ProductOrderExtensionOOB contains the following functions to map a POST request body to a GlideRecord.

- `transformOrderGr()`
- `transformOrdLineItemGr()`
ProductOrderExtensionOOB contains the following functions to map a GlideRecord to a response object for GET or POST requests.

- transformPostOrderResponse()
- transformGetOrderResponse()
- transformProductObj()
- transformRelatedPartyCustomerLineItem()
- transformOrderItemRelationship()
- transformGetOrdLineItmResponse()
- transformProductCharacteristics()
- transformProductSpecification()

Customize field mappings to add and retrieve data for additional fields, or to change the default mappings for fields. To customize the field mappings, override ProductOrderExtensionOOB mapping functions by creating functions with identical names and parameters in ProductOrderProcessor. These new ProductOrderProcessor functions will be used by TMFProductOrderAPIUtil and TMFProductOrderGetAPIUtil to replace the default ProductOrderExtensionOOB mapping functions. In this example, two functions in ProductOrderProcessor override the default functions in ProductOrderExtensionOOB to create mappings for the external_id and sys_id fields.

```javascript
// ProductOrderProcessor
var ProductOrderProcessor = Class.create();
ProductOrderProcessor.prototype =
  Object.extendsObject(ProductOrderExtensionOOB, {
    // Define overriding functions here
    // Function name and parameters must be identical to the function it overrides

    transformOrderGr: function(requestObject, orderGr) {
      orderGr.external_id = requestObject.externalId;
      return orderGr;
    }
  });
```
Product Order Open API - GET /productorder/{id}

Retrieves the specified customer order.

It retrieves the order information from the following related tables:

- Customer Order [sn_ind_tmt_orm_order]
- Order Line Item [sn_ind_tmt_orm_order_line_item]
- Order Characteristic [sn_ind_tmt_orm_order_characteristic_value]
- Order Line Item Contact [sn_ind_tmt_orm_order_line_item_contact]

URL format

Default URL: /api/sn_ind_tmt_orm/productorder/{id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Sys_id of the customer order to retrieve. Located in the Customer Order [sn_ind_tmt_orm_order] table. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| fields| Comma-separated list of fields to return in the response. If this value is not passed in, all fields are returned. Possible values:  
  • channel  
  • expectedCompletionDate |
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td></td>
</tr>
<tr>
<td>note</td>
<td></td>
</tr>
<tr>
<td>productOrderItem</td>
<td></td>
</tr>
<tr>
<td>relatedParty</td>
<td></td>
</tr>
<tr>
<td>requestedCompletionDate</td>
<td></td>
</tr>
<tr>
<td>requestedStartDate</td>
<td></td>
</tr>
<tr>
<td>state</td>
<td></td>
</tr>
<tr>
<td>@type</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. Can be for any of the following reasons:</td>
</tr>
<tr>
<td></td>
<td>• Missing query parameter</td>
</tr>
<tr>
<td></td>
<td>• Invalid URI</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel</td>
<td>Array of objects that describe the channels to use for selling the products.</td>
</tr>
<tr>
<td>channel.id</td>
<td>Unique identifier of the channel to use to sell the associated products. ID values are located in the external_id field of the DistributionChannel table.</td>
</tr>
<tr>
<td>channel.name</td>
<td>Name of the channel to use to sell the associated products.</td>
</tr>
<tr>
<td>expectedCompletionDate</td>
<td>Date on which the order will be completed.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>note</td>
<td>Array of objects that describe notes made by the customer when ordering.</td>
</tr>
<tr>
<td>data type: Array</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>note.author</td>
<td>User name of the person that authored the associated note.</td>
</tr>
<tr>
<td>data type: String</td>
<td></td>
</tr>
<tr>
<td>note.date</td>
<td>Date that the note was created.</td>
</tr>
<tr>
<td>data type: String</td>
<td></td>
</tr>
<tr>
<td>note.text</td>
<td>Additional notes/comments made by the customer while ordering.</td>
</tr>
<tr>
<td>data type: String</td>
<td></td>
</tr>
<tr>
<td>ponr</td>
<td>Flag that indicates whether the point of no return for the order has been reached.</td>
</tr>
<tr>
<td>data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>productOrderItem</td>
<td>Array of objects that describe items associated with the product and their associate action.</td>
</tr>
<tr>
<td>data type: Array</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>productOrderItem.action</td>
<td>Action to carry out on the product. Possible actions are defined on the Choice List tab in the Action Dictionary Entry of the sn_ind_tmt_orm_order_line_item table. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.id</td>
<td>Unique identifier of the line item. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.itemPrice</td>
<td>Array of objects that describe the price associated with the product. Data type: Array</td>
</tr>
<tr>
<td>productOrderItem.itemPrice.price</td>
<td>Description of the price of the associated product. Data type: Object</td>
</tr>
<tr>
<td>productOrderItem.itemPrice.price.taxIncludedAmount</td>
<td>Description of the price of the associated product, including the tax. Data type: Object</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>productOrderItem.itemPrice.price.taxIncludedAmount.unit</td>
<td>Currency code in which the price is depicted. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.itemPrice.price.taxIncludedAmount.value</td>
<td>Price of product, including any tax. Data type: Number</td>
</tr>
<tr>
<td>productOrderItem.itemPrice.priceType</td>
<td>Specifies whether the price is recurring or non-recurring. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.itemPrice.recurringChargePeriod</td>
<td>If the price is recurring, specifies the recurring period, such as month. Data type: String</td>
</tr>
</tbody>
</table>
| productOrderItem.ponr                               | Flag that indicates whether the point of no return for the order has been reached. Valid values:  
  • true: The point of no return for the order has been reached.  
  • false: The point of no return for the order has not been reached. Data type: Boolean |
| productOrderItem.product                            | Description of the instance details of the product purchased by the customer. Data type: Object |

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>productOrderItem.product.id</td>
<td>Unique identifier of the product sold. Located in the <code>sn_ind_tmt_orm_external_id</code> field of the Product Inventory table. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.place</td>
<td>Maps of the locations on which to install the product. Data type: Object</td>
</tr>
<tr>
<td>productOrderItem.product.place.id</td>
<td>Sys_id of the associated record in the Location [cmn_location] table. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.place.@type</td>
<td>Part of TMF Open API standard. Annotation for order line item. This value is always <code>Place</code>. This information is not stored. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.productCharacteristic</td>
<td>Array of objects that describe the characteristics of the associated product. Only product characteristics whose values are different than the <code>previousValue</code> are returned. Data type: Array</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>productOrderItem.product.productCharacteristic.name</code></td>
<td>Name of the characteristic to associated with the product. Located in the Characteristic table. Data type: String</td>
</tr>
<tr>
<td><code>productOrderItem.product.productCharacteristic.previousValue</code></td>
<td>Previous characteristic option values if the update is for change order. The request is a change order if the <code>productOrderItem.action</code> parameter is other than <code>add</code>. For additional information on characteristic option values, see Product catalog data model. Data type: String</td>
</tr>
<tr>
<td><code>productOrderItem.product.productCharacteristic.value</code></td>
<td>Characteristic option values associated with the product. For additional information on characteristic option values, see Product catalog data model. Data type: String</td>
</tr>
<tr>
<td><code>productOrderItem.product.productSpecification</code></td>
<td>Description of the product specification associated with the product. Data type: Object</td>
</tr>
</tbody>
</table>
| `{"productSpecification": { "id": "String", "name": "String", 
@Column length exceeded | ```
<p>| <code>productOrderItem.product.productSpecification.name</code> | Name of the product specification to associate with the product. Located in the Product Specification table. Data type: String |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>productOrderItem.product.productSpecification.@type</td>
<td>Part of TMF Open API standard. Annotation for order line item contact. This value is always set to <code>ProductSpecificationRef</code>. This information is not stored. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.relatedParty</td>
<td>Array of objects that describe the party role linked to an OrderLineItemContact. Data type: Array</td>
</tr>
<tr>
<td>productOrderItem.product.relatedParty.email</td>
<td>Email address of the contact. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.relatedParty.firstName</td>
<td>First name of the contact. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.relatedParty.id</td>
<td>Sys_id of the account or customer contact associated with the order. Located in the account [customer_account] or contact [customer_contact] or Order line item contact [sn_ind_tmt_orm_order_line_item_contact] table. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.relatedParty.lastName</td>
<td>Last name of the contact. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.relatedParty.phone</td>
<td>Business phone number of the contact.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>productOrderItem.product.relatedParty.@referredType</td>
<td>Type of customer. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• OrderLineItemContact</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.relatedParty.@type</td>
<td>Part of TMF Open API standard. Annotation for order line item. This value is</td>
</tr>
<tr>
<td></td>
<td>always RelatedParty. This information is not stored.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>productOrderItem.productOffering</td>
<td>Description of the product offering associated with the product.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;productOffering&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>productOrderItem.productOffering.id</td>
<td>Sys_id of the product offering associated with the product. Located in the</td>
</tr>
<tr>
<td></td>
<td>Product Offering [sn_prd_pm_product_offering] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>productOrderItem.productOffering.name</td>
<td>Name of the product offering associated with the product. Located in the</td>
</tr>
<tr>
<td></td>
<td>Product Offering [sn_prd_pm_product_offering] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>productOrderItem.quantity</td>
<td>Number of product items purchased.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>productOrderItem.productOrderItemRelationship</td>
<td>Array of objects that describe the parent/child relationship between order</td>
</tr>
<tr>
<td></td>
<td>item and related order item.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>productOrderItem.productOrderItemRelationship.id</strong></td>
<td>Same as the <strong>productOrderItem.id</strong> value. Used for parent/child relationships. Data type: String</td>
</tr>
<tr>
<td><strong>productOrderItem.productOrderItemRelationship.relationshipType</strong></td>
<td>Type of relationship between the two line items. This information is used to identify the relationship hierarchy. Possible values: • HasChild • HasParent Data type: String</td>
</tr>
<tr>
<td><strong>productOrderItem.state</strong></td>
<td>State of the order of the associated product item. Data type: String</td>
</tr>
<tr>
<td><strong>productOrderItem.version</strong></td>
<td>The order version. Data type: String</td>
</tr>
<tr>
<td><strong>productOrderItem.@type</strong></td>
<td>Part of TMF Open API specification. Annotation for the product. It always is <strong>ProductOrderItem</strong> and is not stored. Data type: String</td>
</tr>
<tr>
<td><strong>relatedParty</strong></td>
<td>Array of objects that describe the party role linked to an entity. Possible entities include Customer or CustomerContact. Data type: Array</td>
</tr>
</tbody>
</table>

---

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relatedParty.id</td>
<td>Sys_id or external_id of the account or customer contact associated with the order. Located in either the account [customer_account] or contact [customer_contact] table. Data type: String</td>
</tr>
<tr>
<td>relatedParty.name</td>
<td>Name of the account or customer. Data type: String</td>
</tr>
</tbody>
</table>
| relatedParty.@referredType | Type of customer. Possible values:  
  • Customer  
  • CustomerContact  
  Data type: String                                                                 |
| relatedParty.@type       | Part of TMF Open API standard. Annotation for the product. This value is always RelatedParty. This information is not stored. Data type: String |
| requestedCompletionDate  | Delivery date requested by the customer. Data type: String                                                                                   |
| requestedStartDate       | Order start date requested by the customer. Data type: String                                                                                 |
| state                    | Current state of the order. Data type: String                                                                                                 |
| version                  | The order version. Data type: String                                                                                                          |
Name | Description
---|---
@type | Part of TMF Open API standard. Annotation for the product. This value is always `ProductOrder`. This information is not stored.

Data type: String

**Example: cURL request**

The following code example requests an existing customer order.

```bash
curl -X GET
  "https://servicenow-instance/api/sn_ind_tmt_orm/productorder/8d75939453126010a795ddee7f7b126a" \\
-H "Accept: application/json" \\
-u "username":"password" \\
```

**Response body.**

```json
{
  "id": "8d75939453126010a795ddee7f7b126a",
  "ponr": "false",
  "expectedCompletionDate": "2021-05-02T08:13:59.000Z",
  "requestedCompletionDate": "2021-05-02T08:13:59.000Z",
  "requestedStartDate": "2020-05-03T08:13:59.000Z",
  "channel": [
    {
      "id": "1",
      "name": "Agent Assist"
    }
  ],
  "note": [  
    {
      "author": "System Administrator",
      "date": "2021-02-25T14:22:07.000Z",
      "text": "This is a TMF product order illustration no 2"
    },
    {
      "author": "System Administrator",
      "date": "2021-02-25T14:22:06.000Z",
      "text": "This is a TMF product order illustration"
    }
  ],
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
"productOrderItem": [ 
  { 
    "id": "POI130",
    "ponr": "false",
    "quantity": 1,
    "action": "add",
    "itemPrice": [ 
      { 
        "priceType": "recurring",
        "price": { 
          "taxIncludedAmount": { 
            "unit": "USD",
            "value": 20
          }
        }
      },
      { 
        "priceType": "nonRecurring",
        "price": { 
          "taxIncludedAmount": { 
            "unit": "INR",
            "value": 0
          }
        }
      }
    ],
    "product": { 
      "@type": "Product",
      "productCharacteristic": [ 
        { 
          "name": "Security Type",
          "valueType": "Choice",
          "value": "Base",
          "previousValue": ""
        }
      ],
      "productSpecification": { 
        "id": "a6514bd3534560102f18ddee7f7b1247",
        "name": "SD-WAN Security",
        "@type": "ProductSpecificationRef"
      },
      "relatedParty": [ 
        { 
          "id": "417593945312601a795ddee7f7b127d",
        }
      ]
    }
  }
]
"name": "John Smith",
"email": "abc2@example.com",
"phone": "32456768",
"@type": "RelatedParty",
"@referredType": "OrderLineItemContact"
},
{
"id": "c175939453126010a795ddeeff7b127c",
"name": "Joe Doe",
"email": "abc0@example.com",
"phone": "1234567890",
"@type": "RelatedParty",
"@referredType": "OrderLineItemContact"
}
]
"productOffering": {
"id": "69017a0f536520103b6bddeeff7b127d",
"name": "Premium SD-WAN Offering"
},
"productOrderItemRelationship": [
{
"id": "POI100",
"relationshipType": "HasParent"
}
],
"state": "in_progress",
"version": "1",
"@type": "ProductOrderItem"
,
{
"id": "POI100",
"ponc": "false",
"quantity": 1,
"action": "add",
"itemPrice": [
{
"priceType": "recurring",
"price": {
"taxIncludedAmount": {
"unit": "INR",
"value": 0
}
}
}
"priceType": "nonRecurring",
"price": {
  "taxIncludedAmount": {
    "unit": "INR",
    "value": 0
  }
},
"product": {
  "@type": "Product",
  "productSpecification": {
    "id": "cfe5ef6a53702010cd6ddeeef7b12f6",
    "name": "SD-WAN Service Package",
    "@type": "ProductSpecificationRef"
  }
},
"productOffering": {
  "id": "69017a0f536520103b6bddeeef7b127d",
  "name": "Premium SD-WAN Offering"
},
"productOrderItemRelationship": [
  {
    "id": "POI130",
    "relationshipType": "HasChild"
  },
  {
    "id": "POI120",
    "relationshipType": "HasChild"
  },
  {
    "id": "POI110",
    "relationshipType": "HasChild"
  }
],
"state": "in_progress",
"version": "1",
"@type": "ProductOrderItem"
"quantity": 1,
"action": "add",
"itemPrice": [
{
  "priceType": "recurring",
  "price": {
    "taxIncludedAmount": {
      "unit": "USD",
      "value": 20
    }
  }
},
{
  "priceType": "nonRecurring",
  "price": {
    "taxIncludedAmount": {
      "unit": "INR",
      "value": 0
    }
  }
}
],
"product": {
  "@type": "Product",
  "productCharacteristic": [
  {
    "name": "CPE Type",
    "valueType": "Choice",
    "value": "Physical",
    "previousValue": ""
  },
  {
    "name": "WAN Optimization",
    "valueType": "Choice",
    "value": "Advance",
    "previousValue": ""
  },
  {
    "name": "Routing",
    "valueType": "Choice",
    "value": "Premium",
    "previousValue": ""
  }
]
"name": "CPE Model",
"valueType": "Choice",
"value": "ASR",
"previousValue": ""
}
],
"productSpecification": {
"id": "39b627aa53702010cd6dddeeff7b1202",
"name": "SD-WAN Edge Device",
"@type": "ProductSpecificationRef"
}
],
"productOffering": {
"id": "69017a0f536520103b6bddeeff7b127d",
"name": "Premium SD-WAN Offering"
},
"productOrderItemRelationship": [
{
"id": "POI100",
"relationshipType": "HasParent"
}
],
"state": "in_progress",
"version": "1",
"@type": "ProductOrderItem"
],
{
"id": "POI110",
"ponr": "false",
"quantity": 1,
"action": "add",
"itemPrice": [
{
"priceType": "recurring",
"price": {
"taxIncludedAmount": {
"unit": "INR",
"value": 0
}
}
},
{
"priceType": "nonRecurring",
"price": {
"taxIncludedAmount": {
  "unit": "USD",
  "value": 5
},
"product": {
  "@type": "Product",
  "productCharacteristic": [
    {
      "name": "Tenancy",
      "valueType": "Choice",
      "value": "Base (10 site)",
      "previousValue": ""
    }
  ],
  "productSpecification": {
    "id": "216663aa53702010cd6dddeeff7b12b5",
    "name": "SD-WAN Controller",
    "@type": "ProductSpecificationRef"
  },
  "place": {
    "id": "25ab9c4d8a0a0bb300f7dabdc0ca7c1c",
    "@type": "Place"
  }
},
"productOffering": {
  "id": "69017a0f536520103b6bddeeff7b127d",
  "name": "Premium SD-WAN Offering"
},
"productOrderItemRelationship": [
  {
    "id": "POI100",
    "relationshipType": "HasParent"
  }
],
"state": "in_progress",
"version": "1",
"@type": "ProductOrderItem"
},
"relatedParty": {
Product Order Open API - POST /productorder

Creates the specified customer order and customer order line items.

Once processed, new records are created in the following tables:

- Customer Order [sn_ind_tmt_orm_order]
- Order Line Item [sn_ind_tmt_orm_order_line_item]
- Order Characteristic [sn_ind_tmt_orm_order_characteristic_value]
- Order Line Item Contact [sn_ind_tmt_orm_order_line_item_contact]

URL format

Default URL: /api/sn_ind_tmt_orm/productorder

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
## Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel</td>
<td>Array of objects that describe the channels to use for selling the products.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;channel&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Table/field updated: sn_ind_tmt_orm_order</td>
</tr>
<tr>
<td>channel.id</td>
<td>Required. Unique identifier of the channel to use to sell the associated products.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Table/field updated: sn_ind_tmt_orm_order/external_id</td>
</tr>
<tr>
<td>channel.name</td>
<td>Name of the channel to use to sell the associated products. Possible channel names are defined on the Choice List tab in the Channel Dictionary Entry of the sn_ind_tmt_orm_order table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: Blank string</td>
</tr>
<tr>
<td>externalId</td>
<td>Unique identifier for the customer order. This value is determined by an external system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Table/field updated: sn_ind_tmt_orm_order/external_id</td>
</tr>
<tr>
<td></td>
<td>Default: Blank string</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>note</td>
<td>Array of objects that describe notes made by the customer when ordering.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td><strong>&quot;note&quot;:</strong> [</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;text&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td>Table/field updated: sn_ind_tmt_orm_order</td>
</tr>
<tr>
<td>note.text</td>
<td>Required. Additional notes/comments made by the customer while ordering.</td>
</tr>
<tr>
<td>Data type: String</td>
<td><strong>Table/field updated: sn_ind_tmt_orm_order/comments</strong></td>
</tr>
<tr>
<td></td>
<td>Default: Blank string</td>
</tr>
<tr>
<td>productOrderItem</td>
<td>Required. Array of objects that describe items associated with the product order and their associate action.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td><strong>&quot;productOrderItem&quot;:</strong> [</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;action&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;itemPrice&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;product&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;productOffering&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;productOrderItemRelationship&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;quantity&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>@type&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td>Table/field updated: sn_ind_tmt_orm_order_line_item</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>productOrderItem.action</code></td>
<td>Required. Action to carry out on the product. Possible actions are defined on the Choice List tab in the Entry of the <code>sn_ind_tmt_orm_order_line_item</code> table. Data type: String&lt;br&gt;Table/field updated: <code>sn_ind_tmt_orm_order_line_item/action</code>&lt;br&gt;Default: Blank string</td>
</tr>
<tr>
<td><code>productOrderItem.id</code></td>
<td>Required. Unique identifier of the line item. Data type: String&lt;br&gt;Table/field updated: <code>sn_ind_tmt_orm_order_line_item/external_id</code>&lt;br&gt;Maximum length: 40&lt;br&gt;Default: Blank string</td>
</tr>
</tbody>
</table>
| `productOrderItem.itemPrice` | Array of objects that describe the price associated with the product. Data type: Array<br>`"itemPrice": [<br>    {<br        "price": {Object},<br        "priceType": "String",
        "recurringChargePeriod": "String"
    }
]`<br>Table updated: `sn_ind_tmt_orm_order`<br>Default: Blank string |
<p>| <code>productOrderItem.itemPrice.price</code> | Description of the price of the associated product. Data type: Object |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>productOrderItem.itemPrice.price.taxIncludedAmount</code></td>
<td>Description of the price of the associated product, including the tax. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>Table updated: <code>sn_ind_tmt_orm_order_line_item</code></td>
</tr>
<tr>
<td></td>
<td>Default: Blank string</td>
</tr>
<tr>
<td><code>productOrderItem.itemPrice.price.taxIncludedAmount.unit</code></td>
<td>Currency code in which the price is depicted. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Table/field updated: <code>sn_ind_tmt_orm_order_line_item/mrc or nrc</code></td>
</tr>
<tr>
<td></td>
<td><code>productOrderItem.itemPrice.price.taxIncludedAmount.value</code></td>
</tr>
<tr>
<td></td>
<td>Table/field updated: <code>sn_ind_tmt_orm_order_line_item/mrc or nrc</code></td>
</tr>
<tr>
<td></td>
<td><code>productOrderItem.itemPrice.priceType</code></td>
</tr>
<tr>
<td></td>
<td><code>productOrderItem.itemPrice.recurringChargePeriod</code></td>
</tr>
<tr>
<td></td>
<td><code>productOrderItem.product</code></td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
|      | instance details of the product purchased by the customer.  
Data type: Object |
|      | "product:" {  
|      |   "id": "String",  
|      |   "place": {Object},  
|      |   "productCharacteristic": {Object},  
|      |   "productSpecification": {Object},  
|      |   "relatedParty": {Object},  
|      |   "@type": "String"  
|      | } |
|      | Table/field updated:  
|      | sn_ind_tmt_orm_order_line_item |
|      | productOrderItem.product.id  
Unique identifier of the product sold.  
Located in the sn_ind_tmt_orm_product_inventory_table.  
Data type: String  
Table/field updated:  
|      | sn_ind_tmt_orm_order_line_item/external_id  
Default: Blank string |
|      | productOrderItem.product.place  
Maps of the locations on which to install the product.  
Data type: Object |
|      | "place:" {  
|      |   "id": "String",  
|      |   "@type": "String"  
|      | } |
|      | Table/field updated:  
|      | sn_ind_tmt_orm_order_line_item |
|      | productOrderItem.product.place.id  
Required. Sys_id of the associated location record in the Location [cmn_location] table.  
Data type: String |
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table'field updated: sn_ind_tmt_or_m_order_line_item'</td>
<td></td>
</tr>
<tr>
<td>Default: Blank string</td>
<td></td>
</tr>
<tr>
<td>productOrderItem.product.place.@type</td>
<td>Part of TMF Open API standard for order line item contact. This value is always Place. This information is not stored. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.productCharacteristic</td>
<td>Array of objects that define characteristics of the associated product. Data type: Array</td>
</tr>
<tr>
<td>&quot;productCharacteristic&quot;:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;previousValue&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>productOrderItem.product.productCharacteristic.name</td>
<td>Name of the characteristic record associated with the product. Located in the Characteristic [sn_prd_pm_characteristic] table. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.productCharacteristic.previousValue</td>
<td>Previous characteristic option values if the update is for change order. The request is a change order if the productOrderItem.action parameter is other than add. For additional information on characteristic option values, see Product catalog data model. Default: Blank string</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>productOrderItem.product.productCharacteristic.value</td>
<td>Characteristic option values associated with the product. For additional information on characteristic option values, see Product catalog data model. Data type: String, Table/field updated: sn_ind_tmt_orm_order_characteristic_value/characteristic_option_value, Default: Blank string</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>productOrderItem.product.productSpecification.@type</td>
<td>Part of TMF Open API standard for order line item contact. This value is always <code>ProductSpecificationRef</code>. This information is not stored. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.relatedParty</td>
<td>Array of objects that describe the party role linked to an OrderLineItemContact. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;relatedParty&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;email&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;firstName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;lastName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;phone&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;@referredType&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;@type&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>productOrderItem.product.relatedParty.email</td>
<td>Email address of the contact. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.relatedParty.firstName</td>
<td>First name of the contact. Data type: String</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| productOrderItem.product.relatedParty.id | Sys_id of the account associated with the order in the account [customer_account], contact [customer_contact] or Order line item contact [sn_ind_tmt_orm_order_line_item_contact] table.  
  Data type: String  
  Table/field updated: sn_ind_tmt_orm_order_line_item_contact/relatedParty/id  
  Default: Blank string |
| productOrderItem.product.relatedParty.lastName | Last name of the contact.  
  Data type: String  
  Table/field updated: sn_ind_tmt_orm_order_line_item_contact/lastName  
  Default: Blank string |
| productOrderItem.product.relatedParty.phone | Business phone number of the contact.  
  Data type: String  
  Table/field updated: sn_ind_tmt_orm_order_line_item_contact/business_phone  
  Default: Blank string |
| productOrderItem.product.relatedParty.@referredType | Required. Type of customer.  
  Possible values:  
  • OrderLineItemContact  
  Data type: String |
| productOrderItem.product.relatedParty.@type | Part of TMF Open API standard. Annotation for order line item contact. This value is always RelatedParty. This information is not stored.  
  Data type: String |
# Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>productOrderItem.product.@type</td>
<td><strong>Part of TMF Open API standard. Annotation for the product. This value is always</strong>&lt;br&gt;<strong>Product. This information is not stored.</strong>&lt;br&gt;Data type: String</td>
</tr>
<tr>
<td>productOrderItem.productOffering</td>
<td><strong>Required. Description of the product offering associated with the product.</strong>&lt;br&gt;Data type: Object</td>
</tr>
<tr>
<td>productOrderItem.productOffering.id</td>
<td><strong>Required. Sys_id of the product offering associated with the product.</strong>&lt;br&gt;Located in the Product Offering [sn_prd_pm_product_offering] table.&lt;br&gt;Data type: String</td>
</tr>
<tr>
<td>productOrderItem.productOffering.name</td>
<td><strong>Name of the product offering associated with the product.</strong>&lt;br&gt;Located in the Product Offering [sn_prd_pm_product_offering] table.&lt;br&gt;Data type: String</td>
</tr>
<tr>
<td>productOrderItem.productOrderItem.quantity</td>
<td><strong>Number of the associated items.</strong>&lt;br&gt;Data type: Number&lt;br&gt;Table/field updated: sn_ind_tmt_orm_order_line_item.quantity&lt;br&gt;Default: null</td>
</tr>
</tbody>
</table>
| productOrderItem.productOrderItemRelationship            | **Required. Array of objects that describe the parent/child relationship between order items.**<br>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td><em>productOrderItemRelationship</em></td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;relationshipType&quot;: &quot;&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>productOrderItem.productOrderItemRelationship.id</td>
<td>Required. Same as the productOrderItem.id value. Used for parent/child relationship. Data type: String</td>
</tr>
<tr>
<td></td>
<td>Table/field updated:</td>
</tr>
<tr>
<td></td>
<td>sn_ind_tmt_orm_order_line_item</td>
</tr>
<tr>
<td></td>
<td>Default: Blank string</td>
</tr>
<tr>
<td>productOrderItem.productOrderItemRelationship.relationshipType</td>
<td>Required. Type of relationship between two line items. This information is used to identify a relationship hierarchy. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• HasChild</td>
</tr>
<tr>
<td></td>
<td>• HasParent</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>productOrderItem.quantity</td>
<td>Number of items ordered. Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Table/field updated:</td>
</tr>
<tr>
<td></td>
<td>sn_ind_tmt_orm_order.quantity</td>
</tr>
<tr>
<td></td>
<td>Default: Blank string</td>
</tr>
<tr>
<td>productOrderItem.@type</td>
<td>Part of TMF Open API standard. Annotation for the product. This value is always ProductOrderItem. This information is not stored.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relatedParty</td>
<td>Array of objects that describe the party role linked to an entity. Possible entities include Customer or CustomerContact.</td>
</tr>
<tr>
<td>relatedParty.id</td>
<td>Sys_id or external_id of the account or customer contact associated with the order. Located in either the account [customer_account] or contact [customer_contact] table.</td>
</tr>
<tr>
<td>relatedParty.name</td>
<td>Name of the account or customer.</td>
</tr>
<tr>
<td>relatedParty.@referredType</td>
<td>Required. Type of customer. Possible values: Customer, CustomerContact</td>
</tr>
<tr>
<td>relatedParty.@type</td>
<td>Part of TMF Open API standard. This value is always RelatedParty. This information is not stored.</td>
</tr>
<tr>
<td>requestedCompletionDate</td>
<td>Delivery date requested by the customer.</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Table/field updated: sn_ind_tmt_orm_order/expected_end_date</td>
</tr>
<tr>
<td></td>
<td>Default: Blank string</td>
</tr>
<tr>
<td>requestedStartDate</td>
<td>Order start date requested by the customer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Table/field updated: sn_ind_tmt_orm_order/expected_start_date</td>
</tr>
<tr>
<td></td>
<td>Default: Blank string</td>
</tr>
<tr>
<td>@type</td>
<td>Part of TMF Open API standard. For the product. Data type: String</td>
</tr>
<tr>
<td></td>
<td>This value is always ProductOrder. This information is not stored.</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json Or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json Or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
</table>
| 201         | Successful. If there are any issues with the characteristics or characteristics option information, the endpoint stores the following comments in the work notes fields of the associated Customer Order Line Item record:  
  • The following Order Item characteristics does not exist: Review specification `<characteristic.name>` and correct the characteristic and characteristic option in the order line item prior to approving the order.  
  • Order Item characteristic: `<characteristic.name>` with characteristic value: `<characteristic.value>` is invalid. Correct the characteristic values before approving the order. |
| 400         | Bad Request. Could be any of the following reasons:  
  • Invalid payload: Request body missing - Payload was not passed in the request body.  
  • Invalid payload: productOrderItem is missing - Product order line item object or JSON is missing.  
  • Invalid payload: productOrderItem id is missing - The `id` parameter is missing in the product order line item of the payload.  
  • Invalid payload: productOrderItem action is missing - The `action` parameter is missing in the product order line item of the payload.  
  • Invalid payload: productOrderItem productOffering is missing - The product offering object or JSON is missing from the product order line item in the payload.  
  • Invalid payload: productOffering id is missing - The `id` parameter is missing in the product order line item of the product offering object in the payload.  
  • Invalid payload: Product offering does not exist - The product offering in the product order line item is not valid. |
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Invalid payload: productOrderItem product is missing</td>
<td>The product object or JSON in the product order line item is missing from the payload.</td>
</tr>
<tr>
<td>• Invalid payload: product productSpecification is missing</td>
<td>The product specification object or JSON in the product order line item is missing from the payload.</td>
</tr>
<tr>
<td>• Invalid payload: productSpecification id is missing</td>
<td>The <code>id</code> parameter in the product order line item of the product specification object is missing from the payload.</td>
</tr>
<tr>
<td>• Invalid payload: Product specification does not exist</td>
<td>The product specification in the product order line item is not valid.</td>
</tr>
<tr>
<td>• Invalid payload: Product Inventory does not exist</td>
<td>In a change order (action = change), the quantity of an item is greater than what is in stock.</td>
</tr>
<tr>
<td>• Invalid payload: Product inventory ID is missing</td>
<td>In change order, the <code>product.id</code> is missing in the payload.</td>
</tr>
<tr>
<td>• Invalid payload: Sold Product is inactive</td>
<td>In a change order, a product specified in the payload is inactive.</td>
</tr>
<tr>
<td>• Invalid payload: relatedParty is missing</td>
<td>The related party object is missing from the payload.</td>
</tr>
<tr>
<td>• Invalid payload: Customer Account is missing</td>
<td>The related party customer object is missing from the payload.</td>
</tr>
<tr>
<td>• Invalid payload: Customer Account does not exist</td>
<td>The specified related party customer does not exist in the ServiceNow instance.</td>
</tr>
<tr>
<td>• Invalid payload: Order creation failed</td>
<td>Not able to create the requested order.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel</td>
<td>Array of objects that define the channels to use for selling. Data type: Array</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;channel:&quot;</td>
<td>&quot;channel:&quot; [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>channel.id</td>
<td>Unique identifier of the channel to use to sell the associated products.</td>
</tr>
<tr>
<td></td>
<td>Channel ID values are located in the external_id field of the Distribution</td>
</tr>
<tr>
<td></td>
<td>Channel table [sn_prd_pm_distribution_channel].</td>
</tr>
<tr>
<td>channel.name</td>
<td>Name of the channel to use to sell the associated products.</td>
</tr>
<tr>
<td>id</td>
<td>Sys_id of the customer order created for this request.</td>
</tr>
<tr>
<td>externalId</td>
<td>External identifier for the customer order, such as a purchase order.</td>
</tr>
<tr>
<td>note</td>
<td>Array of objects that describe additional notes made by the customer</td>
</tr>
<tr>
<td></td>
<td>while ordering.</td>
</tr>
<tr>
<td>note.text</td>
<td>Additional notes/comments made by the customer while ordering.</td>
</tr>
</tbody>
</table>

Data type: String

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>productOrderItem</td>
<td>Array of objects that describe items associated with the product and their associate action. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;productOrderItem&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;action&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;itemPrice&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;product&quot;: (Object),</td>
</tr>
<tr>
<td></td>
<td>&quot;productOffering&quot;: (Object),</td>
</tr>
<tr>
<td></td>
<td>&quot;productOrderItemReleationship&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;quantity&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;state&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;@type&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

| productOrderItem.action   | Action to carry out on the product. Possible actions are defined based on the Choice List tab in the Action Dictionary Entry of the sn_ind_tmt_orm_order_line_item table. Data type: String |
| productOrderItem.id       | Unique identifier of the line item. Data type: String                   |
| productOrderItem.itemPrice| Array of objects that describe the price associated with the product. Data type: Array |
|                           | "itemPrice": [                                                         |
|                           |   {                                                                     |
|                           |     "price": (Object),                                                 |
|                           |     "priceType": "String",                                            |
|                           |     "recurringChargePeriod": "String"                                  |
|                           | }                                                                       |

| productOrderItem.itemPrice.price | Description of the price of the associated product. |

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>productOrderItem.itemPrice.price.taxIncludeAmount</code></td>
<td>Description of the price of the associated product, including the tax.</td>
</tr>
<tr>
<td>Data type: Object</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;price&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;taxIncludedAmount&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;unit&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: Number</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><code>productOrderItem.itemPrice.price.taxIncludeAmount.unit</code></td>
<td>Currency code in which the price is depicted.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td><code>productOrderItem.itemPrice.price.taxIncludeAmount.value</code></td>
<td>Price of product, including any tax.</td>
</tr>
<tr>
<td>Data type: Number</td>
<td></td>
</tr>
<tr>
<td><code>productOrderItem.itemPrice.priceType</code></td>
<td>Specifies whether the price is recurring or non-recurring.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td><code>productOrderItem.itemPrice.recurringChargePeriod</code></td>
<td>If the price is recurring, specifies the recurring period, such as month.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td><code>productOrderItem.product</code></td>
<td>Description of the instance details of the product purchased by the customer.</td>
</tr>
<tr>
<td>Data type: Object</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;product&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;place&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;productCharacteristic&quot;:</td>
</tr>
<tr>
<td></td>
<td>&quot;productSpecification&quot;:</td>
</tr>
<tr>
<td></td>
<td>&quot;relatedParty&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;@type&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>productOrderItem.product.id</code></td>
<td>Unique identifier of the product sold. Located in the <code>sn_ind_tmt_orm_external_id</code> field of the Product Inventory <code>sn_ind_tmt_orm_product_inventory</code> table. Data type: String</td>
</tr>
<tr>
<td><code>productOrderItem.product.place</code></td>
<td>Maps of the locations on which to install the product. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;place&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;@type&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><code>productOrderItem.product.place.id</code></td>
<td>Sys_id of the associated location record in the Location <code>cmn_location</code> table. Data type: String</td>
</tr>
<tr>
<td><code>productOrderItem.product.productCharacteristic</code></td>
<td>Array of objects that describe the characteristics of the associated product. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;productCharacteristic&quot;:</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;previousValue&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><code>productOrderItem.product.productCharacteristic.name</code></td>
<td>Name of the characteristic to associate with the product. Located in the Characteristic <code>sn_prd_pm_characteristic</code> table. Data type: String</td>
</tr>
<tr>
<td><code>productOrderItem.product.productCharacteristic.previousValue</code></td>
<td>Previous characteristic option values if the update is for a change order. The request is a change order if the <code>productOrderItem.action</code> parameter is...</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>productOrderItem.product.productCharacteristic.value</td>
<td>Characteristic option values associated with the product. For additional information on characteristic option values, see Product catalog data model. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.productSpecification</td>
<td>Description of the product specification associated with the product. Data type: Object</td>
</tr>
<tr>
<td>productOrderItem.product.productSpecification.@type</td>
<td>Part of TMF Open API standard. Annotation for order line item contact. This value is always ProductSpecificationRef. This information is not stored. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.relatedParty</td>
<td>Array of objects that define the role linked to an OrderLineItem.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><strong>productOrderItem.product.relatedParty.email</strong></td>
<td>Email address of the contact.</td>
</tr>
<tr>
<td><strong>productOrderItem.product.relatedParty.firstName</strong></td>
<td>First name of the contact.</td>
</tr>
<tr>
<td><strong>productOrderItem.product.relatedParty.id</strong></td>
<td>Sys_id of the account or customer contact associated with the order. Located in the account [customer_account], contact [customer_contact] or Order line it item contact [sn_ind_tmt_orm_order_line_item_contact] table.</td>
</tr>
<tr>
<td><strong>productOrderItem.product.relatedParty.lastName</strong></td>
<td>Last name of the contact.</td>
</tr>
<tr>
<td><strong>productOrderItem.product.relatedParty.phone</strong></td>
<td>Business phone number of the contact.</td>
</tr>
<tr>
<td><strong>productOrderItem.product.relatedParty.@referredType</strong></td>
<td>Type of customer. Possible values: OrderLineItemContact</td>
</tr>
</tbody>
</table>

**Data type:** Array

```
"relatedParty": [  
  {   
    "email": "String",  
    "firstName": "String",  
    "id": "String",  
    "lastName": "String",  
    "phone": "String",  
    "@referredType": "String",  
    "@type": "String"  
  }  
]
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>productOrderItem.product.relatedParty.@type</td>
<td>Part of TMF Open API standard. Annotation for order line item contact. This value is always RelatedParty. This information is not stored. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.product.@type</td>
<td>Part of TMF Open API standard. Annotation for the product. This value is always Product. This information is not stored. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.productOffering</td>
<td>Description of the product offering associated with the product. Data type: Object</td>
</tr>
<tr>
<td>productOrderItem.productOffering.id</td>
<td>Sys_id of the product offering associated with the product. Located in the Product Offering [sn_prd_pm_product_offering] table. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.productOffering.name</td>
<td>Name of the product offering associated with the product. Located in the Product Offering [sn_prd_pm_product_offering] table. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.productOrderItemRelationship</td>
<td>Array of objects that describe the parent/child relationship between order items. Data type: Array</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>productOrderItem.productOrderItemRelationship.id</td>
<td>Same as the <code>productOrderItem.id</code>. Used for parent/child relationship. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.productOrderItemRelationship.relationshipType</td>
<td>Type of relationship between items. This information is used to identify relationship hierarchy. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.quantity</td>
<td>Number of the associated items to order. Data type: Number</td>
</tr>
<tr>
<td>productOrderItem.state</td>
<td>Current state of the product. This value is always <code>new</code>. Data type: String</td>
</tr>
<tr>
<td>productOrderItem.@type</td>
<td>Part of TMF Open API standard. Annotation for the product. This value is always <code>ProductOrderItem</code>. Not stored. Data type: String</td>
</tr>
<tr>
<td>relatedParty</td>
<td>Array of objects that define role linked to an entity. Possible entities include Customer or CustomerContact. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"relatedParty:
 |
 |
 |
 |
 |
 |
 |
 |
 |
 |
 ```

<p>| relatedParty.id | Sys_id or external_id of the account or customer contact associated with the order. Located in either the |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>account [customer_account] or [customer_contact] table.</td>
<td>Data type: String</td>
</tr>
<tr>
<td>relatedParty.name</td>
<td>Name of the account or customer.</td>
</tr>
<tr>
<td>relatedParty.type</td>
<td>Type of customer. Possible values: Customer, CustomerContact</td>
</tr>
<tr>
<td>requestedCompletionDate</td>
<td>Delivery date requested by the customer.</td>
</tr>
<tr>
<td>requestedStartDate</td>
<td>Order start date requested by the customer.</td>
</tr>
<tr>
<td>state</td>
<td>Current state of the order. For this endpoint, this value is always new.</td>
</tr>
<tr>
<td>@type</td>
<td>Part of TMF Open API standard. Annotation for the product. This value is always ProductOrder. This information is not stored.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

The following code example creates a customer order.

```
curl -X POST "https://servicenow-instance/api/sn_ind_tmt_orm/productorder" \
-H "Accept: application/json" \
-H "Content-Type: application/json" \
-u "username":"password" \ 
-d {"requestedCompletionDate": "2021-05-02T08:13:59.506Z", "requestedStartDate": "2020-05-03T08:13:59.506Z"},
```
"externalId": "PO-456",
"channel": [
{
   "id": "2",
   "name": "Online channel"
}
],
"note": [
{
   "text": "This is a TMF product order illustration"
},
{
   "text": "This is a TMF product order illustration no 2"
}
],
"productOrderItem": [
{
   "id": "POI100",
   "quantity": 1,
   "action": "change",
   "product": {
      "id": "fa6d13f45b5620102dff5e92dc81c77f",
      "@type": "Product",
      "productSpecification": {
         "id": "cfe5ef6a53702010cd6dddeeff7b12f6",
         "name": "SD-WAN Service Package",
         "@type": "ProductSpecificationRef"
      },
      "place": {
         "id": "25ab9c4d0a0bb300f7dabdc0ca7c1c",
         "@type": "Place"
      }
   },
   "productOffering": {
      "id": "69017a0f536520103b6bddeeff7b127d",
      "name": "Premium SD-WAN Offering"
   },
   "productOrderItemRelationship": [
      {
         "id": "POI120",
         "relationshipType": "HasChild"
      },
      {
         "id": "POI130",
         "relationshipType": "HasChild"
      }
   ]
}
"relationshipType": "HasChild"
}
}

"@type": "ProductOrderItem"
}

{
"id": "POI120",
"quantity": 1,
"action": "change",
"itemPrice": [ 
{
"priceType": "recurring",
"recurringChargePeriod": "month",
"price": { 
	"taxIncludedAmount": { 
		"unit": "USD",
		"value": 20
	}
	}
}
],
"product": { 
"id": "766d13f45b5620102dff5e92dc81c78a",
"@type": "Product",
"productCharacteristic": [ 
{
	"name": "WAN Optimization",
	"value": "Base",
	"previousValue": "Advance"
}
],
"productSpecification": { 
"id": "39b2d27aa53702010cd6dddeff7b1202",
"name": "SD-WAN Edge Device",
"@type": "ProductSpecificationRef"
},
"relatedParty": [ 
{
"id": "51670151c35420105252716b7d40dfe",
"firstName": "Joe",
"lastName": "Doe",
"email": "abc@example.com",
"phone": "1234567890",
"@type": "RelatedParty",
}]}
"@referredType": "OrderLineItemContact"
},
"place": {
"id": "25ab9c4d0a0a8a0bb300f7dabdc0ca7c1c",
"@type": "Place"
},
"productOffering": {
"id": "69017a0f536520103b6bddeeff7b127d",
"name": "Premium SD-WAN Offering"
},
"productOrderItemRelationship": [
{
"id": "POI100",
"relationshipType": "HasParent"
}
],
"@type": "ProductOrderItem"
},
{
"id": "POI130",
"quantity": 1,
"action": "add",
"itemPrice": [
{
"priceType": "recurring",
"recurringChargePeriod": "month",
"price": {
"taxIncludedAmount": {
"unit": "USD",
"value": 20
}
}
}
],
"product": {
"@type": "Product",
"productCharacteristic": [
{
"name": "Security Type",
"value": "Base",
"previousValue": "Advance"
}]}
Response body.

```json
{
  "requestedCompletionDate": "2021-05-02T08:13:59.506Z",
  "requestedStartDate": "2020-05-03T08:13:59.506Z",
  "externalId": "PO-456",
  "channel": [
    {
      "id": "2",
      "name": "Online channel"
    }
  ],
  "note": [
    {
      "text": "This is a TMF product order illustration"
    },
    {
      "text": "This is a TMF product order illustration no 2"
    }
  ],
  "productOrderItem": [
    {
      "id": "POI100",
      "quantity": 1,
      "action": "change",
      "product": {
        "id": "fa6d13f45b5620102dff5e92dc81c77f",
        "@type": "Product",
        "productSpecification": {
          "id": "cfe5ef6a53702010cd6dddeeff7b12f6",
          "name": "SD-WAN Service Package",
          "@type": "ProductSpecificationRef"
        },
        "place": {
          "id": "25ab9c4d0a0a0bb300f7dabdc0ca7c1c",
          "@type": "Place"
        }
      }
    }
  ]
}
```
"productOffering": {
  "id": "69017a0f536520103b6bddee7fb127d",
  "name": "Premium SD-WAN Offering"
},
"productOrderItemRelationship": [
  {
    "id": "POI120",
    "relationshipType": "HasChild"
  },
  {
    "id": "POI130",
    "relationshipType": "HasChild"
  }
],
"@type": "ProductOrderItem",
"state": "new"
],
{
  "id": "POI120",
  "quantity": 1,
  "action": "change",
  "itemPrice": [
    {
      "priceType": "recurring",
      "recurringChargePeriod": "month",
      "price": {
        "taxIncludedAmount": {
          "unit": "USD",
          "value": 20
        }
      }
    }
  ],
  "product": {
    "id": "766d13f45b5620102dff5e92dc81c78a",
    "@type": "Product",
    "productCharacteristic": [
      {
        "name": "WAN Optimization",
        "value": "Base",
        "previousValue": "Advance"
      }
    ]
  }
}
"taxIncludedAmount": {
  "unit": "USD",
  "value": 20
},
"product": {
  "@type": "Product",
  "productCharacteristic": [
    {
      "name": "Security Type",
      "value": "Base",
      "previousValue": "Advance"
    }
  ],
  "productSpecification": {
    "id": "a6514bd35345601018f18d3eff7b1247",
    "name": "SD-WAN Security",
    "@type": "ProductSpecificationRef"
  },
  "relatedParty": [
    {
      "id": "51667fb51c3542010525d4b3d0ddfe",
      "firstName": "Joe",
      "lastName": "Doe",
      "email": "abc@example.com",
      "phone": "1234567890",
      "@type": "RelatedParty",
      "@referredType": "OrderLineItemContact"
    }
  ],
  "place": {
    "id": "25ab9c4d0a0bb50f7dabdc0ca7c1c",
    "@type": "Place"
  }
},
"productOffering": {
  "id": "69017a0f536520103b6bddef7b127d",
  "name": "Premium SD-WAN Offering"
},
"productOrderItemRelationship": [
  {
    "id": "POI100",
    "relatedParty": [
      {
        "id": "51667fb51c3542010525d4b3d0ddfe",
        "firstName": "Joe",
        "lastName": "Doe",
        "email": "abc@example.com",
        "phone": "1234567890",
        "@type": "RelatedParty",
        "@referredType": "OrderLineItemContact"
      }
    ],
    "place": {
      "id": "25ab9c4d0a0bb50f7dabdc0ca7c1c",
      "@type": "Place"
    }
  }
]
"relationshipType": "HasParent"
},
"@type": "ProductOrderItem",
"state": "new"
}
],
"relatedParty": [
{
"id": "eaf68911c35420105252716b7d40ddde",
"name": "Sally Thomas",
"@type": "RelatedParty",
"@referredType": "CustomerContact"
},
{
"id": "ffc68911c35420105252716b7d40dd55",
"name": "Funco Intl",
"@type": "RelatedParty",
"@referredType": "Customer"
}
],
"@type": "ProductOrder",
"id": "6be0a925c3a220103e2e73ce3640ddfe",
"state": "new"
}

Push Installation API

The Push Installation API enables you to register and unregister the tokens that enable mobile devices to receive push notifications from an application.

A push notification is a text message that appears on a user’s mobile device to alert them about something important or to ask them to perform an action.

When creating custom mobile applications, in order to set up Push notifications, you must register the device with the manufacturer’s service. The service must know the address of a user’s device so it can properly send notifications to that device using push tokens. The device also needs to be able to receive the notifications and act on them. For instructions on how to register a device and get a push token, refer to the device’s operating systems developer documentation. For example, for Apple devices, refer to https://developer.apple.com/documentation/usernotifications/registering_your_app_with_apns.
Push Installation - POST /now/push/{pushApplicationName}/installation

Adds or updates tokens that enable devices to receive push notifications from the specified application.

**URL format**

Versioned URL: /api/now/{api_version}/push/{pushApplicationName}/installation

Default URL: /api/now/push/{pushApplicationName}/installation

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>pushApplicationName</td>
<td>Name of the application for which to register the push notifications. This parameter must be the same as the value in the Name column for the associated application in the Push Application [sys_push_application] table. Data type: String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Request body parameters (XML or JSON)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>platform</td>
<td>Platform on which to register the push notification. This parameter must be the Name value from the Push Platform [sys_push_platform] table. Data type: String</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>token</td>
<td>Token supplied by the device manufacturer, such as Apple, when the mobile application registers to receive push notifications. The endpoint writes the token value to the Push Notification Installation [sys_pushnotif_app_install] table. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Example: Sample cURL request

```bash
curl -X POST \
    https://instance.servicenow.com/api/now/push/ServiceNowPushApp/installation \
    --header "Content-Type: application/json" \
    --user "username":"password" \
    --data 
    '{"platform":"Apple","token":"544836d77ac2e551bbf78204630509a7d494fc87f52e85da52e5b5d42910e2cf"}' \
```

None

### Example: Sample Python request

```python
#Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/push/ServiceNowPushApp/installation'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {"Content-Type":"application/json"}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, 
data="{"platform":"Apple","token":"544836d77ac2e551bbf78204630509a7d494fc87f52e85da52e5b5d42910e2cf"}")
```
Push Installation - POST /now/push/{pushApplicationName}/removeInstallation
Deactivates the tokens that enable mobile devices to receive push notifications from the specified application.

**URL format**

Versioned URL: /api/now/{api_version}/push/{pushApplicationName}/removeInstallation
Default URL: /api/now/push/{pushApplicationName}/removeInstallation

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>pushApplicationName</td>
<td>Name of the application for which to unregister the push notifications. This parameter must be the same as the value in the Name column for the associated application in the Push Application [sys_push_application] table. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform</td>
<td>Platform on which to register the push notification. This parameter must be the Name value from the Push Platform [sys_push_platform] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. A logic error on the server-side code occurred.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Example: Sample cURL request

```
curl -X POST \  
   https://instance.servicenow.com/api/now/push/ServiceNowPushApp/removeInstallation \  
   --header "Content-Type: application/json" \  
   --user "username":"password" \  
   --data '{"platform":"Apple"}' \
```

Example: Sample Python request

```
# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/push/ServiceNowPushApp/removeInstallation'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Content-Type':'application/json'}

# Do the HTTP request
```
Push Metrics API

The Push Metrics API provides endpoints that enable integrators to push raw Operational Intelligence data from an external source to the MID Server.

This API is based on client-side tools that push the raw data from the external source to the MID Server. The Operational Intelligence Metrics extension is required to be configured with the Enable REST Listener option enabled. For more information on setting up the MID Server, see Get started with Operational Intelligence.

For information on authentication requirements, see Configure the MID Web Server extension.

Push Metrics - POST /mid/sa/metrics

Pushes raw Operational Intelligence data from an external source to the MID Server.

URL format

Default URL: /api/mid/sa/metrics

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ci2metric_id</td>
<td>List of key-value pairs to use to identify the configuration item (CI). It is formerly known as <code>ci_identifier</code>. The script continues to accept <code>ci_identifier</code>. Data type: Object</td>
</tr>
<tr>
<td>metric_type</td>
<td>Name of the metric. Data type: String</td>
</tr>
<tr>
<td>node</td>
<td>IP address, FQDN, name of the CI, or host. In the example below, the name of the Linux server where the disks are installed. Data type: String</td>
</tr>
<tr>
<td>resource</td>
<td>Information about the resource for which metric data is being collected. In the example below, <code>C:\</code> is the resource for which metric data is collected. Data type: String</td>
</tr>
<tr>
<td>source</td>
<td>Data source monitoring the metric type. Data type: String</td>
</tr>
<tr>
<td>timestamp</td>
<td>Epoch time stamp of the metric. Data type: Number, Unit: Milliseconds</td>
</tr>
<tr>
<td>value</td>
<td>Value of the metric. Data type: Number</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Example: Sample cURL request

```bash
curl -X POST "https://servicenow-instance/api/mid/sa/metrics"
-H "Content-Type: application/json"
-U "username":"password"
-d "{
   "metric_type": "Disk C: % Free Space",
}"
```
Remote help request API

The Remote help request API enables creating, viewing, and fetching lists of IT service requests and their details from an electronic medical record (EMR) system. An IT service request is associated with a task type such as an incident in the ServiceNow instance.

You can only use this API when the EMR Help application (sn_ind_rmt_help) is installed from the ServiceNow Store. Visit the ServiceNow Store website to view all the available apps and for information about submitting requests to the store. For cumulative release notes information for all released apps, see the ServiceNow Store version history release notes.

Role required to access the endpoints of this API: sn_ind_rmt_help.requester.

Remote help request - POST /remote_help_request/{req_defn_id}

Inserts electronic medical record (EMR) system data into the corresponding ServiceNow tables.

You need to specify a request definition ID in addition to the parameters for the task and additional data from the EMR system persisted in the request data table associated with the task. You pass two types of data into this endpoint. The first is the request data, which the endpoint inserts into the Remote Request Data [sn_ind_rmt_help_request_data] table and its associated child tables. The second is the task parameters of the IT service request, which the endpoint inserts into the Remote Request Parameter [sn_ind_rmt_help_request_param] table. Only data or fields that are defined in the request definitions are processed by the endpoint. For additional information on this data model, see EMR Help data model.

Records are identified using the request definition ID.

URL format

Versioned URL: /api/sn_ind_rmt_help/{api_version}/remote_help_request/{req_defn_id}
Default URL: /api/sn_ind_rmt_help/remote_help_request/{req_defn_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>req_defn_id</td>
<td>Unique ID of the request definition. This value corresponds to the ID column in the Remote Request Definition [sn_ind_rmt_help_request_defn] table. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_data</td>
<td>Required. Name-value pairs of the request data from the EMR system to add in the Remote Request Data [sn_ind_rmt_help_request_data] table and its child tables. For example:</td>
</tr>
</tbody>
</table>

```
"request_data":{
  "additional_info": "String",
  "application": "String",
  "environment": "String",
  "issue_type": "String",
  "millennium_username": "String",
  "position": "String",
  "server": "String",
  "session_recording_id": "String",
  "user_is_physician": "String",
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**Request body parameters (JSON) (continued)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;work_station&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>Only pass parameters configured in a request definition in the <strong>request_data</strong> object. Any other parameters are ignored. To learn more, see Configure request definitions for EMR systems.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>source</td>
<td>Required. Name of the EMR system invoking the endpoint as specified in a request definition.</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td>&quot;source&quot;: &quot;Cerner&quot;</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>task_parameters</td>
<td>Required. Name-value pairs that describe the task parameters of the IT service request. Each element in the object corresponds to a column in the corresponding task tables such as the Incident [incident] table.</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td>&quot;task_parameters&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;caller_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;contact_type&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;impact&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;short_description&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>Only pass parameters configured in a request definition in the <strong>task_parameters</strong> object. Any other parameters are ignored. To learn more, see Configure request definitions for EMR systems.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

<table>
<thead>
<tr>
<th>Request headers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Header</strong></td>
</tr>
<tr>
<td>Accept</td>
</tr>
<tr>
<td>Content-Type</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response headers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Header</strong></td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status codes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Status code</strong></td>
</tr>
<tr>
<td>200</td>
</tr>
<tr>
<td>401</td>
</tr>
<tr>
<td>500</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Result object.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result: {</td>
<td></td>
</tr>
<tr>
<td>&quot;code&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;error&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;errorTranslated&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;status&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;task_id&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;task_table&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;warning&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;warningTranslated&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>Data type: Object</td>
<td></td>
</tr>
<tr>
<td>result.code</td>
<td>Application error code when the <code>result.status</code> is returned as failure.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.error</td>
<td>Error message that is included if the <code>result.status</code> is failure.</td>
</tr>
<tr>
<td></td>
<td>This message is in English only.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.errorTranslated</td>
<td>Optional localized error message.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.status</td>
<td>Status of the response.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• success</td>
</tr>
<tr>
<td></td>
<td>• failure</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.task_id</td>
<td>Sys_id of the created task.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.task_table</td>
<td>Name of the task table in which the task was created as defined in the request definition.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.warning</td>
<td>Optional warning message. May be included if the <code>result.status</code> is success.</td>
</tr>
<tr>
<td></td>
<td>This message is in English only.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td><code>result.warningTranslated</code></td>
<td>Optional localized warning message.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

Insert data from a Cerner EMR system.

```bash
curl
  "https://instance.servicenow.com/api/sn_ind_rmt_help/v1/remote_help_request/sn_it_request"
--request POST \
--header "Accept: application/json" \
--header "Content-Type: application/json" \
--data "{
  "source":"Cerner",
  "task_parameters":{
    "short_description":"Unable to load patient data",
    "caller_id":"82d4ecb4db40e8100e28aa594b96195c",
    "impact":"2",
    "contact_type":"email"
  },
  "request_data":{
    "application":"Powerchart",
    "server":"CTXCHSITN453",
    "environment":"CTX24",
    "issue_type":"Helpdesk",
    "millennium_username":"JOHN JASON",
    "position":"Lab Tech",
    "session_recording_id":"s5ds34dd96491b959a35010651896k",
    "user_is_physician":"Yes",
    "work_station":"PC354FLR3STATION7",
    "additional_info":"MRN 222333"
  }
}
"--user "username":"password"
```

The following output shows both a successful response and an error response.

```json
// Successful response
{
  "result": {
    "task_id": "75b09061db2cac100e28aa594b9619fa",
    "error": null
  }
}
```
Remote help request - PUT /remote_help_request/{req_defn_id}/task/{task_id}

Updates a record in the specified remote request data table for the task.

Enables you to specify a request definition ID in addition to the task sys_id to update a remote request that was created earlier.

**URL format**

**Versioned URL:** /api/sn_ind_rmt_help/{api_version}/remote_help_request/{req_defn_id}/task/{task_id}

**Default URL:** /api/sn_ind_rmt_help/remote_help_request/{req_defn_id}/task/{task_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>req_defn_id</td>
<td>Unique ID of the request definition. This value corresponds to the ID column in the Remote Request Definition [sn_ind_rmt_help_request_defn] table.</td>
</tr>
</tbody>
</table>
### Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>task_id</td>
<td>Sys_id of the task to update. This field is provided by the EMR system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_data</td>
<td>Required. Name-value pairs of the request data from the EMR system to update the Remote Request Data [sn_ind_rmt_help_request_data] table or its child table. For example:</td>
</tr>
<tr>
<td></td>
<td>&quot;request_data&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;additional_info&quot; : &quot;Please contact my office for more information.&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>source</td>
<td>Required. Name of the EMR system invoking the endpoint as specified in a request definition. For example:</td>
</tr>
<tr>
<td></td>
<td>&quot;source&quot;: &quot;Cerner&quot;</td>
</tr>
</tbody>
</table>

Data type: Object

- **Note:** Only pass parameters configured in a request definition in the request_data object. Any other parameters are ignored. To learn more, see Configure request definitions for EMR systems.
Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| task_parameters  | Required. Name-value pairs that describe the task parameters of the IT service request. Each element in the object corresponds to a column in the corresponding task tables such as the Incident [incident] table. For example:  
```json
"task_parameters": {
  "impact":"1",
  "contact_type":"phone"
}
```

**Note:** Only pass parameters configured in a request definition in the task_parameters object. Any other parameters are ignored. To learn more, see Configure request definitions for EMR systems.

Data type: Object

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Result object.</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;code&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;errorTranslated&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;warning&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;warningTranslated&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>result.code</td>
<td>Application error code when the result.status is returned as failure.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.error</td>
<td>Error message that is included if the result.status is failure.</td>
</tr>
<tr>
<td></td>
<td>This message is in English only.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.errorTranslated</td>
<td>Optional localized error message.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>result.status</td>
<td>Status of the response. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• success</td>
</tr>
<tr>
<td></td>
<td>• failure</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.warning</td>
<td>Optional warning message. May be included if the result.status is success.</td>
</tr>
<tr>
<td></td>
<td>This message is in English only.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>result.warningTranslated</td>
<td>Optional localized warning message.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

**Example: cURL request**

Update an IT service request created earlier in a Cerner EMR system

```bash
curl
  "https://instance.servicenow.com/api/sn_ind_rmt_help/v1/remote_help_request/sn_it_request/task/207e57c1db60a410f50fd5b4b96192e"
  --request PUT \n  --header "Accept: application/json" \n  --header "Content-Type: application/json" \n  --data "{
    "source":"Cerner",
    "task_parameters":{
      "impact":"1",
      "contact_type":"phone"
    },
    "request_data":{
      "additional_info" : "Please contact my office for more information."
    }
  }
  --user "username":"password"
```

The following output shows both a successful response and an error response.

```json
// Successful response
{
```
Remote help request - GET /remote_help_request/{req_defn_id}
Fetches a list of tasks that match the specified ID and query filter.

⚠️ Note:
- Reference and choice type fields are always returned as JSON objects with the value and display_value.
- DateTime fields are returned as strings and always in UTC.

URL format

Versioned URL: /api/sn_ind_rmt_help/{api_version}/remote_help_request/{req_defn_id}
Default URL: /api/sn_ind_rmt_help/remote_help_request/{req_defn_id}

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>req_defn_id</td>
<td>Unique ID of the request definition to update. This value corresponds to the ID column in the Remote Request Definition [sn_ind_rmt_help_request_defn] table. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>Required. Encoded query used to filter the result set. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Return results.</td>
</tr>
<tr>
<td></td>
<td>&quot;result&quot;: {</td>
</tr>
</tbody>
</table>
|                     |   "code": "String",
|                     |   "error": "String",
|                     |   "errorTranslated": "String",
|                     |   "status": "String",
|                     |   "task_list": [Array],
|                     |   "warning": "String",
|                     |   "warningTranslated": "String" |
|                     | } |

Data type: Object

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.code</td>
<td>Application error code when the result.status is returned as failure.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.error</td>
<td>Error message that is included if the result.status is failure.</td>
</tr>
<tr>
<td></td>
<td>This message is in English only.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.errorTranslated</td>
<td>Optional localized error message.</td>
</tr>
<tr>
<td>result.status</td>
<td>Status of the response.</td>
</tr>
</tbody>
</table>
|               | Possible values:
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• failure</td>
</tr>
<tr>
<td></td>
<td>• success</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.task_list</td>
<td>List of tasks.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>result.warning</td>
<td>Optional warning message. May be included if the result.status is success.</td>
</tr>
<tr>
<td></td>
<td>This message is in English only.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.warningTranslated</td>
<td>Optional localized warning message.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

Fetch data from a Cerner EMR system

curl
  "https://instance.servicenow.com/api/remote_help_request/v1/sn_it_request/?query=active=true" \
  --header "Accept: application/json" \
  --header "Content-Type: application/json"
  
  { 
      "source":"Cerner",
  } \
  --user "username":"password"

The following output shows both a successful response and an error response.

```json
// Successful response
{
    "result": {
        "status": "success",
        "task_list": [
            {
                "number": "INC0010096",
                "short_description": "Unable to load data - 1",
                "assigned_to": {
                    "value": "7a381da2dbfb5410f50fdc5b4b9619f2",
                    "display_value": "Abel Tuter (IT agent)"
                }
            }
        ]
    }
}
```


// Error response
{
  "result": {
    "status": "failure",
    "code": "1019",
  }
}
Remote help request - GET /remote_help_request/{req_defn_id}/task/{task_id}

Fetches a single task as specified in the task_id.

Enables you to specify a request definition ID and task ID and retrieve the task details.

⚠️ Note:

- Reference and choice type fields are always returned as JSON objects with the value and display_value.
- DateTime fields are returned as strings and always in UTC.
- Only additional comments are supported in journal type fields.
- Additional comments are passed back as a JSON array of objects. Each object represents a comment with created_on, created_by, and value fields. The latest comments are sent first.

### URL format

**Versioned URL:** /api/sn_ind_rmt_help/{api_version}/remote_help_request/{req_defn_id}/task/{task_id}

**Default URL:** /api/sn_ind_rmt_help/remote_help_request/{req_defn_id}/task/{task_id}

### Supported request parameters

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>req_defn_id</td>
<td>Unique ID of the request definition. This value corresponds to the ID column in the Remote Request Definition [sn_ind_rmt_help_request_defn] table. Data type: String</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>task_id</td>
<td>Sys_id of the task to return. This field is provided by the EMR system.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Return results.</td>
</tr>
<tr>
<td>result.code</td>
<td>Application error code when the result.status is returned as failure.</td>
</tr>
<tr>
<td>result.error</td>
<td>Error message that is included if the result.status is failure. This message is in English only.</td>
</tr>
<tr>
<td>result.errorTranslated</td>
<td>Optional localized error message.</td>
</tr>
<tr>
<td>result.status</td>
<td>Status of the response. Possible values:</td>
</tr>
</tbody>
</table>

Data type: Object

```
"result": {
    "code": "String",
    "error": "String",
    "errorTranslated": "String",
    "status": "String",
    "task_parameters": (Object),
    "warning": "String",
    "warningTranslated": "String"
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• failure \n• success \nData type: String</td>
</tr>
<tr>
<td>result.task_parameters</td>
<td>Required. Name-value pairs that describe the task parameters of the IT service request. Each element in the object corresponds to a column in the corresponding task tables such as the Incident [incident] table. \nData type: Object</td>
</tr>
<tr>
<td>result.warning</td>
<td>Optional warning message. May be included if the result.status is success. \nThis message is in English only. \nData type: String</td>
</tr>
<tr>
<td>result.warningTranslated</td>
<td>Optional localized warning message. \nData type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

Fetch data from a Cerner EMR system

curl
   "https://instance.servicenow.com/api/sn_ind_rmt_help/v1/remote_help_request/sn_it_request/task/207e57c1db60a410f50f5b4b96192e"\n   --request GET \n   --header "Accept: application/json"\n   --user "username":"password"

The following output shows both a successful response and an error response.

```json
// Successful response
{
  "result": {
    "status": "success",
    "task_parameters": {
      "number": "INC0010096",
      "short_description": "Unable to load data - 1",
      "state": {
        "value": "2",
        "display_value": "In Progress"
      }
    }
  }
}
```
Response Templates API


This API requires the Templated Responses plugin (com.sn_templated_snip), which is activated by default. This API is provided within the sn_templated_snip namespace.

For additional information on response templates, see Response templates.

Response Templates - POST /response_templates/get_templates


URL format

Default URL: /api/sn_templated_snip/response_templates/get_templates
# Supported request parameters

## Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

## Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

## Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errorFormat</td>
<td>HTML formatting to use for errors. Data type: String Default: &quot;&lt;span style='color:#ff0000'&gt;${%s}&lt;/span&gt;&quot;</td>
</tr>
<tr>
<td>includeEvaluatedBody</td>
<td>Flag that indicates whether to render the template variables. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• false: Do not render the template variables. The response contains the message variables. For example:</td>
</tr>
<tr>
<td></td>
<td>Please note that your case ${number} has been escalated to ${assignment_group}.</td>
</tr>
<tr>
<td></td>
<td>• true: Renders the template variables and returns evaluated_response in the return results. For example:</td>
</tr>
<tr>
<td></td>
<td>Please note that your case INC100001 has been escalated to Facilities.</td>
</tr>
<tr>
<td>limit</td>
<td>Maximum number of response templates to return. Data type: Boolean Default: false</td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Default: 50</td>
</tr>
<tr>
<td>offset</td>
<td>For pagination, the index at which to start the search.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Default: 0</td>
</tr>
<tr>
<td>opts</td>
<td>Parameters to pass to the <code>sn_tempated_snip.response_template</code> extension point. The format and content of these parameters are dependent on the implementation of the extension point. For additional information on extension points, see <a href="#">Using extension points to extend application functionality</a>.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>recordId</td>
<td>Required. Sys_id of the record to use to render the variables in the response template.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>searchTerm</td>
<td>Text to use to filter the list of matching response templates. The endpoint performs a CONTAINS search of this text on the name and body fields and a STARTS WITH search on the short name field. For example, if the search term is &quot;crash&quot;, the endpoint returns any response template that matches the query criteria and has crash in the name or body or the short name starts with crash. Response templates with exact matches on short name appear first in the return results. All other returned response templates are sorted by name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Default: Return all matching response templates.</td>
</tr>
<tr>
<td>tableName</td>
<td>Required. Name of the table to use to search the Response Templates table to locate corresponding response templates. For example, &quot;incident&quot; or &quot;sn_hr_core_case&quot;.</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

**Status codes**

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Array of objects that list all of the templates that match the specified search criteria. Data type: Array</td>
</tr>
<tr>
<td>result.&lt;object&gt;</td>
<td>Response templates with exact matches on short name appear first in the return results. All other returned response templates are sorted by name. Data type: Object</td>
</tr>
</tbody>
</table>

```
"result": [  
  (Object) 
]
```

```
{  
  "evaluated_response": [Array],  
  "name": "String",  
}  
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **result.<object>.evaluated_response**
  - Array that lists the results of the template evaluation.
  - Data type: Array

```
  "elevated_response": [
    "error": [Array],
    "evaluated_body": "String",
    "success": Boolean
  ]
```

- **result.<object>.evaluated_response.error**
  - List of entries for each evaluation error that occurred.
  - Data type: Array

```
  "error": {
    "inAccessible": ...
  }
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.&lt;object&gt;.evaluated_response.error.inAccessibleVariables</td>
<td>Variables in the response template body that could not be resolved. Data type: String</td>
</tr>
<tr>
<td>result.&lt;object&gt;.evaluated_response.error.message</td>
<td>Error message. Data type: String</td>
</tr>
<tr>
<td>result.&lt;object&gt;.evaluated_response.error.unEvaluatedVariables</td>
<td>Variables in the response template body that were not evaluated. Data type: String</td>
</tr>
<tr>
<td>result.&lt;object&gt;.evaluated_response.evaluated_body</td>
<td>Evaluated response template body in HTML. Data type: String</td>
</tr>
<tr>
<td>result.&lt;object&gt;.evaluated_response.success</td>
<td>Flag that indicates whether all of</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>the variables rendered properly.</td>
<td>Possible values:</td>
</tr>
<tr>
<td>• true: All variables rendered properly.</td>
<td></td>
</tr>
<tr>
<td>• false: Variables did not render properly.</td>
<td></td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>result.&lt;object&gt;.name</td>
<td>Name of the response template.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.&lt;object&gt;.short_name</td>
<td>Short name of the response template.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.&lt;object&gt;.short_name_match</td>
<td>Flag that indicates whether an exact match occurred on the short name of the response template.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>result.&lt;object&gt;.sys_id</td>
<td>Sys_id of the response template.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>result.&lt;object&gt;.template_body</td>
<td>Response template body in HTML.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

curl
```
    "https://instance.service-now.com/api/sn_templated_snip/response_templates/get_templates"
\n--request POST \
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--data "{
  "tableName": "sn_hr_core_case_total_rewards",
  "recordId": "2c9e43320b30220097432da0d5673a9c"
}"
--user "username":"password"
```
```json
{
  "result": [
    
```
SAM-Software Usage Data Integration API

The SAM-Software Usage Data Integration API enables you to create and update records in the Software Usage [samp_sw_usage] table.

This API requires activation of the Software Asset Management Professional (com.snc.samp) plugin. It also requires that you have either the sam_admin or sam_developer role to access the endpoint.

SAM-Software Usage Data Integration - POST /sam_software_usage_integration/createOrUpdate

Creates or updates a specified record in the Software Usage [samp_sw_usage] table.

You can create/update one or more software usage records in a single call.

URL format

Versioned URL: /api/now/{api_version}/sam_software_usage_integration/createOrUpdate

Default URL: /api/now/sam_software_usage_integration/createOrUpdate
# Supported request parameters

## Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

## Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

## Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| items | List of column names and values for each field to store in the software usage record. The column information that you must pass depends on the value you specify for reclamation_type (which is a required field): Possible reclamation_type values and the column information that must be passed:  
  • total_usage:  
    ◦ configuration_item  
    ◦ last_used_time  
    ◦ month_used  
    ◦ norm_product  
    ◦ year  
  • last_used_date:  
    ◦ configuration_item  
    ◦ norm_product |
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The combination of these fields uniquely identify a record. If the record is not found in the table, a new record with the specified fields is created; otherwise the existing record is updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>405</td>
<td>Invalid method. The functionality is disabled.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td>Name-value pair list of all fields in the software usage record. Data type: Array (content varies)</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example illustrates how to create a new software usage record.

```bash
curl -X POST
    "https://instance.servicenow/api/now/sam_software_usage_integration/createOrUpdate"
    -H "Accept: application/json"
    -H "Content-Type: application/json"
    -d "{"items": [
        {
            'configuration_item': 'eaa9604d3790200044e0bfc8bcbe5dc1',
            'month_used': 2,
            'norm_product': '40209c060b3022002d6c650d37673a89',
            'norm_publisher': '24dccc20b3022002d6c650d37673ab3',
            'reclamation_type': 'total_usage',
            'usage_count': 13,
            'total_usage_time': 13,
            'year': 2020
        }
    ]}
```

{ "result": {} }
The following example illustrates how to update multiple software usage records in a single call.

curl -X POST
  "https://instance.servicenow/api/now/sam_software_usage_integration/createOrUpdate" \
  -H "Accept: application/json" \
  -H "Content-Type: application/json" \
  -d "{'items': [
    {
      'configuration_item': 'eaa9604d3790200044e0bfc8bcbe5dc1',
      'month_used': 2,
      'norm_product': '40209c060b3022002d6c650d37673a89',
      'norm_publisher': '24dcccc20b3022002d6c650d37673ab3',
      'reclamation_type': 'total_usage',
      'usage_count': 13,
      'total_usage_time': 13,
      'year': 2020,
      'sysId': '13d15fe59d511010f8772c637e39afe5',
      'status': "INSERT"
    },
    {
      'configuration_item': '0aa9e80d3790200044e0bfc8bcbe5da0',
      'month_used': 2,
      'norm_product': '40209c060b3022002d6c650d37673a89',
      'norm_publisher': '24dcccc20b3022002d6c650d37673ab3',
      'reclamation_type': 'total_usage',
      'usage_count': 13,
      'total_usage_time': 13,
      'year': 2020
    }
  ]}
"Ok"
"usage_count": 19,
'total_usage_time': 19,
'year': 2020,
},
{"configuration_item": 'a0a9a80d3790200044e0bfc8bcb6e5d3c',
'month_used': 2,
'norm_product': '40209c060b3022002d6c650d37673a89',
'norm_publisher': '24dcccc20b3022002d6c650d37673ab3',
'reclamation_type': 'total_usage',
'usage_count': 7,
'total_usage_time': 7,
'year': 2020},
{"configuration_item": '48a9280d3790200044e0bfc8bcb6e5d55',
'month_used': 2,
'norm_product': '40209c060b3022002d6c650d37673a89',
'norm_publisher': '24dcccc20b3022002d6c650d37673ab3',
'reclamation_type': 'total_usage',
'usage_count': 1,
'total_usage_time': 1,
'year': 2020},
{"configuration_item": '61a9680d3790200044e0bfc8bcb6e5d11',
'month_used': 2,
'norm_product': '40209c060b3022002d6c650d37673a89',
'norm_publisher': '24dcccc20b3022002d6c650d37673ab3',
'reclamation_type': 'total_usage',
'usage_count': 18,
'total_usage_time': 18,
'year': 2020}]}
Scorecards API

The Scorecards REST API enables you to query data about Performance Analytics indicators. The query parameters equate to the functions available on the Analytics Hub for drilling down into an indicator.

This API requires the Performance Analytics (com.snc.pa.analytics_center) application.

Scorecards - GET /now/pa/scorecards

Retrieves details about indicators from the Analytics Hub.

Access to tables through the REST API is restricted by BasicAuth. Access control lists defined for tables are enforced to restrict access to data.

To make queries using the Performance Analytics API, you must also have the pa_viewer role.
Almost all queries use the `sysparm_uuid` parameter. This parameter consists of the indicator `sys_id` followed by a colon-separated list of additional, optional `sys_ids`. The `sysparm_uuid` is always the first parameter.

**URL format**

Versioned URL: `/api/now/{api_version}/pa/scorecards`

Default URL: `/api/now/pa/scorecards`

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td><code>api_version</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td><code>sysparm_breakdown</code></td>
</tr>
<tr>
<td><code>sysparm_breakdown_relation</code></td>
</tr>
<tr>
<td><code>sysparm_contains</code></td>
</tr>
<tr>
<td><code>sysparm_display</code></td>
</tr>
</tbody>
</table>
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| sysparm_display_value            | Data retrieval operation for reference and choice fields. Based on this value, retrieves the display value and/or the actual value from the database. Data type: String Valid values:  
  - true: Returns the display values for all fields.  
  - false: Returns the actual values from the database.  
  - all: Returns both actual and display values.  
  Default: false |

**Note:** There is no preferred method for setting this parameter. However, specifying the display value may cause performance issues since it is not reading directly from the database and may include referencing other fields and records. For more information on display values and actual values, see [Table API FAQs (KB0534905)](https://www.servicenow.com/kb/KB0534905). |

| sysparm_elements_filter          | Sys_id of an elements filter to apply to the returned data. You cannot obtain the sys_id of an indicator group through this API. Instead, get the sys_id from the indicator group record. For more information about obtaining sys_ids from records, see [Unique record identifier (sys_id)](https://www.servicenow.com/kb/KB0534905). Data type: String |

| sysparm_exclude_reference_link   | Flag that indicates whether to hide additional information provided for reference fields, such as the URI to the reference resource. Valid values: |

---

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• true: Hide additional information provided for reference fields.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not hide additional information provided for reference fields.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Boolean <strong>Default:</strong> false</td>
</tr>
<tr>
<td>sysparm_favorites</td>
<td>Flag that indicates whether to return only indicators that are favorites of the querying user.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid values:</strong></td>
</tr>
<tr>
<td></td>
<td>• true: Return only indicators that are favorites of the querying user.</td>
</tr>
<tr>
<td></td>
<td>• false: Return all indicators.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Boolean <strong>Default:</strong> false</td>
</tr>
<tr>
<td>sysparm_from</td>
<td>Earliest date to return scores from. Only scores from this date or later are returned.</td>
</tr>
<tr>
<td></td>
<td>This parameter requires that <code>sysparm_include_scores</code> is set to true.</td>
</tr>
<tr>
<td></td>
<td><strong>Date type:</strong> String <strong>Format:</strong> ISO-8601 standard</td>
</tr>
<tr>
<td>sysparm_include_available_aggregates</td>
<td>Flag that indicates whether to return all available aggregates for an indicator when no aggregate has been applied.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid values:</strong></td>
</tr>
<tr>
<td></td>
<td>• true: Return all available aggregates for an indicator when no aggregate has been applied.</td>
</tr>
<tr>
<td></td>
<td>• false: Return no aggregates.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Boolean <strong>Default:</strong> false</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_include_aggregates</td>
<td>Flag that indicates whether to return all possible aggregates for an indicator, including aggregates that have already been applied.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Return all possible aggregates for an indicator, including aggregates that have already been applied.</td>
</tr>
<tr>
<td></td>
<td>• false: Return no aggregates.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>sysparm_include_available_breakdowns</td>
<td>Flag that indicates whether to return all available breakdowns for an indicator.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Return all available breakdowns for an indicator.</td>
</tr>
<tr>
<td></td>
<td>• false: Return no breakdowns.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>sysparm_include_forecast_scores</td>
<td>Flag that indicates whether to return the forecast_scores element that contains an array of date-value pairs that define the forecast data for the Analytics Hub.</td>
</tr>
<tr>
<td></td>
<td>This parameter requires that the sysparm_include_scores parameter is also set to true.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Return the forecast_scores element.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not return the forecast_scores element.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>sysparm_include_realtime</td>
<td>Flag that indicates whether to return the realtime_enabled element which indicates if real-time scores are enabled for the indicator, and the realtime_value element which contains the real-time score value. This parameter is not supported for formula indicators.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| sysparm_include_score_notes  | Flag that indicates whether to return all notes associated with the score. The note element contains the note text as well as the author and timestamp when the note was added. Valid values:  
  • true: Return all notes associated with the score.  
  • false: Do not return all notes associated with the score.  
  Data type: Boolean  
  Default: false |
| sysparm_include_scores       | Flag that indicates whether to return indicator scores for the entire time range selected on the Analytics Hub. If a value is not specified, this parameter defaults to false and returns only the most recent score value.  
  To constrain the date range of the scores that are returned, combine this parameter with `sysparm_from` and `sysparm_to`.  
  Valid values:  
  • true: Return indicator scores for the entire selected time range.  
  • false: Do not return indicator scores for the entire time range.  
  Data type: Boolean  
  Default: false |
| sysparm_include_target_color_scheme | Flag that indicates whether to return the target_color_scheme element that contains the minimum and maximum values, and the color of each section of the target color scheme for the Analytics Hub. Valid values:  
  • true: Return the realtime_enabled element.  
  • false: Do not return the realtime_enabled element.  
  Data type: Boolean  
  Default: false |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| sysparm_include_trendline_scores  | Flag that indicates whether to return the `trendline_scores` element that contains an array of date-value pairs that define the Analytics Hub trendline. This parameter requires that the `sysparm_include_scores` parameter is also set to `true`. Valid values:  
  - true: Return the `trendline_scores` element.  
  - false: Do not return the `trendline_scores` element.  
  Data type: Boolean  
  Default: false                                                                                                                                                                                                 |
| sysparm_key                       | Flag that indicates whether to return results only for key indicators. Valid values:  
  - true: Return the `trendline_scores` element.  
  - false: Do not return the `trendline_scores` element.  
  Data type: Boolean  
  Default: false                                                                                                                                                                                                                                                                 |
| sysparm_limit                     | Maximum number of scores to return.  
  Data type: Number                                                                                                                                                                                                                                                                                                                             |
| sysparm_page                      | Page number. For example, when querying 20 indicators with the default `sysparm_per_page` value (10), specify a `sysparm_page` value of 2 to retrieve indicators 11-20.  
  Data type: Number                                                                                                                                                                                                                                                                 |
<p>| sysparm_per_page                  | Maximum number of indicators each query can return on a page.                                                                                                                                                                                                                                                                                  |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_sortby</td>
<td>Value to use when sorting results. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• bullet</td>
</tr>
<tr>
<td></td>
<td>• change</td>
</tr>
<tr>
<td></td>
<td>• changeperc</td>
</tr>
<tr>
<td></td>
<td>• date</td>
</tr>
<tr>
<td></td>
<td>• default</td>
</tr>
<tr>
<td></td>
<td>• direction</td>
</tr>
<tr>
<td></td>
<td>• duedate</td>
</tr>
<tr>
<td></td>
<td>• frequency</td>
</tr>
<tr>
<td></td>
<td>• gap</td>
</tr>
<tr>
<td></td>
<td>• gapperc</td>
</tr>
<tr>
<td></td>
<td>• group</td>
</tr>
<tr>
<td></td>
<td>• indicator_group</td>
</tr>
<tr>
<td></td>
<td>• name</td>
</tr>
<tr>
<td></td>
<td>• order</td>
</tr>
<tr>
<td></td>
<td>• target</td>
</tr>
<tr>
<td></td>
<td>• trend</td>
</tr>
<tr>
<td></td>
<td>• value</td>
</tr>
<tr>
<td>sysparm_sortdir</td>
<td>Sort direction. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• asc: Denotes ascending</td>
</tr>
<tr>
<td></td>
<td>• des: Denotes descending</td>
</tr>
</tbody>
</table>

Data type: Number
Default: 10
Maximum: 100

Data type: String
Default: value
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_step</td>
<td>Numeric value to skip scores, based on the indicator frequency. For example, specify a value of 3 to return scores from every third day for a daily indicator, or every third week for a weekly indicator.</td>
</tr>
<tr>
<td>sysparm_tags</td>
<td>Indicator group sys_id in which to return the indicators of that group. Do not use <code>sysparm_uuid</code> with this parameter. You cannot obtain the sys_id of an indicator group through this API. Instead, get the sys_id from the indicator group record. For more information about obtaining sys_ids from records, see Unique record identifier (sys_id).</td>
</tr>
</tbody>
</table>
| sysparm_target   | Flag that indicates whether to return results only for indicators that have a target set on the Analytics Hub. Valid values:  
  - true: Only return results for indicators that have a target set.  
  - false: Return results for all applicable indicators. |
| sysparm_to       | Latest date from which to return scores. Only scores from this date or earlier are returned.  
  This parameter requires that `sysparm_include_scores` be `true`. |
| sysparm_uuid     | Colon-separated list of sys_id values to specify which indicators, breakdowns, aggregates, and domains to query. The parameter follows this format: `<indicator_sys_id>:<breakdown_sys_id>:<element_sys_ids>:<lvl-2 breakdown_sys_id>:<lvl-3 breakdown_sys_id>:...` |
Query parameters (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>elements_filter_sys_id or element_sys_ids:&lt;aggregate_sys_id&gt;;&lt;domain_sys_id&gt;</td>
<td>The parameter must begin with the sys_id of an indicator record. Optionally, you can append the sys_id values of a breakdown and breakdown element to group the response based on the breakdown, and the sys_id of an aggregate to apply that aggregate. You can use a breakdown aggregate, or use only one.</td>
</tr>
</tbody>
</table>

**Note**: If an indicator is configured to use a Default time series, all values that this API retrieves for that indicator use the selected aggregate.

For examples of fully-constructed sysparm_uuid values, see Performance Analytics REST API examples.

Data type: String

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Depends on request parameters.</td>
</tr>
</tbody>
</table>

Example: Return an indicator filtered by a breakdown and element

```bash
# The GET request gets the details for
# the indicator Number of open
# incidents, where the Priority of the
# incident is 1 - Critical. The sys_id
# values for the indicator, the
# Priority breakdown, and the 1 -
# Critical element are all passed in
# the sysparm_uuid parameter.

curl --verbose --user "username":"password" \
     --header "Accept: application/json" \
     "https://instance.servicenow.com/api/now/v1/pa/scorecards?sysparm_uuid=fb007202d7130100b96d45a3ce6103b4:0df47e02d7130100b96d45a3ce610399:e5900140200331007665978299a805f3"
```
{  
    // The request is made Dec 12. The score is 76, which is unchanged from Dec 11, and which exceeds the target of 70.
    "result": [  
        {  
            "indicator": {  
                "display_value": "Number of open incidents",
                "link": "https://instance.service-now.com/api/now/v1/table/pa_indicators/fbo07202d7130100b96d45a3ce6103b4",
                "value": "fb007202d7130100b96d45a3ce6103b4"
            },  
            "gap_formatted": "-6",
            "benchmarking": false,
            "frequency_label": "Daily",
            "changeperc_formatted": "0.0%",
            "direction_label": "Minimize",
            "precision": 0,
            "breakdown": {  
                "display_value": "Priority",
                "link": "https://instance.service-now.com/api/now/v1/table/pa_breakdowns/0df47e02d7130100b96d45a3ce610399",
                "value": "0df47e02d7130100b96d45a3ce610399"
            },  
            "personal_target": null,
            "description": "Number of incidents open based on resolved date is empty.",
            "value_color": "#ff8c00",
            "uuid": "fb007202d7130100b96d45a3ce6103b4:0df47e02d7130100b96d45a3ce610399:5f012106db5123003ee8f93baf9619bd",
            "frequency": 10,
            "gap": -6.0,
            "value_unit": "76",
            "indicator_frequency": 10,
            "value": 76.0,
            "indicator_aggregate": 1,
            "facts_table": {  
                "name": "incident",
                "label": "Incidents"
            }
        }
    ]
}
"key":false,
"indicator_frequency_label":"Daily",
"direction":2,
"element": { 
  "display_value":"1 - Critical",
  "link":"https://instance.service-now.com/api/now/v1/table/sys_choice/5f012106db5123003ee8f93baf9619bd",
  "value":"5f012106db5123003ee8f93baf9619bd"
},
"period_title":"Dec 12",
"period":"Dec 12",
"target_formatted":"70",
"change":0.0,
"gapperc_formatted":"-8.6%",

"query":"opened_atONToday@javascript:gs.beginningOfToday()@javascript:gs.endOfToday()^ORopened_at<javascript:gs.beginningOfToday()^resolved_atISEMPTY^ORresolved_at>javascript:gs.endOfToday()^state!=8^priority=1^EQ",
"realtime_enabled":true,
"changeperc":0.0,
"target":70.0,
"unit": { 
  "display_value":"#",
  "link":"https://instance.service-now.com/api/now/v1/table/pa_units/17b365e2d7320100ba986f14ce6103ad",
  "type":"formatted","value":"17b365e2d7320100ba986f14ce6103ad"
},
"value_formatted":"76",
"name":"Number of open incidents > Priority = 1 - Critical",
"gapperc":-0.08571428571428572,
"change_formatted":"0",
"favorite":true,
"personal_target_formatted":""
}
] ]
}

Example: Sample Python request

```python
import requests

url = "https://instance.servicenow.com/api/now/v1/pa/scorecards"
```
user = 'username'
pwd = 'password'

querystring =
{"sysparm_uuid":"fb007202d7130100b96d45a3ce6103b4:0df47e02d7130100b96d45a3ce610399:5f012106db5123003ee8f93baf9619bd"}

headers = {"Accept":"application/xml"}

response = requests.get(url, auth=(user,pwd), headers=headers, params=querystring)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <indicator>
      <display_value>Number of open incidents</display_value>
      <link>https://instance.service-now.com/api/now/v1/table/pa_indicators/fb007202d7130100b96d45a3ce6103b4</link>
      <value>fb007202d7130100b96d45a3ce6103b4</value>
    </indicator>
    <gap_formatted>-6</gap_formatted>
    <benchmarking>false</benchmarking>
    <frequency_label>Daily</frequency_label>
    <changeperc_formatted>0.0%</changeperc_formatted>
    <direction_label>Minimize</direction_label>
    <precision>0</precision>
    <breakdown>
      <display_value>Priority</display_value>
      <link>https://instance.service-now.com/api/now/v1/table/pa_breakdowns/0df47e02d7130100b96d45a3ce610399</link>
      <value>0df47e02d7130100b96d45a3ce610399</value>
    </breakdown>
  </result>
</response>
Script Debugger API

The Script Debugger API enables you to debug lines of server-side JavaScript code, such as business rules and script includes.

For more information about the Script Debugger, see Script Debugger and Session Log.

This API is available by default.

Script Debugger - GET /js/debugpoints/script/{tableName}/{sysId}/{fieldName}

Retrieve a list of breakpoints or logpoints in a server-side script, such as a business rule or script include.

URL format

Default URL: /api/now/js/debugpoints/script/{tableName}/{sysId}/{fieldName}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>Name of the table that contains the server-side script, such as sys_script or sys_script_include. Data type: String</td>
</tr>
<tr>
<td>sysId</td>
<td>Sys_id for the server-side script. Data type: String</td>
</tr>
<tr>
<td>fieldName</td>
<td>Name of the field that contains the server-side script content. Data type: String</td>
</tr>
</tbody>
</table>
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>403</td>
<td>User Not Authorized. The user who executed the request does not have permission to access breakpoint or log point information for the specified record.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| canWrite           | Flag indicating whether the currently logged in user can set or modify breakpoints for the script. Possible values:  
|                    | • true: You can set or modify breakpoints.  
|                    | • false: You cannot set or modify breakpoints.  
|                    | Data type: Boolean |
| debugpoints        | List of all the breakpoints and logpoints in the script.  
|                    | "debugpoints": {  
|                    |   "breakpoint": {Object}  
|                    |   "logpoint": {Object}  
|                    | }  
<p>| debugpoints.breakpoint | List of all the breakpoints in the |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>script, presented as key-value pairs. The key is the line number for the breakpoint. The value is an object containing the breakpoint's sys_id and text.</td>
</tr>
<tr>
<td></td>
<td>&quot;breakpoint&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&lt;line_number&gt;:</td>
</tr>
<tr>
<td></td>
<td>{Object}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>debugpoints.breakpoint.&lt;line_number&gt;</td>
<td>Line number for the breakpoint.</td>
</tr>
<tr>
<td></td>
<td>&lt;line_number&gt;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;evaluationString&quot;:</td>
</tr>
<tr>
<td></td>
<td>&quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;sysId&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>debugpoints.breakpoint.&lt;line_number&gt;.evaluationString</td>
<td>Text of the breakpoint. If the breakpoint is conditional, the evaluationString contains the condition text. If the breakpoint is not conditional, the evaluationString is an empty string.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>debugpoints.breakpoint.&lt;line_number&gt;.sysId</td>
<td>Sys_id of the breakpoint.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>debugpoints.logpoint</td>
<td>List of all the logpoints in the script, presented as key-value pairs. The key is the line number for the logpoint. The value is an object containing the logpoint's sys_id and text.</td>
</tr>
<tr>
<td></td>
<td>&quot;logpoint&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&lt;line_number&gt;: {</td>
</tr>
<tr>
<td></td>
<td>{Object}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>debugpoints.logpoint.&lt;line_number&gt;</td>
<td>Line number for the logpoint.</td>
</tr>
<tr>
<td></td>
<td>&lt;line_number&gt;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;evaluationString&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;sysId&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>debugpoints.logpoint.&lt;line_number&gt;.evaluationString</td>
<td>Text of the logpoint. Contains the message being logged.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>debugpoints.logpoint.&lt;line_number&gt;.sysId</td>
<td>Sys_id of the logpoint.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>key</td>
<td>Object providing details about the script being debugged.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>key</strong>:</td>
<td><strong>Object</strong></td>
</tr>
<tr>
<td><code>scriptField</code></td>
<td>Name of the field that contains the script.</td>
</tr>
<tr>
<td><code>scriptId</code></td>
<td>Sys_id of the script.</td>
</tr>
<tr>
<td><code>scriptType</code></td>
<td>Name of the table that contains the script.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>The path parameters used to make the request. Listed in the following order.</td>
</tr>
<tr>
<td></td>
<td>• <code>tableName</code></td>
</tr>
<tr>
<td></td>
<td>• <code>sysId</code></td>
</tr>
<tr>
<td></td>
<td>• <code>fieldName</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td>Name of the script.</td>
</tr>
<tr>
<td><code>script</code></td>
<td>Text displaying the script’s code.</td>
</tr>
</tbody>
</table>
Example cURL request

Example: cURL request

Retrieves a list of logpoints and breakpoints for a server-side script. This script has a logpoint on line 2, a breakpoint on line 11, and a conditional breakpoint on line 18.

curl
"https://instance.servicenow.com/api/now/js/debugpoints/script/sys_script_include/d65f78c40a0a0b6900196656f35913d3/script" \ 
--request GET \ 
--header "Accept:application/json" \ 
--header "Content-Type:application/json" \ 
--user 'username':'password"

{
    "result": {
        "canWrite": true,
        "debugpoints": {
            "LOGPOINT": {
                "2": {
                    "evaluationString": "A log message",
                    "sysId": "ba28b0fa739310101c233096fbf6a75e"
                }
            },
            "BREAKPOINT": {
                "11": {
                    "evaluationString": "",
                    "sysId": "dc5f5bf341256010f877587fbdf5ec1d"
                },
                "18": {
                    "evaluationString": "a == true",
                    "sysId": "3d4f5bf341256010f877587bd5e6f"e"
                }
            }
        }
    },
    "script": "// script code",
    "name": "AbstractAjaxProcessor",
    "key": {
        "scriptType": "sys_script_include",
        "scriptId": "d65f78c40a0b6900196656f35913d3",
        "scriptField": "script",
        "value": "sys_script_include.d65f78c40a0b6900196656f35913d3.script"
    }
}
Script Debugger - POST js/debugpoints/process

Add, update, or remove breakpoints or logpoints in a server-side script, such as a business rule or script include. Process several breakpoints or logpoints at a time.

URL format

Default URL: /api/now/js/debugpoints/process

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fetchAll</td>
<td>When true, returns a list of all the breakpoints or logpoints added or updated by the user. Data type: Boolean Default: false</td>
</tr>
</tbody>
</table>
| fetchAllFilter     | Returns a filtered list of all the breakpoints or logpoints added or updated by the user. Valid values:  
• debugpointType. Set to either breakpoint or logpoint.  
• scriptId. Set to the sys_id of the script in which to search for debug points.  
• scope. Set to the name of the scope in which to search for debug points.  
Use the caret (^) symbol as a separator for multiple properties. For example, fetchAllFilter=debugpointType=logpoint^scope=MyApp Data type: String |
### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>debugpointType</td>
<td>Required. Type of debug point to add, delete, or update.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• breakpoint</td>
</tr>
<tr>
<td></td>
<td>• logpoint</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>evaluationString</td>
<td>Text for a logpoint or conditional breakpoint.</td>
</tr>
<tr>
<td></td>
<td>For a logpoint, the <strong>evaluationString</strong> is the log message.</td>
</tr>
<tr>
<td></td>
<td>For a conditional breakpoint, the <strong>evaluationString</strong> is the condition.</td>
</tr>
<tr>
<td></td>
<td>For example, if you're looping through a list of user IDs and you want the</td>
</tr>
<tr>
<td></td>
<td>debugger to pause only when the user ID is 38493, you can add a conditional</td>
</tr>
<tr>
<td></td>
<td>breakpoint inside the loop with <code>userID == 38493</code> as the condition. The debugger will only pause at this breakpoint when the condition is true.</td>
</tr>
<tr>
<td></td>
<td>If the breakpoint is not a conditional breakpoint, the <strong>evaluationString</strong></td>
</tr>
<tr>
<td></td>
<td>is an empty string.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>fieldName</td>
<td>Required. Name of the field that contains the server-side script content.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>lineNumber</td>
<td>Line number in the server-side script specifying where to add, delete, or</td>
</tr>
<tr>
<td></td>
<td>update the breakpoint or logpoint.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>operation</td>
<td>Required. Operation to perform for the breakpoint or logpoint.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• add</td>
</tr>
<tr>
<td></td>
<td>• delete</td>
</tr>
<tr>
<td></td>
<td>• update</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>scriptId</td>
<td>Required. Sys_id for the server-side script.</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tableName</td>
<td>Required. Name of the table that contains the server-side script, such as sys_script or sys_script_include. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestedDebugpoints</td>
<td>List of the requested breakpoints and logpoints. Presented as an array of objects.</td>
</tr>
<tr>
<td></td>
<td>&quot;requestedDebugpoints&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;scriptId&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;evaluationString&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;operation&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;tableName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;fieldName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;status&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;lineNumber&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;debugpointType&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>requestedDebugpoints.ScriptId</td>
<td>Sys_id for the server-side script.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>requestedDebugpoints.evaluationString</td>
<td>Text added for a logpoint or conditional breakpoint. For a logpoint, the evaluationString is the log message. For a conditional breakpoint, the evaluationString is the condition.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>requestedDebugpoints.operation</td>
<td>Operation performed for the breakpoint or logpoint. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• add</td>
</tr>
<tr>
<td></td>
<td>• delete</td>
</tr>
<tr>
<td></td>
<td>• update</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>requestedDebugpoints.tableName</td>
<td>Name of the table that contains the server-side script, such as sys_script or sys_script_include. Data type: String</td>
</tr>
<tr>
<td>requestedDebugpoints.fieldName</td>
<td>Name of the field that contains the server-side script content. Data type: String</td>
</tr>
<tr>
<td>requestedDebugpoints.status</td>
<td>Result of the instruction to add, delete, or update a specific breakpoint or logpoint. Possible values: failure, ignored, success</td>
</tr>
<tr>
<td></td>
<td>A request to add, delete, or update a debug point is ignored in the following situations:</td>
</tr>
<tr>
<td></td>
<td>• It requests to delete a breakpoint or logpoint that doesn’t exist.</td>
</tr>
<tr>
<td></td>
<td>• It requests to add a breakpoint or logpoint identical to one that already exists.</td>
</tr>
<tr>
<td>requestedDebugpoints.lineNumber</td>
<td>Line number in the server-side script specifying where the breakpoint or logpoint was added, deleted, or updated. Data type: Number</td>
</tr>
<tr>
<td>requestedDebugpoints.debugpointType</td>
<td>Type of debug point added, deleted, or updated. Possible values: breakpoint, logpoint</td>
</tr>
</tbody>
</table>
Example: cURL request

This example adds a conditional breakpoint at line 12 in the script.

```
curl "https://instance.servicenow.com/api/now/js/debugpoints/process" \
  --request POST \
  --header "Accept: application/json" \
  --header "Content-Type: application/json" \
  --data "\n  {\n    "tableName": "sys_script_include",\n    "scriptId": "d65f78c40a0b6900196656f35913d3\",\n    "fieldname": "script\",\n    "lineNumber": 12,\n    "evaluationString": "a == false\",\n    "debugpointType": "breakpoint\",\n    "operation": "add\"\n  }\n  "\n  --user 'username': 'password'
```

```json
{
  "result": {
    "requestedDebugpoints": [
    {
      "scriptId": "d65f78c40a0b6900196656f35913d3\",\n      "evaluationString": "a == false\",\n      "operation": "add\",\n      "tableName": "sys_scriptInclude\",\n      "fieldName": "script\",\n      "status": "success\",\n      "lineNumber": 12,\n      "debugpointType": "breakpoint\"
    }],
    "status": 200
  }
}```
Service Catalog API

The Service Catalog API provides access Service Catalog configuration and actions from within a Service Portal.

Service Catalog - DELETE /sn_sc/servicecatalog/cart/{cart_item_id}
Deletes the specified item from the current cart.

URL format

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/cart/{cart_item_id}
Default URL: /api/sn_sc/servicecatalog/cart/{cart_item_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>cart_item_id</td>
<td>Sys_id of the item to delete from the current cart. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Indicates that the request completed successfully and the specified item was deleted from the cart.</td>
</tr>
<tr>
<td>400</td>
<td>Indicates that either the specified cart_item_id is invalid or the user does not have access to the cart item.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
### Example: cURL request
```
curl
  "https://instance.servicenow.com/api/sn_sc/v1/servicecatalog/cart/1f4cc597db21120064301150f0b8f5fc" \n  --request DELETE \n  --user "username":"password"
```

### Example: Python request
```
# Install requests package for python
import requests

# Set the request parameters
url =
  'https://instance.servicenow.com/api/sn_sc/v1/servicecatalog/cart/1f4cc597db21120064301150f0b8f5fc'

# Set the user credentials
user = 'username'
pwd = 'password'

# Make the HTTP request
response = requests.delete(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 204
if response.status_code != 204:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)
```

### Service Catalog - DELETE /sn_sc/servicecatalog/cart/{sys_id}/empty

Deletes a specified cart, and the contents of the cart.

The capabilities of what a user can delete (items and/or cart) depends on the role in which they use to authenticate.

User roles needed to delete cart and cart contents:
• admin/catalog_admin: Can empty any user cart.
• admin: Can delete the cart after all the contents of the cart are deleted.
• catalog_admin: Can delete all the items in the cart.

**URL format**

Versioned URL: `/api/sn_sc/{api_version}/servicecatalog/cart/{sys_id}/empty`
Default URL: `/api/sn_sc/servicecatalog/cart/{sys_id}/empty`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the cart to empty and delete. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Request completed successfully and the items in the cart are successfully checked out.</td>
</tr>
</tbody>
</table>
| 400          | Indicates an error for one of the following reasons:  
|              | • Invalid cartId.  
|              | • User does not have admin/catalog_admin role, or is trying to empty another user cart. |
| 401          | Unauthorized. The user credentials are incorrect or have not been passed. |
| 500          | Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error. |

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
**Example: cURL request**

curl
  "https://instance.servicenow.com/api/sn_sc/servicecatalog/cart/dde34ad713426300027879d96144b072/empty" \
  --request DELETE \ 
  --user 'username':'password'

None

**Example: Python request**

```python
# Install requests package for python
import requests

# Set the request parameters
url =
  'https://instance.servicenow.com/api/sn_sc/servicecatalog/cart/dde34ad713426300027879d96144b072/empty'

# Set the user credentials
user = 'username'
pwd = 'password'

# Make the HTTP request
response = requests.delete(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 204
if response.status_code != 204:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)
```

None

**Service Catalog - GET /sn_sc/servicecatalog/cart**

Retrieves the details of the items within the logged in user's cart.

**URL format**

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/cart
Default URL: /api/sn_sc/servicecatalog/cart

Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
## Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Indicates that the request completed successfully. If there is no item in the cart, it returns basic cart information and pricing for user. If the cart contains any items, then the information about the items and their individual pricing is also included.</td>
</tr>
<tr>
<td>400</td>
<td>Indicates that the cart is empty and cannot check out.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cart_id</td>
<td>Sys_id of the cart. Data type: String</td>
</tr>
</tbody>
</table>
| <recurring_frequency>   | List of items in the cart having a specific recurring frequency. Possible values include:  
  • daily  
  • monthly  
  • yearly  
  • none (no recurring fees)  
  Data type: Object  

```json
"<recurring_frequency>": {  
  "frequency_label": "String",  
  "items": [Array],  
  "show_subtotal_price": "String",  
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;subtotal_price&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;subtotal_recurring_frequency&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;subtotal_recurring_price&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;subtotal_title&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;total_title&quot;: &quot;String&quot;</td>
<td></td>
</tr>
</tbody>
</table>

**<recurring_frequency>.frequency_label**

Frequency of the associated recurring fee.

Data type: String

Default: Null

**<recurring_frequency>.items**

List of objects that contain details of each item associated with the specified recurring frequency category.

Data type: Array

```
"items": [
  {
    "cart_item_id": "String",
    "catalog_item_id": "String",
    "created_by": "String",
    "created_on": "String",
    "delivery_time": "String",
    "item_id": "String",
    "item_name": "String",
    "localized_price": "String",
    "localized_recurring_price": "String",
    "name": "String",
    "order_guide": "String",
    "picture": "String",
    "price": "String",
    "quantity": "String",
    "recurring_frequency": "String",
    "recurring_price": "String",
    "recurring_subtotal": Number,
    "recurring_subtotal_price": "String",
    "short_description": "String",
    "show_price": Boolean,
    "show_quantity": Boolean,
    "show_recurring_price": Boolean,
    "subtotal": Number,
    "subtotal_price": "String",
  }
]```
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.cart_item_id</code></td>
<td>Sys_id of the item in this cart. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.catalog_item_id</code></td>
<td>Sys_id of catalog associated the item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.created.by</code></td>
<td>User that initially placed the item in the cart. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.created_on</code></td>
<td>Date on which the item was initially placed in the cart. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.delivery_time</code></td>
<td>Time that the specified item takes to be delivered. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.item_id</code></td>
<td>Catalog sys_id of the item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.item_name</code></td>
<td>Name of the specified item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.localized_price</code></td>
<td>Price of the item in local currency. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.localized_recurring_price</code></td>
<td>Recurring fee for the specified item in local currency. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.name</code></td>
<td>Item name. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.order_guide</code></td>
<td>Name of the order guide to which this item is associated.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.picture</code></td>
<td>File name of the picture of the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.price</code></td>
<td>Price of the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.quantity</code></td>
<td>Number of the specified item in the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.recurring_frequency</code></td>
<td>Frequency in which the recurring fee is charged for the specified item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String, Default: Null</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.recurring_price</code></td>
<td>Recurring fee for the specified item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.recurring_subtotal</code></td>
<td>Subtotal of the recurring fees for the item (recurring_price x quantity).</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.recurring_subtotal_price</code></td>
<td>Subtotal of the recurring fees for the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.short_description</code></td>
<td>Short description of the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.show_price</code></td>
<td>Flag that indicates whether the price of the item is shown in the cart.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Show the price of the item in the cart.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not show the price of the item in the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.show_quantity</code></td>
<td>Flag that indicates whether to show the item quantity in the cart.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Show the item quantity in the cart.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not show the item quantity in the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.show_recurring_price</code></td>
<td>Flag that indicates whether the recurring fees should appear in the cart.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Show the recurring fees in the cart.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not show the recurring fees in the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.itemssubtotal</code></td>
<td>Subtotal number of items.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.itemssubtotal_price</code></td>
<td>Subtotal for the item (price x quantity).</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.sys_class_name</code></td>
<td>System classification name of the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.sys_id</code></td>
<td>Sys_id of the item record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.itemsupdated_by</code></td>
<td>User that updated the item after it was initially placed in the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.itemsupdated_on</code></td>
<td>Last date/time on which the item was either initially placed in the cart or</td>
</tr>
<tr>
<td></td>
<td>updated.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.variables</code></td>
<td>List of name/value pairs of the item variables. Data type: Object</td>
</tr>
</tbody>
</table>
| `<recurring_frequency>.show_subtotal_price` | Flag that indicates whether to show a subtotal for the associated recurring fee category. Possible values:  
  - true: Show subtotals in the cart.  
  - false: Do not show subtotals in the cart.  
  Data type: Boolean |
<p>| <code>&lt;recurring_frequency&gt;.subtotal_price</code> | Subtotal of the cost of all items for the associated recurring frequency category. Data type: String |
| <code>&lt;recurring_frequency&gt;.subtotal_recurring_frequency</code> | Type of subtotal recurring frequency, such as daily, monthly, etc. Same as <code>subtotal_recurring_frequency</code>. Data type: String |
| <code>&lt;recurring_frequency&gt;.subtotal_recurring_price</code> | Subtotal of the recurring fees of the items within the associated recurring frequency category. Data type: String |
| <code>&lt;recurring_frequency&gt;.subtotal_title</code> | If the <code>show_subtotal_price</code> is set to true, the text to display in the cart for the associated subtitle. Data type: String |
| <code>&lt;recurring_frequency&gt;.total_title</code> | Text to display for the total title for the associated recurring frequency category. Data type: String |
| <code>show_subtotal_price</code> | Flag that indicates whether to show the subtotal price in the cart. Possible values: |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• true: Show subtotals in the cart.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not show subtotals in the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>subtotal_price</td>
<td>Subtotal of all items in the cart in whole dollars.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>subtotal_recurring_frequency</td>
<td>Type of subtotal recurring frequency, such as daily, monthly, etc. Same as</td>
</tr>
<tr>
<td></td>
<td><code>&lt;recurring_frequency&gt;.subtotal_recurring_frequency</code>.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>subtotal_recurring_price</td>
<td>Subtotal of the recurring frequency fees for all items in the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>subtotal_title</td>
<td>Text to display in the cart for the subtotal field title.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>total_title</td>
<td>Text to display in the cart for the total field title.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

curl "https://instance.servicenow.com/api/sn_sc/servicecatalog/cart" \ 
    --request GET \ 
    --header "Accept:application/json" \ 
    --user "username":"password"

```json
{
    "result": {
        "cart_id": "6d88bc29137d2300027879d6144b04b",
        "subtotal_price": "$1,598.00",
        "subtotal_recurring_frequency": ",",
        "subtotal_recurring_price": "$0.00",
        "total_title": "Total",
        "none": {
```
Example: Python request

# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_sc/v1/servicecatalog/cart'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/json'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

Response body for an empty cart.

{  
"result": {
   "cart_id": "0830db8413a56300397533e2e144b0ba",
   "subtotal_price": "-",
   "subtotal_recurring_frequency": "",
   "subtotal_recurring_price": "$0.00",
   "total_title": "Total",
   "show_subtotal_price": "false",
   "subtotal_title": "Subtotal"
  }
}
Service Catalog - GET /sn_sc/servicecatalog/cart/delivery_address/{user_id}

Retrieves the shipping address of the specified user based on the glide.sc.req_for.roles property and the default behavior configured in the glide.sc.req_for.roles.defaultproperty.

URL format

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/cart/delivery_address/{user_id}

Default URL: /api/sn_sc/servicecatalog/cart/delivery_address/{user_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>user_id</td>
<td>Sys_id of the user whose shipping address is to be retrieved. Located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully; returns the shipping address of the specified user.</td>
</tr>
<tr>
<td>400</td>
<td>User sys_id is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Address of the specified user. Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

curl
   "https://instance.servicenow.com/api/sn_sc/servicecatalog/cart/delivery_address/62826bf03710200044e0bfc8bcbe5df1" \
   --request GET \
   --header "Accept:application/json" \
   --user "username":"password"

{
   'result': '25 New Chardon Street, Boston\n   Boston, MA, 02114-1801\n   USA'
}

Example: Python request

# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_sc/servicecatalog/cart/delivery_address/62826bf03710200044e0bfc8bcbe5df1'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept':"application/json"}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
   print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
   exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)
Service Catalog - GET /sn_sc/servicecatalog/catalogs

Retrieves a list of catalogs to which the user has access based on the passed in parameters.

**URL format**

**Versioned URL:** /api/sn_sc/{api_version}/servicecatalog/catalogs

**Default URL:** /api/sn_sc/servicecatalog/catalogs

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. For requests that exceed this number of records, use the sysparm_offset parameter to paginate record retrieval. This limit is applied before ACL evaluation. If no records return, including records you have access to, rearrange the record order so records you have access to return first.</td>
</tr>
<tr>
<td>sysparm_text</td>
<td>Keyword to search for in the &quot;title&quot; field of the available service catalogs.</td>
</tr>
</tbody>
</table>

[i] **Note:** Unusually large sysparm_limit values can impact system performance.

Data type: Number

Default: 1000
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_view</td>
<td>Device on which the item can display:</td>
</tr>
<tr>
<td></td>
<td>• desktop: On the desktop.</td>
</tr>
<tr>
<td></td>
<td>• mobile: On a mobile device.</td>
</tr>
<tr>
<td></td>
<td>• both: Both on a desktop and on a mobile device.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Indicates that the request completed successfully. If a valid query returns no results, the response body only contains an empty result array.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>Description of the catalog. Data type: String</td>
</tr>
<tr>
<td>desktop_image</td>
<td>Image that appears on the desktop for this catalog. Data type: String</td>
</tr>
<tr>
<td>has_categories</td>
<td>Flag that indicates whether the catalog contains categories. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Catalog contains categories.</td>
</tr>
<tr>
<td></td>
<td>• false: Catalog does not contain categories. Data type: Boolean</td>
</tr>
<tr>
<td>has_items</td>
<td>Flag that indicates whether the catalog contains items. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Catalog contains items.</td>
</tr>
<tr>
<td></td>
<td>• false: Catalog does not contain items. Data type: Boolean</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the catalog. Data type: String</td>
</tr>
<tr>
<td>title</td>
<td>Catalog title. Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

```bash
curl "https://instance.servicenow.com/api/sn_sc/servicecatalog/catalogs?sysparm_limit=10" \
--request GET \
--header "Accept:application/json" \
--user "username":"password"
```

```json
{
  "result": [
    {
      "title": "Technical Catalog",
      "sys_id": "742ce428d7211100f2d224837e61036d",
      "has_categories": true,
      "has_items": true,
      "description": "Products and services for the IT department",
      "desktop_image": "7a7c8271475211002ee987e8dee4906d.iix"
    },
    {
      "title": "Service Catalog",
      "sys_id": "e0d08b13c3330100c8b837659bba8fb4",
      "has_categories": true,
      "has_items": true,
      "description": "Service Catalog - IT Now",
      "desktop_image": "adbcc271475211002ee987e8dee49001.iix"
    }
  ]
}
```

Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_sc/servicecatalog/catalogs?sysparm_limit=10'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept':"application/json"}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)
```
# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
response.json())
exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": [
        {
            "title": "Technical Catalog",
            "sys_id": "742ce428d7211100f2d224837e61036d",
            "has_categories": true,
            "has_items": true,
            "description": "Products and services for the IT department",
            "desktop_image": "7a7c8271475211002ee987e8dee4906d.iix"
        },
        {
            "title": "Service Catalog",
            "sys_id": "e0d08b13c3330100c8b837659bba8fb4",
            "has_categories": true,
            "has_items": true,
            "description": "Service Catalog - IT Now",
            "desktop_image": "adbcc271475211002ee987e8dee49001.iix"
        }
    ]
}

**Service Catalog - GET /sn_sc/servicecatalog/catalogs/{sys_id}/categories**

Retrieves the list of available categories for the specified catalog.

**URL format**

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/catalogs/{sys_id}/categories

Default URL: /api/sn_sc/servicecatalog/catalogs/{sys_id}/categories
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the catalog whose categories are requested. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| sysparm_limit   | Maximum number of records to return. For requests that exceed this number of records, use the sysparm_offset parameter to paginate record retrieval.  
This limit is applied before ACL evaluation. If no records return, including records you have access to, rearrange the record order so records you have access to return first. 

⚠️ **Note:** Unusually large sysparm_limit values can impact system performance.  
Data type: Number  
Default: 1000 |
| sysparm_offset  | Starting record index for which to begin retrieving records. Use this value to paginate record retrieval.  
This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks.  
For example, the first time you call this endpoint, sysparm_offset is set to "0". To simply page through all available records, use sysparm_offset=sysparm_offset+sysparm_limit, until you reach the end of all records. |
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_offset</td>
<td>Do not pass a negative number in the <code>sysparm_offset</code> parameter. Data type: Number Default: 0</td>
</tr>
<tr>
<td>sysparm_top_level_only</td>
<td>Only return those categories whose parent is a catalog. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Only return those categories whose parent is a catalog.</td>
</tr>
<tr>
<td></td>
<td>• false: Return all categories.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>sysparm_view</td>
<td>UI view for which to render the data. Determines the fields returned in the response. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• desktop</td>
</tr>
<tr>
<td></td>
<td>• mobile</td>
</tr>
<tr>
<td></td>
<td>• both</td>
</tr>
<tr>
<td></td>
<td>If you also specify the <code>sysparm_fields</code> parameter, it takes precedent.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returns a list of categories in a catalog. If there are no query results, then an empty array is returned.</td>
</tr>
<tr>
<td>400</td>
<td>Indicates that the request or catalog sys_id is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>Number of items within the category. Data type: Number</td>
</tr>
<tr>
<td>description</td>
<td>Brief category description. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>full_description</td>
<td>Detailed category description. Data type: String</td>
</tr>
<tr>
<td>header_icon</td>
<td>Icon used on category headers. Data type: String</td>
</tr>
<tr>
<td>homepage_image</td>
<td>Image that appears for this category on the category's home page. Data type: String</td>
</tr>
<tr>
<td>icon</td>
<td>Icon to use to signify this category. Data type: String</td>
</tr>
<tr>
<td>subcategories</td>
<td>List of subcategories within the category. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;subcategories&quot;: [</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;title&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>subcategories.sys_id</td>
<td>Sys_id of the subcategory. Data type: String</td>
</tr>
<tr>
<td>subcategories.title</td>
<td>Title of the subcategory. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the category. Data type: String</td>
</tr>
<tr>
<td>title</td>
<td>Category title. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl "https://instance.servicenow.com/api/sn_sc/servicecatalog/catalogs/e0d08b13c3330100c8b837659bba8fb4/categories?sysparm_limit=2" --request GET --header "Accept:application/json" --user "username":"password"
```

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
```
url =
'https://instance.service-now.com/api/sn_sc/servicecatalog/catalogs/e0d08b13c3330100c8b837
659bba8fb4/categories'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Accept":"application/json"}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
response.json())
exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)
{
"result": [
{
"title": "Services",
"description": null,
"full_description": null,
"icon": "",
"header_icon": "",
"homepage_image": "",
"count": 2,
"subcategories": [],
"sys_id": "4cb69d19c3921200b0449f2974d3ae69"
},
{
"title": "Support",
"description": null,
"full_description": null,
"icon": "",
"header_icon": "",
"homepage_image": "",

6957

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.


Service Catalog - GET /sn_sc/servicecatalog/catalogs/{sys_id}
Retrieves the available information for a specified catalog.

URL format
Versioned URL: /api/sn_sc/{api_version}/servicecatalog/catalogs/{sys_id}
Default URL: /api/sn_sc/servicecatalog/catalogs/{sys_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the catalog to retrieve. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| sysparm_view | Device on which the item can display:  
  - desktop: On the desktop.  
  - mobile: On a mobile device.  
  - both: Both on a desktop and on a mobile device.  
  Data type: String |
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Indicates that the request has completed successfully and returns the available information for the specified catalog.</td>
</tr>
</tbody>
</table>
| 400         | Indicates that request is invalid for one of the following reasons:  

- User does not have access to the catalog.  
- Invalid sys_id.  

401 Unauthorized. The user credentials are incorrect or have not been passed. |
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>categories</td>
<td>List of objects that describe categories within the catalog.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;categories&quot;: [</td>
</tr>
<tr>
<td>categories.description</td>
<td>Description of the category.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>categories.header_image</td>
<td>Category header image.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>categories.sys_id</td>
<td>Unique category identifier (sys_id).</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>categories.title</td>
<td>Category title.</td>
</tr>
<tr>
<td>description</td>
<td>Catalog description.</td>
</tr>
<tr>
<td>desktop_image</td>
<td>Source of the image that displays with the catalog in the desktop view.</td>
</tr>
<tr>
<td>has_categories</td>
<td>Flag that indicates whether the catalog has categories.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>Description</td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Catalog has categories.</td>
</tr>
<tr>
<td></td>
<td>• false: Catalog does not have categories.</td>
</tr>
<tr>
<td>Data type</td>
<td>Boolean</td>
</tr>
<tr>
<td>has_items</td>
<td>Flag that indicates whether there are items within the category.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Catalog has items associated with it.</td>
</tr>
<tr>
<td></td>
<td>• false: Catalog does not have any items associated with it.</td>
</tr>
<tr>
<td>Data type</td>
<td>Boolean</td>
</tr>
<tr>
<td>sys_id</td>
<td>Unique catalog sys_id.</td>
</tr>
<tr>
<td>title</td>
<td>Catalog title.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl "https://instance.servicenow.comaqsw" \
  --request GET \
  --header "Accept:application/json" \
  --user "username":"password"
```

```json
{
  "result": {
    "title": "Customer Service",
    "sys_id": "65bcd377c3011200b12d9f2974d3ae0",
    "has_categories": true,
    "categories": [
      {
        "header_image": "",
        "sys_id": "4cb69d19c3921200b0449f2974d3ae69",
        "description": null,
        "title": "Services"
      },
      {
        "header_image": "",
      }
  }
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_sc/servicecatalog/catalogs/e0d08b13c3330100c8b837659bba8fb4'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
    <result>
        <title>Service Catalog</title>
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Service Catalog - GET /sn_sc/servicecatalog/categories/{sys_id}
Retrieves the available information for a specified category.

URL format

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/categories/{sys_id}
Default URL: /api/sn_sc/servicecatalog/categories/{sys_id}

Supported request parameters

| Path parameters |
|-----------------|---------------------------------------------------------------|
| Name            | Description                                                                 |
| api_version     | Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String |
| sys_id          | Sys_id of the category for which to return information. Data type: String |
Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_view</td>
<td>UI view for which to render the data. Determines the fields returned in the response. Valid values: • desktop • mobile • both</td>
</tr>
<tr>
<td></td>
<td>If you also specify the <code>sysparm_fields</code> parameter, it takes precedent. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
| 400         | Indicates that request is invalid for one of the following reasons:  
  - User does not have access to the category.  
  - Invalid sys_id. |
| 401         | Unauthorized. The user credentials are incorrect or have not been passed. |
| 500         | Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error. |

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| child_categories          | List of objects that describe the child categories that belong to this category.  
  Data type: Array          |
|                           |                                                                                          |
|                           | "child_categories": [  
|                           |   "count": Number,  
|                           |   "isParentCategory": "String",  
|                           |   "parent": "String",  
|                           |   "subcategories": [Array],  
|                           |   "sys_id": "String",  
|                           |   "title": "String"  
|                           | ]                                                                                     |
| child_categories.count    | Number of items in the child category.  
  Data type: Number         |
| child_categories.isParentCategory | Flag that indicates if this is a parent category.  
  Possible values:           |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
|                                  | • true: Parent category.  
• false: Not the parent category.  
Data type: Boolean               |
| child_categories.parent          | Sys_id of the child category's parent.  
Data type: String                |
| child_categories.subcategories   | List of objects that describe the subcategories for this child category.   
Data type: Array                  |
|                                  | "subcategories": [  
  "sys_id": "String",  
  "title": "String"  
]                         |
| child_categories.subcategories.sys_id | Sys_id of the child subcategory.  
Data type: String                |
| child_categories.subcategories.title | Child subcategory title.  
Data type: String                |
| child_categories.sys_id          | Sys_id of the child category.  
Data type: String                |
| child_categories.title           | Child category title.  
Data type: String                |
| description                      | Short category description.  
Data type: String                |
| full_description                 | Detailed category description.  
Data type: String                |
| header_icon                      | Source of the icon displayed beside the category header when the category is listed at the top-level category.  
Data type: String                |
<p>| homepage_image                   | Source of the home page image for the category.                            |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>icon</td>
<td>Source of the small icon displayed beside the category name when the category is listed as a subcategory. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the category. Data type: String</td>
</tr>
<tr>
<td>title</td>
<td>Category title. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

curl

```
"https://instance.servicenow.com/api/sn_sc/servicecatalog/categories/b0fd6b01932002009ca87
a75e57ffbe9" \ 
--request GET \ 
--header "Accept:application/json" \ 
--user "username":"password"
```

```json
{
"result": {
"title": "Office",
"description": "Office services such as printing, supplies requisition\n\nt\nt\nand document shipping and delivery.\n\nt\nt",
"full_description": null,
"icon": "",
"header_icon": "",
"homepage_image": "498b0271475211002ee987e8dee490ea.iix",
"sys_id": "109cddf8c6112276003b17991a09ad65",
"child_categories": [
{
"sys_id": "109f0438c6112276003ae8ac13e7009d",
"title": "Services",
"count": 16,
"parent": "109cddf8c6112276003b17991a09ad65",
"subcategories": [],
"isParentCategory": false
}
]
}
```
Example: Python request

# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_sc/servicecatalog/categories/2809952237b1300054b6a3549dbe5dd4'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/json'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "title": "Software",
        "description": "A range of software products available for installation on your corporate laptop or desktop computer.\n\n",
        "full_description": null,
        "icon": "",
        "header_icon": "",
        "homepage_image": "21f973e9471211002ee987e8dee49019.iix",
        "sys_id": "2809952237b1300054b6a3549dbe5dd4",
        "child_categories": []
    }
}
Service Catalog - GET /sn_sc/servicecatalog/items

Retrieves a list of catalog items based on the specified parameters.

**URL format**

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/items

Default URL: /api/sn_sc/servicecatalog/items

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_catalog</td>
<td>Catalog sys_id of the item. Use this parameter to locate items in a specific catalog. Data type: String</td>
</tr>
<tr>
<td>sysparm_category</td>
<td>Category sys_id of the item. Use this parameter to locate catalog items in a specific category. Data type: String</td>
</tr>
<tr>
<td>sysparm_limit</td>
<td>Maximum number of records to return. For requests that exceed this number of records, use the sysparm_offset parameter to paginate record retrieval. This limit is applied before ACL evaluation. If no records return, including records you have access to, rearrange the record order so records you have access to return first.</td>
</tr>
</tbody>
</table>

**Note:** Unusually large sysparm_limit values can impact system performance.
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_offset</td>
<td>Data type: Number, Default: 1000. Starting record index for which to begin retrieving records. Use this value to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, sysparm_offset is set to &quot;0&quot;. To simply page through all available records, use sysparm_offset=sysparm_offset + sysparm_limit, until you reach the end of all records. Do not pass a negative number in the sysparm_offset parameter. Data type: Number, Default: 0.</td>
</tr>
<tr>
<td>sysparm_text</td>
<td>Data type: String, Specific text for which to search for in the category items. For example, sysparm_text=&quot;iPhone&quot; or sysparm_text=&quot;Mobile devices&quot;.</td>
</tr>
<tr>
<td>sysparm_type</td>
<td>Data type: String, Type of item. Valid search values: Record Producer, Order Guide. If any other value is passed, the information is ignored.</td>
</tr>
<tr>
<td>sysparm_view</td>
<td>Data type: String, UI view for which to render the data. Determines the fields returned in the response. Valid values: desktop, mobile, both.</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If you also specify the <code>sysparm_fields</code> parameter, it takes precedence.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>. Default: <code>application/json</code></td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returns a list of catalogs and items based on the specified parameters. If a valid query results in no items, then the response body contains only an empty result array.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalogs</td>
<td>List of objects that describe the catalogs that contain this item. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;catalogs&quot;: [{ &quot;sys_id&quot;: &quot;String&quot;, &quot;title&quot;: &quot;String&quot; }]</td>
</tr>
<tr>
<td>catalogs.sys_id</td>
<td>Sys_id of the catalog. Data type: String</td>
</tr>
<tr>
<td>catalogs.title</td>
<td>Catalog title. Data type: String</td>
</tr>
<tr>
<td>category</td>
<td>Describes the category that contains this item. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;catalogs&quot;: { &quot;sys_id&quot;: &quot;String&quot;, &quot;title&quot;: &quot;String&quot; }</td>
</tr>
<tr>
<td>category.sys_id</td>
<td>Sys_id of the category.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>category.title</strong></td>
<td>Category title</td>
</tr>
<tr>
<td><strong>content_type</strong></td>
<td>Type of content.</td>
</tr>
<tr>
<td><strong>description</strong></td>
<td>Detailed description of the item.</td>
</tr>
<tr>
<td><strong>icon</strong></td>
<td>Source of the small icon displayed beside the item name.</td>
</tr>
<tr>
<td><strong>kb_article</strong></td>
<td>Knowledge based article associated with the item.</td>
</tr>
<tr>
<td><strong>local_currency</strong></td>
<td>Country currency code for the local currency.</td>
</tr>
<tr>
<td><strong>localized_price</strong></td>
<td>Price of the item expressed in the currency defined by <code>local_currency</code>.</td>
</tr>
<tr>
<td><strong>localized_recurring_price</strong></td>
<td>Price of the recurring fee expressed in the currency defined by <code>local_currency</code>.</td>
</tr>
<tr>
<td><strong>name</strong></td>
<td>Name of the item.</td>
</tr>
<tr>
<td><strong>order</strong></td>
<td>Pre-defined order to which the item belongs.</td>
</tr>
<tr>
<td><strong>picture</strong></td>
<td>Path to the image to display for the item.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>price</td>
<td>Price of item expressed in the currency defined by <code>price_currency</code>. Data type: String</td>
</tr>
<tr>
<td>price_currency</td>
<td>Country currency code for which the item price is expressed. Data type: String</td>
</tr>
<tr>
<td>recurring_frequency</td>
<td>Frequency at which the recurring fees are applied, such as daily, monthly, or yearly; null if there are no fees for the item. Data type: String</td>
</tr>
<tr>
<td>recurring_price</td>
<td>Amount of the recurring fee associated with the item. Data type: String</td>
</tr>
<tr>
<td>recurring_price_currency</td>
<td>Country currency code for which the recurring fees are expressed. Data type: String</td>
</tr>
<tr>
<td>short_description</td>
<td>Brief description of the item. Data type: String</td>
</tr>
<tr>
<td>show_price</td>
<td>Flag that indicates whether to show item pricing in the cart. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Name-value pairs are returned in the response parameters.</td>
</tr>
<tr>
<td></td>
<td>• false: Name-value pairs are not returned in the response parameters.</td>
</tr>
<tr>
<td>show_quantity</td>
<td>Flag that indicates whether to display the item quantity in the cart. Possible values:</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| • true: Display the item quantity in the cart.  
• false: Do not display the item quantity in the cart. |
| sys_class_name    | Table to which the record belongs, such as sc_cat_item_producer. Data type: String                                                        |
| sys_id            | Sys_id of the item. Data type: String                                                                                                     |
| type              | Type of item. Data type: String                                                                                                             |
| url               | URL of the item. Data type: String                                                                                                         |

**Example: cURL request**

```bash
curl  
"https://instance.servicenow.com/api/sn_sc/servicecatalog/items?sysparm_catalog=e0d08b13c3330100c8b837659bba8fb4&sysparm_limit=2" 
--request GET 
--header "Accept:application/json" \  
--user "username":"password"
```

```json
{
  "result": [
    {
      "short_description": null,
      "kb_article": ",
      "icon": "images/icons/catalog_item.gifx",
      "description": "<p class="p1"><font size="2"><span class="s1">Request an existing Standard Change Template is made unavailable when it is no longer required or no longer acceptable as a Standard Change. This will be confirmed by your Change Management team.</span></font></p>",
      "show_price": false,
      "type": "record_producer",
      "show_quantity": true,
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
A role delegator may delegate any role they have to any member of the specified group.

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**Example: Python request**

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_sc/servicecatalog/items?sysparm_text=iPhone&sysparm_limit=1'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Accept":"application/json"}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)
```

```json
{
    "result": [
      {
        "short_description": "Apple iPhone 5",
        "kb_article": "",
        "icon": "images/service_catalog/generic_small.gifx",
        "description": "\n            <font face="arial, helvetica, sans-serif" size="3">\n            iPhone © 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
        </font>"
      }
    ]
}
```
A slim and stylish design makes the Apple iPhone 5 smartphone lightweight and easy-to-carry around.

Key Features:

- Storage Capacity: 64 GB
- Color: Black and Slate
- Network Generation: 2G, 3G, 4G
- Style: Smartphone
- Camera: 8.0 MP
- Operating System: iOS

<table>
<thead>
<tr>
<th>Show_Price</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recurring_Price</td>
<td>&quot;$30.00&quot;</td>
</tr>
<tr>
<td>Type</td>
<td>catalog_item</td>
</tr>
<tr>
<td>Show_Quantity</td>
<td>true</td>
</tr>
<tr>
<td>Local_Currency</td>
<td>USD</td>
</tr>
<tr>
<td>Sys_Class_Name</td>
<td>pc_hardware_cat_item</td>
</tr>
<tr>
<td>Picture</td>
<td>7aefa450cb021200f2de77a4634c9c9a.iix</td>
</tr>
<tr>
<td>Url</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>Sys_Id</td>
<td>0d08837237153000158bbfc8bcb5e502</td>
</tr>
<tr>
<td>Recurring_Price_Currency</td>
<td>USD</td>
</tr>
<tr>
<td>Content_Type</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>Localized_Price</td>
<td>&quot;$599.99&quot;</td>
</tr>
<tr>
<td>Price</td>
<td>&quot;$599.99&quot;</td>
</tr>
<tr>
<td>Catalogs</td>
<td>[]</td>
</tr>
</tbody>
</table>

- Sys_Id | e0d08b13c3330100c8b837659bba8fb4 |
  - Title | Service Catalog |

- Sys_Id | d68eb4d637b1300054b6a3549dbe5db2 |
  - Title | Mobiles |

- Price_Currency | USD |
- Order | 10 |
Service Catalog - GET /sn_sc/servicecatalog/items/{item_sys_id}/delegation/{user_sys_id}

Verifies whether the specified delegated user has acquisition rights to the specified service catalog item.

**URL format**

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/items/{item_sys_id}/delegation/{user_sys_id}

Default URL: /api/sn_sc/servicecatalog/items/{item_sys_id}/delegation/{user_sys_id}

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>api_version</td>
</tr>
<tr>
<td>item_sys_id</td>
</tr>
<tr>
<td>user_sys_id</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Flag that indicates whether the specified user has acquisition rights to the specified service catalog item. Valid values: • true: User has acquisition rights. • false: User does not have acquisition rights.</td>
</tr>
</tbody>
</table>

Example: cURL request

```bash
curl "https://instance.servicenow.com/api/sn_sc/servicecatalog/items/5c7e9fd20f1010108af26b198767eb8/delegation/62826bf03710200044e0bfc8bcbe5df1" \
--request GET \
--header "Accept:application/json" \
--user "username":"password"
```

```
{
  "result": {
    "result": true
  }
}
```

Service Catalog - GET /sn_sc/servicecatalog/items/{sys_id}
Retrieves a specified catalog item.

URL format
Versioned URL: /api/sn_sc/{api_version}/servicecatalog/items/{sys_id}
Default URL: /api/sn_sc/servicecatalog/items/{sys_id}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>Sys_id of the catalog item to return. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_view</td>
<td>UI view for which to render the data. Determines the fields returned in the response. Valid values: desktop, mobile, both. If you also specify the sysparm_fields parameter, it takes precedent. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml.</td>
</tr>
</tbody>
</table>
Request headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Indicates that the request has completed successfully and returns the details of the catalog item.</td>
</tr>
</tbody>
</table>
| 400         | Indicates that request is invalid for one of the following reasons:  
  • User does not have access to the catalog item.  
  • Invalid sys_id. |
| 401         | Unauthorized. The user credentials are incorrect or have not been passed. |
| 500         | Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error. |

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| catalogs | List of objects that describe the catalogs that contain this item.  
Data type: Array |

```json
"catalogs": [
  "sys_id": "String",
]"
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalogs.sys_id</td>
<td>Sys_id of the catalog.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>catalogs.title</td>
<td>Catalog title.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>categories</td>
<td>List of objects that describe the categories that contains this item.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>categories.active</td>
<td>Flag that indicates whether the category is active.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Category is active.</td>
</tr>
<tr>
<td></td>
<td>• false: Category is inactive.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>categories.category</td>
<td>Describes the associated category.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>categories.category.active</td>
<td>Flag that indicates whether the category is active.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Category is active.</td>
</tr>
<tr>
<td></td>
<td>• false: Category is inactive.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>categories.category.sys_id</td>
<td>Sys_id of the category. Data type: String</td>
</tr>
<tr>
<td>categories.category.title</td>
<td>Category title. Data type: String</td>
</tr>
<tr>
<td>categories.sys_id</td>
<td>Parent category of the category specified in <code>categories.category</code>. Data type: String</td>
</tr>
<tr>
<td>category</td>
<td>Primary category to which the item belongs. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;category&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;title&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>category.sys_id</td>
<td>Sys_id of the primary category. Data type: String</td>
</tr>
<tr>
<td>category.title</td>
<td>Primary category's title. Data type: String</td>
</tr>
<tr>
<td>client_script</td>
<td>List of all the catalog client scripts defined on the catalog item. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;client_script&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;onChange&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;onLoad&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;onSubmit&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>client_script.onChange</td>
<td>List of onChange client scripts associated with the item. Data type: Array</td>
</tr>
<tr>
<td>client_script.onSubmit</td>
<td>List of onsubmit client scripts associated with the item. Data type: Array</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>client_script.onLoad</td>
<td>List of onLoad client scripts associated with the item. Data type: Array</td>
</tr>
<tr>
<td>content_type</td>
<td>Type of content. Data type: String</td>
</tr>
<tr>
<td>data_lookup</td>
<td>List of data lookups defined on the catalog item. Data type: Array</td>
</tr>
<tr>
<td>description</td>
<td>Detailed description of the item. Data type: String</td>
</tr>
<tr>
<td>icon</td>
<td>Source of the small icon displayed beside the item name. Data type: String</td>
</tr>
<tr>
<td>kb_article</td>
<td>Knowledge based article associated with the item. Data type: String</td>
</tr>
<tr>
<td>local_currency</td>
<td>Country currency code for the local currency. Data type: String</td>
</tr>
<tr>
<td>localized_price</td>
<td>Price of the item expressed in the currency defined by <code>local_currency</code>. Data type: String</td>
</tr>
<tr>
<td>localized_recurring_price</td>
<td>Price of the recurring fee expressed in the currency defined by <code>local_currency</code>. Data type: String</td>
</tr>
<tr>
<td>name</td>
<td>Name of the item. Data type: String</td>
</tr>
<tr>
<td>order</td>
<td>Number of orders for the item. Data type: Number</td>
</tr>
<tr>
<td>picture</td>
<td>File name of the image to display for the item. Data type: String</td>
</tr>
<tr>
<td>price</td>
<td>Price of item expressed in the currency defined by <code>price_currency</code>. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>price_currency</td>
<td>Country currency code for which the item price is expressed.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>recurring_frequency</td>
<td>Frequency at which the recurring fees are applied, such as daily, monthly, or yearly.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>recurring_price</td>
<td>Amount of the recurring fee associated with the item</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>recurring_price_currency</td>
<td>Country currency code for which the recurring fees are expressed.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>short_description</td>
<td>Brief description of the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>show_price</td>
<td>Flag that indicates whether to show item pricing in the cart.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Pricing name/value pairs are returned in the response parameters.</td>
</tr>
<tr>
<td></td>
<td>• false: Pricing name/value pairs are not returned in the response parameters.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>show_quantity</td>
<td>Flag that indicates whether to display the item quantity in the cart.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Display the item quantity in the cart.</td>
</tr>
<tr>
<td></td>
<td>• false: Do not display the item quantity in the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>sys_class_name</td>
<td>Name of the table to which this record belongs.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the item.</td>
</tr>
<tr>
<td>type</td>
<td>Type of item.</td>
</tr>
<tr>
<td>ui_policy</td>
<td>List of UI policies associated with the item.</td>
</tr>
<tr>
<td>url</td>
<td>URL of item image.</td>
</tr>
<tr>
<td>variables</td>
<td>List of available variables associated with the item.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl "https://instance.servicenow.com/api/sn_sc/servicecatalog/items/0cf7477237153000158bbfc8be5dec" \
    --request GET \
    --header "Accept:application/json" \
    --user "username":"password"
```

```json
{
    "result": {
        "short_description": "Apple iPad 3",
        "kb_article": "",
        "icon": "e6b3bc0647311200e0ef563dbb9a71fc.iix",
        "description": "iPad with Retina display Wi-Fi 16GB - Black
			<p>
				<font size="3">
					<span style="color: gray;">Retina display (2048x1536 resolution)</span>
				</font>
			</p>
			<ul>
			<li style="color: gray;">16GB storage</li>
			<li style="color: gray;">Wi-Fi enabled</li>
			</ul>
			Built-in speaker, microphone and
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_sc/servicecatalog/items/0cf7477237153000158bbfc8bce5dec'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8">
<response>
  <result>
    <short_description>Google Nexus 7</short_description>
    <icon>images/service_catalog/generic_small.gif</icon>
  </result>
</response>
```
&lt;h3&gt;Google Nexus 7 Black Wi-Fi 16GB Tablet, Model NEXUS7ASUS1B16&lt;/h3&gt;

**Key Features:**

- The world's 1st Android™ 4.1 Jelly Bean tablet
- World's first 7" quad-core tablet delivers an advanced multimedia experience with up to 9.5* hours of battery life
- World's best NFC experience on a tablet with a textured tactile design for enhanced comfort measuring just 10.45mm thin and 340g light
- ASUS TruVivid technology with Corning® Fit Glass for improved color clarity and scratch resistance
- 178 wide-viewing angle IPS display ensures unrivaled visual acuity

&lt;show_price /&gt;

&lt;recurring_price&gt;$10.00&lt;/recurring_price&gt;

&lt;type&gt;catalog_item&lt;/type&gt;

&lt;local_currency&gt;USD&lt;/local_currency&gt;

&lt;sys_id&gt;cf7477237153000158bbfc8bcbefdec&lt;/sys_id&gt;

&lt;recurring_price_currency&gt;USD&lt;/recurring_price_currency&gt;

&lt;localized_price&gt;$199.99&lt;/localized_price&gt;

&lt;client_script /&gt;

&lt;price&gt;$199.99&lt;/price&gt;

&lt;catalogs&gt;
  &lt;sys_id&gt;e0d08b13c3330100c8b837659bba8fb4&lt;/sys_id&gt;
    &lt;title&gt;Service Catalog&lt;/title&gt;
&lt;/catalogs&gt;

&lt;recurring_frequency&gt;Weekly&lt;/recurring_frequency&gt;

&lt;name&gt;Google Nexus 7&lt;/name&gt;

&lt;localized_recurring_price&gt;$10.00&lt;/localized_recurring_price&gt;

&lt;categories&gt;
  &lt;sys_id&gt;d258b953c611227a0146101fb1be7c31&lt;/sys_id&gt;
    &lt;active /&gt;
    &lt;category&gt;
      &lt;sys_id&gt;b06546f23731300054b6a3549dbe5dd8&lt;/sys_id&gt;
        &lt;active /&gt;
        &lt;title&gt;Tablets&lt;/title&gt;
    &lt;/category&gt;
    &lt;title&gt;Hardware&lt;/title&gt;
&lt;/categories&gt;

&lt;category&gt;
  &lt;sys_id&gt;b06546f23731300054b6a3549dbe5dd8&lt;/sys_id&gt;
    &lt;title&gt;Tablets&lt;/title&gt;
&lt;/category&gt;

&lt;price_currency&gt;USD&lt;/price_currency&gt;
Service Catalog - GET /sn_sc/servicecatalog/wishlist
Retrieves the list of items in the logged in user's wish list.

URL format
Versioned URL: /api/sn_sc/{api_version}/servicecatalog/wishlist
Default URL: /api/sn_sc/servicecatalog/wishlist

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cart_id</td>
<td>Sys_id of the wish list cart. Data type: String</td>
</tr>
<tr>
<td>items</td>
<td>List of objects that describe the items in the wish list cart. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"items": [{
"cart_item_id": "String",
}]
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;catalog_item_id&quot;: &quot;String&quot;,</td>
<td>&quot;created_by&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;created_on&quot;: &quot;String&quot;,</td>
<td>&quot;delivery_time&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;item_id&quot;: &quot;String&quot;,</td>
<td>&quot;item_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;localized_price&quot;: &quot;String&quot;,</td>
<td>&quot;localized_recurring_price&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;name&quot;: &quot;String&quot;,</td>
<td>&quot;order_guide&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;picture&quot;: &quot;String&quot;,</td>
<td>&quot;price&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;quantity&quot;: &quot;String&quot;,</td>
<td>&quot;recurring_frequency&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;recurring_price&quot;: &quot;String&quot;,</td>
<td>&quot;recurring_subtotal&quot;: Number,</td>
</tr>
<tr>
<td>&quot;recurring_subtotal_price&quot;: &quot;String&quot;,</td>
<td>&quot;short_descriptions&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;show_price&quot;: Boolean,</td>
<td>&quot;show_quantity&quot;: Boolean,</td>
</tr>
<tr>
<td>&quot;show_recurring_price&quot;: Boolean,</td>
<td>&quot;subtotal&quot;: Number,</td>
</tr>
<tr>
<td>&quot;subtotal_price&quot;: &quot;String&quot;,</td>
<td>&quot;sys_class_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
<td>&quot;updated_by&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td>&quot;updated_on&quot;: &quot;String&quot;,</td>
<td>&quot;variables&quot;: {Object}</td>
</tr>
<tr>
<td>items.cart_item_id</td>
<td>Sys_id of the item. Same as <code>items.sys_id</code>.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>items.catalog_item_id</td>
<td>Sys_id of the catalog identifier for this item in the wish list cart.</td>
</tr>
<tr>
<td>Data type: String</td>
<td>Same as <code>items.item_id</code>.</td>
</tr>
<tr>
<td>items.created_by</td>
<td>User that initially put the item in the wish list cart.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>items.created_on</td>
<td>Date and time that the item was initially placed in the wishlist cart.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>items.delivery_time</td>
<td>Amount of time that it takes to deliver the item.</td>
</tr>
<tr>
<td>items.item_id</td>
<td>Unique catalog identifier for this item in the wish list cart (sys_id).</td>
</tr>
<tr>
<td>items.item_name</td>
<td>Name of the item.</td>
</tr>
<tr>
<td>items.localized_price</td>
<td>Price of the item expressed in the local currency.</td>
</tr>
<tr>
<td>items.localized_recurring_price</td>
<td>Price of the recurring fee expressed in the local currency.</td>
</tr>
<tr>
<td>items.name</td>
<td>Name of the item.</td>
</tr>
<tr>
<td>items.order_guide</td>
<td>Order guide to which the item is associated.</td>
</tr>
<tr>
<td>items.picture</td>
<td>Path to the image to display for the item.</td>
</tr>
<tr>
<td>items.price</td>
<td>Price of item.</td>
</tr>
<tr>
<td>items.quantity</td>
<td>Number of this item in the wish list cart.</td>
</tr>
<tr>
<td>items.recurring_frequency</td>
<td>Frequency at which the recurring fees are applied, such as daily, monthly, or yearly. Null if no recurring fees for this item.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>items.recurring_price</td>
<td>Amount of the recurring fee associated with the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.recurring_subtotal</td>
<td>Subtotal for the recurring fees for this item (recurring_price x quantity).</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>items.recurring_subtotal_price</td>
<td>Subtotal for the recurring fees for this item (recurring_price x quantity).</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.short_description</td>
<td>Brief description of the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.show_price</td>
<td>Flag that indicates whether to show item pricing in the cart. If this parameter is set to &quot;false&quot;, pricing name/value pairs are not returned in the response parameters.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>items.show_quantity</td>
<td>Flag that indicates whether to display the item quantity in the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>items.show_recurring_price</td>
<td>Flag that indicates whether to display the recurring fees in the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>itemssubtotal</td>
<td>Subtotal for this item - not including cents (price x quantity).</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>itemssubtotal_price</td>
<td>Subtotal for this item - including cents (price x quantity).</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.sys_class_name</td>
<td>Child class to which the associated item record belongs.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><strong>Name</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>items.sys_id</td>
<td>Unique identifier of the item (sys_id). Same as items.cart_item_id. Data type: String</td>
</tr>
<tr>
<td>items.updated_by</td>
<td>User that updated this item in the wish list cart. Data type: String</td>
</tr>
<tr>
<td>items.updated_on</td>
<td>Date and time that the item was put on the wishlist. Data type: String</td>
</tr>
<tr>
<td>items.variables</td>
<td>Name-value pairs of variables associated with the item. Data type: Object</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl "http://instance.servicenow.com/api/sn_sc/servicecatalog/wishlist" \
--request GET \
--header "Accept:application/json" \
--user 'username':'password'
```

```json
{
"result": {
"cart_id": "3a68c46013f12300027879d879d96144b0ce",
"items": [
{
"updated_on": "2018-11-16 05:54:41",
"catalog_item_id": "774906834fbb4200086eed81810c737",
"short_description": "Macbook Pro",
"recurring_subtotal_price": "$100.00",
"delivery_time": "5 Days 0 seconds",
"sys_class_name": "sc_cat_item",
"sys_id": "3b8d887113b5a300027879d96144b0c2",
"price": "$1,499.00",
"recurring_frequency": "Annually",
"subtotal_price": "$1,499.00",
"variables": {
"Adobe Photoshop": "false",
"Adobe Acrobat": "false",
"Optional Software": "true",
"Additional software requirements": "",
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'http://instance.servicenow.com/api/sn_sc/servicecatalog/wishlist'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
```
```python
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
    response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
    <result>
        <cart_id>02a559a7c3b02200d68d3b0ac3d3ae5d</cart_id>
        <items>
            <catalog_item_id>04b7e94b4f7b4200086eeed18110c7fd</catalog_item_id>
            <variables>
                <Adobe Photoshop></Adobe Photoshop>
                <Adobe Acrobat></Adobe Acrobat>
                <Optional Software></Optional Software>
                <Additional software requirements></Additional software requirements>
            </variables>
            <quantity>1</quantity>
            <localized_price>$1,100.00</localized_price>
            <price>$1,100.00</price>
            <recurring_frequency>Annually</recurring_frequency>
            <localized_recurring_price>$100.00</localized_recurring_price>
            <recurring_price>$100.00</recurring_price>
            <item_name>Standard Laptop &</item_name>
            <cart_item_id>d31be364c3012200d68d3b0ac3d3aeecf</cart_item_id>
            <delivery_time>5 Days</delivery_time>
        </items>
    </result>
</response>
```

**Service Catalog - GET /sn_sc/servicecatalog/wishlist/{cart_item_id}**

Retrieves the details of the specified item stored in the wish list cart.

**URL format**

Versioned URL: /api/sn_sc/v1/servicecatalog/wishlist/{cart_item_id}

Default URL: /api/sn_sc/servicecatalog/wishlist/{cart_item_id}
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cart_item_id</td>
<td>Sys_id of the wish list cart item to retrieve.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Indicates a failed request. Incorrect cart item id or the item does not exist in the wish list cart.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cart_item_id</td>
<td>Sys_id of the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>catalog_item_id</td>
<td>Sys_id of the catalog identifier for this item in the wish list cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>delivery_time</td>
<td>Amount of time that it takes to deliver the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>item_name</td>
<td>Name of the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>localized_price</td>
<td>Price of the item expressed in the local currency.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>localized_recurring_price</td>
<td>Price of the recurring fee expressed in the local currency.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>price</td>
<td>Price of item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>quantity</td>
<td>Number of this item in the wish list cart.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>recurring_frequency</td>
<td>Frequency at which the recurring fees are applied, such as daily, monthly, or yearly. Null if no recurring fees for this item.</td>
</tr>
<tr>
<td>recurring_price</td>
<td>Amount of the recurring fee associated with the item.</td>
</tr>
<tr>
<td>variables</td>
<td>Name-value pairs of the variables associated with the item.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl
   "http://instance.servicenow.com/api/sn_sc/servicecatalog/wishlist/774906834fbb4200086eeed18110c737" \
   --request GET \
   --header "Accept:application/json" \
   --user "username":"password"
{
   "result": {
      "catalog_item_id": "04b7e94b4f7b4200086eeed18110c7fd",
      "variables": {
         "Adobe Photoshop": ",",
         "Adobe Acrobat": ",",
         "Optional Software": ",",
         "Additional software requirements": ","
      },
      "quantity": "1",
      "localized_price": "$1,100.00",
      "price": "$1,100.00",
      "recurring_frequency": "Annually",
      "localized_recurring_price": "$100.00",
      "recurring_price": "$100.00",
      "item_name": "Standard Laptop ",
      "cart_item_id": "d31be364c3012200d68d3b0ac3d3aecc",
      "delivery_time": "5 Days"
   }
}
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url =
'http://instance.servicenow.com/api/sn_sc/servicecatalog/wishlist/d31be364c3012200d68d3b0ac3d3aece'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/xml'}

# Make the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
    <result>
        <catalog_item_id>04b7e94b4f7b4200086eeed18110c7fd</catalog_item_id>
        <quantity>1</quantity>
        <variables>
            <Adobe Photoshop/></Adobe Photoshop>
            <Adobe Acrobat/></Adobe Acrobat>
            <Optional Software/></Optional Software>
            <Additional software requirements/></Additional software requirements>
        </variables>
        <localized_price>$1,100.00</localized_price>
    </result>
</response>
```
Service Catalog - POST /sn_sc/servicecatalog/cart/checkout

Retrieves and processes the checkout for the current cart based on whether the two-step checkout process is enabled.

If one-step checkout, the method checks out (saves) the cart and returns the request number and the request order ID. If two-step checkout, the method returns the cart order status and all the information required for two-step checkout.

URL format

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/cart/checkout
Default URL: /api/sn_sc/servicecatalog/cart/checkout

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>api_version</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

#### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Indicates that the request completed successfully. Returns the total and subtotal information and information on the items in the cart.</td>
</tr>
<tr>
<td>400</td>
<td>Indicates that the cart is empty and cannot check out.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cart_id</td>
<td>Sys_id of the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>delivery_address</td>
<td>Address to which to deliver the cart items.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;</code></td>
<td>List of items in the cart having a specific recurring frequency. Possible values include:</td>
</tr>
<tr>
<td></td>
<td>• daily</td>
</tr>
<tr>
<td></td>
<td>• monthly</td>
</tr>
<tr>
<td></td>
<td>• yearly</td>
</tr>
<tr>
<td></td>
<td>• none (no recurring fees)</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;&lt;recurring_frequency&gt;&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;frequency_label&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;items&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;show_subtotal_price&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;subtotal_price&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;subtotal_recurring_frequency&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;subtotal_recurring_price&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;subtotal_title&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;total_title&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.frequency_label</code></td>
<td>Frequency of the associated recurring fee. If there are no recurring fees for the item, this parameter contains &quot;null&quot;.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items</code></td>
<td>Array of objects that contains details of each item associated with the specified recurring frequency category.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| <recurring_frequency>.items.cart_item_id | Sys_id of the item in this cart.  
Data type: String                                   |
| <recurring_frequency>.items.catalog_item_id | Catalog sys_id of the item.  
Data type: String                                   |
| <recurring_frequency>.items.created.by | User that initially placed the item in the cart.  
Data type: String                                   |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.created_on</code></td>
<td>Date on which the item was initially placed in the cart. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.delivery_time</code></td>
<td>Time that the specified item takes to be delivered. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.item_id</code></td>
<td>Sys_id of the item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.item_name</code></td>
<td>Name of the specified item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.localized_price</code></td>
<td>Price of the item in local currency. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.localized_recurring_price</code></td>
<td>Recurring fee for the specified item in local currency. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.name</code></td>
<td>Item name. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.order_guide</code></td>
<td>Name of the order guide to which this item is associated. If this item is not associated with an order guide this parameter contains an empty string. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.picture</code></td>
<td>File name of the picture of the item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.price</code></td>
<td>Price of the item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.quantity</code></td>
<td>Number of the specified item in the cart. Data type: String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.recurring_frequency</code></td>
<td>How often the recurring fee is charged for the specified item. If there are no recurring fees for the item, this parameter contains &quot;null&quot;. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.recurring_price</code></td>
<td>Recurring fee for the specified item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.recurring_subtotal</code></td>
<td>Subtotal of the recurring fees for the item (recurring_price * quantity). Data type: Number</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.recurring_subtotal_price</code></td>
<td>Subtotal price of the recurring fees for the item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.short_description</code></td>
<td>Short description of the item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.show_price</code></td>
<td>Flag that indicates whether the price of the item is shown in the cart. Possible values: • true: Show the price of the item in the cart. • false: Do not show the price of the item in the cart. Data type: Boolean</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.show_quantity</code></td>
<td>Flag that indicates whether to show the item quantity in the cart. Possible values: • true: Show the item quantity in the cart. • false: Do not show the item quantity in the cart. Data type: Boolean</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| `<recurring_frequency>.items.show_recurring_price` | Flag that indicates whether the recurring fees should appear in the cart. Possible values:  
  - true: Show the recurring fees in the cart.  
  - false: Do not show the recurring fees in the cart.  
  Data type: Boolean |
| `<recurring_frequency>.items.subtotal` | Subtotal number of items.  
  Data type: Number |
| `<recurring_frequency>.items.subtotal_price` | Subtotal for the item \(( \text{price} \times \text{quantity} )\).  
  Data type: String |
| `<recurring_frequency>.items.sys_class_name` | System classification name of the item.  
  Data type: String |
| `<recurring_frequency>.items.sys_id` | Sys_id of the item record.  
  Data type: String |
| `<recurring_frequency>.items.updated_by` | User that updated the item after it was initially placed in the cart.  
  Data type: String |
| `<recurring_frequency>.items.updated_on` | Last date/time on which the item was either initially placed in the cart or updated.  
  Data type: String |
| `<recurring_frequency>.items.variables` | List of name-value pairs of the item variables.  
  Data type: Object |
<p>| <code>&lt;recurring_frequency&gt;.show_subtotal_price</code> | Flag that indicates whether to show a subtotal for the associated recurring fee category. Possible values: |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>show_subtotal_price</td>
<td>Flag that indicates whether to show the subtotal price in the cart. Possible values:</td>
</tr>
<tr>
<td>subtotal_title</td>
<td>If the show_subtotal_price is set to true, the text to display in the cart for the associated subtitle. Data type: String</td>
</tr>
<tr>
<td>total_title</td>
<td>Text to display for the total title for the associated recurring frequency category. Data type: String</td>
</tr>
<tr>
<td>requested_for</td>
<td>Sys_id of the user for whom the item was ordered. Data type: String</td>
</tr>
<tr>
<td>requested_for_user</td>
<td>User that requested the item on behalf of someone else. Data type: String</td>
</tr>
<tr>
<td>requested_for_User</td>
<td>User that requested the item on behalf of someone else. Data type: String</td>
</tr>
<tr>
<td>recurring_frequency</td>
<td>Recurring frequency category, such as daily, monthly, etc. Same as subtotal_recurring_frequency. Data type: String</td>
</tr>
<tr>
<td>subtotal_recurring_price</td>
<td>Subtotal of the recurring fees of the items within the associated recurring frequency category. Data type: String</td>
</tr>
<tr>
<td>subtotal_price</td>
<td>Subtotal of the cost of all items for the associated recurring frequency category. Data type: String</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td>• true: Show subtotals in the cart. • false: Do not show subtotals in the cart.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td>• true: Show subtotals in the cart. • false: Do not show subtotals in the cart.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td>• true: Show subtotals in the cart. • false: Do not show subtotals in the cart.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td>• true: Show subtotals in the cart. • false: Do not show subtotals in the cart.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>true: false.</td>
<td>true: Show subtotals in the cart.</td>
</tr>
<tr>
<td>false: Do not show</td>
<td>false: Do not show subtotals in the cart.</td>
</tr>
<tr>
<td>subtotals in the</td>
<td></td>
</tr>
<tr>
<td>cart.</td>
<td></td>
</tr>
<tr>
<td>special_instructions</td>
<td>Any instructions given by the customer on the item.</td>
</tr>
<tr>
<td>subtotal_price</td>
<td>Subtotal of all items in the cart in whole dollars.</td>
</tr>
<tr>
<td>subtotal_recurring_f</td>
<td>Type of subtotal recurring frequency, such as daily, monthly, etc. Same as &lt;recurring_frequency&gt;.subtotal_recurring_frequency.</td>
</tr>
<tr>
<td>subtotal_recurring_p</td>
<td>Subtotal of the recurring frequency fees for all items in the cart.</td>
</tr>
<tr>
<td>subtotal_title</td>
<td>Text to display in the cart for the subtotal field title.</td>
</tr>
<tr>
<td>total_title</td>
<td>Text to display in the cart for the total field title.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl "https://instance.servicenow.com/api/sn_sc/servicecatalog/cart/checkout" \
    --request POST \
    --header "Accept:application/json" \
    --user "username":"password"
```

```
{
  "result": {
    "cart_id": "0830db8413a56300397533e2e144b0ba",
    "subtotal_price": "$1,598.00",
```
"subtotal_recurring_frequency": ",
"delivery_address": ",
"special_instructions": ",
"subtotal_recurring_price": ",
"total_title": "Total",
"requested_for_user": "System Administrator",
"none": {
"subtotal_price": "$1,598.00",
"subtotal_recurring_frequency": ",
"subtotal_recurring_price": "$0.00",
"total_title": "Total",
"frequency_label": null,
"items": [

"updated_on": "2018-11-06 17:01:21",
"catalog_item_id": "07f1666b0f00300eba42da0d5673ab0",
"short_description": "Apple iPhone 7 ",
"recurring_subtotal_price": "$0.00",
"delivery_time": "2 Days",
"sys_class_name": "sc_cat_item",
"sys_id": "a846872a1369e300027879d96144b07e",
"price": "$799.00",
"recurring_frequency": null,
"subtotal_price": "$799.00",
"variables": {},
"quantity": "1",
"item_id": "07f1666b0f00300eba42da0d5673ab0",
"recurring_price": "$0.00",
"show_price": true,
"item_name": "Apple iPhone 7",
"cart_item_id": "a846872a1369e300027879d96144b07e",
"recurring_subtotal": 0,
"show_quantity": true,
"created_by": "admin",
"picture": "e823ee6b0bf00300eba42da0d5673a95.bxk",
"created_on": "2018-11-06 17:01:21",
"localized_price": "$799.00",
"subtotal": 799,
"show_recurring_price": false,
"name": "Apple iPhone 7",
"updated_by": "admin",
"localized_recurring_price": "$0.00",
"order_guide": ""
},

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_sc/servicecatalog/cart/checkout'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Accept":"application/xml"}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

<!-- This example response is returned when two-step checkout is enabled -->

```xml
<root>
    <result>
        <cart_id>616ebdb8db71120076861150f0b8f552</cart_id>
        <subtotal_price>$9,599.88</subtotal_price>
        <subtotal_recurring_frequency>Monthly</subtotal_recurring_frequency>
        <subtotal_recurring_price>$348.00</subtotal_recurring_price>
        <total_title>Total</total_title>
        <delivery_address>Brasilia, Brasil</delivery_address>
        <special_instructions></special_instructions>
        <requested_for_user>System Administrator</requested_for_user>
        <monthly>
            <subtotal_price>$9,599.88</subtotal_price>
            <subtotal_recurring_frequency>Monthly</subtotal_recurring_frequency>
            <subtotal_recurring_price>$348.00</subtotal_recurring_price>
            <total_title>Total</total_title>
        </monthly>
    </result>
</root>
```
Service Catalog - POST /sn_sc/servicecatalog/cart/submit_order

Checks out the user cart, based on the current check-out type (one-step or two-step).

URL format

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/cart/submit_order

Default URL: /api/sn_sc/servicecatalog/cart/submit_order
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Indicates that the request completed successfully and the items in the cart are successfully checked out.</td>
</tr>
<tr>
<td>400</td>
<td>Indicates that the user cart is empty.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent_id</td>
<td>If available, sys_id of the parent record from which the request is created. For more information, see Configure a parent mapping for a request.</td>
</tr>
<tr>
<td>parent_table</td>
<td>Name of the parent table.</td>
</tr>
<tr>
<td>request_id</td>
<td>Sys_id of the order request generated.</td>
</tr>
<tr>
<td>request_number</td>
<td>Number of the request generated.</td>
</tr>
</tbody>
</table>

Example: cURL request

```bash
curl "https://instance.servicenow.com/api/sn_sc/v1/servicecatalog/cart/submit_order " \
--request POST \
--header "Accept:application/json" \
--user "username":"password"
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_sc/v1/servicecatalog/cart/submit_order '

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Accept': 'application/json'}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)
```

Service Catalog - POST /sn_sc/servicecatalog/items/{sys_id}/add_to_cart

Adds the specified item to the cart of the current user.
**URL format**

Versioned URL: `/api/sn_sc/{api_version}/servicecatalog/items/{sys_id}/add_to_cart`

Default URL: `/api/sn_sc/servicecatalog/items/{sys_id}/add_to_cart`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, <code>v1</code> or <code>v2</code>. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the item to add to the current cart. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_also_request_for</td>
<td>Comma-separated string of user sys_ids of other users for which to order the specified item. User sys_ids are located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>• You cannot specify more than 50 additional users to receive the item.</td>
</tr>
<tr>
<td></td>
<td>• If any person in the list doesn’t qualify for the item, the entire request is rejected.</td>
</tr>
<tr>
<td></td>
<td>• If the associated item does not have the <code>requested_for</code> variable set, the request is rejected.</td>
</tr>
<tr>
<td></td>
<td>• If the associated item has the <code>requested_for</code> variable set, but the <code>Enable also Request for</code></td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>variable is not configured, or the calling user doesn’t have the proper roles, the request is rejected.</td>
</tr>
<tr>
<td>sysparm_quantity</td>
<td>Required. Quantity of the item. Cannot be a negative number.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>variables</td>
<td>Name-value pairs of all mandatory cart item variables. Mandatory variables are defined on the associated form.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>The following errors can occur:</td>
</tr>
<tr>
<td></td>
<td>• <em>Invalid quantity value</em>: Indicates that either the <code>sysparm_quantity</code> parameter is not provided or contains an invalid value.</td>
</tr>
<tr>
<td></td>
<td>• <em>Security constraints prevent ordering of Item</em>: Indicates that either the cart item <code>sys_id</code> specified in the path parameters is invalid or the user does not have access to the item.</td>
</tr>
<tr>
<td></td>
<td>• <em>Mandatory Variables are required</em>: Indicates that one or more of the mandatory variables is not provided in the request.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cart_id</td>
<td>Cart sys_id. Cart sys_id. Data type: String</td>
</tr>
<tr>
<td>items</td>
<td>Details of all of the items in the cart. Data type: Array</td>
</tr>
</tbody>
</table>

```json
"items": [
  {
    "cart_item_id": "String",
    "catalog_item_id": "String",
    "item_name": "String",
    "localized_recurring_price": "String",
    "localized_price": "String",
    "price": "String",
    "quantity": "String",
    "recurring_frequency": "String",
    "recurring_price": "String"
  }
]```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.cart_item_id</td>
<td>Sys_id of the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.catalog_item_id</td>
<td>Catalog sys_id of the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.item_name</td>
<td>Name of the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.localized_price</td>
<td>Price of the item, in the user's local currency.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.localized_recurring_price</td>
<td>Recurring price of the item, in the user's local currency.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.price</td>
<td>Base price.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.quantity</td>
<td>Quantity of the item in the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.recurring_frequency</td>
<td>Frequency at which the recurring price is applied.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.recurring_price</td>
<td>Recurring price.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>subtotal</td>
<td>Subtotal of the items in the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request and response**

```
curl
  "https://instance.servicenow.com/api/sn_sc/servicecatalog/items/0d08837237153000158bbfc8be5d02/add_to_cart" \
  --request POST \
  --header "Accept:application/json" \
  --data "{ 'sysparm_quantity' : '1',
```

Example: Sample Python request and response

```
# Install requests package for python
import requests

# Set the request parameters
url =
    'https://instance.servicenow.com/api/sn_sc/servicecatalog/items/0d08837237153000158bbfc8bcbe5d02/add_to_cart'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Content-Type': 'application/json', 'Accept': 'application/json'}

# Make the HTTP request
```
response = requests.post(url, auth=(user, pwd), headers=headers,
data="{"sysparm_quantity": ":2\}"*

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "cart_id": "0830db8413a56300397533e2e144b0ba",
        "subtotal": "$799.00",
        "items": [
            {
                "catalog_item_id": "07f1666b0bf000300eb42da0d5673ab0",
                "quantity": "2",
                "localized_price": "$799.00",
                "price": "$799.00",
                "recurring_frequency": null,
                "localized_recurring_price": "$0.00",
                "recurring_price": "$0.00",
                "item_name": "Apple iPhone7",
                "cart_item_id": "7d138fa21329e300027879d96144b065"
            }
        ]
    }
}

Service Catalog - POST /sn_sc/servicecatalog/items/{sys_id}/add_to_wishlist
Adds the specified item to the wish list cart.

URL format
Versioned URL: /api/sn_sc/{api_version}/servicecatalog/items/{sys_id}/
add_to_wishlist
Default URL: /api/sn_sc/servicecatalog/items/{sys_id}/add_to_wishlist
**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the item to add to the wish list. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Indicates a failed request. Invalid request data or the user cannot add item to the wish list.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Example: cURL request

```
curl
  "http://instance.servicenow.com/api/sn_sc/servicecatalog/items/04b7e94b4f7b4200086eeed18110c7fd/add_to_wishlist" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --data "{"sysparm_quantity":1"}"
```

```json
  "result": {
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'http://instance.service-now.com/api/sn_sc/servicecatalog/items/04b7e94b4f7b4200086eed18110c7fd/add_to_wishlist'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Content-Type":"application/json","Accept":"application/xml"}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, data="{"sysparm_quantity":"1"})
```
```python
# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

<?xml version="1.0" encoding="UTF-8"?>
<response>
    <result>
        <cart_id>02a559a7c3b02200d68d3b0ac3d3ae5d</cart_id>
        <items>
            <variables>
                <Adobe Photoshop></Adobe Photoshop>
                <Adobe Acrobat></Adobe Acrobat>
                <Optional Software></Optional Software>
                <Additional software requirements></Additional software requirements>
            </variables>
            <catalog_item_id>04b7e94b4f7b4200086eeed18110c7fd</catalog_item_id>
            <quantity>1</quantity>
            <localized_price>$1,100.00</localized_price>
            <price>$1,100.00</price>
            <recurring_frequency>Annually</recurring_frequency>
            <localized_recurring_price>$100.00</localized_recurring_price>
            <recurring_price>$100.00</recurring_price>
            <item_name>Standard Laptop</item_name>
            <cart_item_id>d31be364c3012200d68d3b0ac3d3aecf</cart_item_id>
            <delivery_time>5 Days</delivery_time>
        </items>
    </result>
</response>
```

Service Catalog - POST /sn_sc/servicecatalog/items/{sys_id}/checkout_guide
Retrieves an array of contents requested for checkout.

**URL format**

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/items/{sys_id}/checkout_guide

Default URL: /api/sn_sc/servicecatalog/items/{sys_id}/checkout_guide
# Supported request parameters

## Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the checkout guide for which to return the information.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

## Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

## Request body parameters (XML or JSON)

```
(items: [{
  "sys_id": "String",
  "sysparm_quantity": "String",
  "variables": {Object}
}])
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>items.sys_id</td>
<td>Required. Sys_id of the catalog item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items.sysparm_quantity</td>
<td>Number of items.</td>
</tr>
<tr>
<td></td>
<td>Data type: String, Default: 1</td>
</tr>
<tr>
<td>items.variables</td>
<td>Name-value pairs of variables associated with the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Default: No variables</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Indicates that the request has completed successfully.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><em>Is not two-step:</em> Items are added to the cart and the cart contents are returned.</td>
</tr>
<tr>
<td></td>
<td><em>Is two-step:</em> Returns the request number and request sys_id after checkout.</td>
</tr>
<tr>
<td>400</td>
<td>Indicates that the request is invalid. Could be due to one of the following reasons:</td>
</tr>
<tr>
<td></td>
<td>• One or more items sent do not exist.</td>
</tr>
<tr>
<td></td>
<td>• User does not have access to one or more of the items.</td>
</tr>
<tr>
<td></td>
<td>• Mandatory variables of one or more items have not been answered. (Only variables defined as mandatory are honored.)</td>
</tr>
<tr>
<td></td>
<td>• Quantity value is sent and is not a positive integer.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

#### Elements returned in the response body when two-step checkout is false

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cart_id</td>
<td>Sys_id of the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>items</td>
<td>List of objects that contain details of all the items in the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>
|        | `{
|        |   "cart_item_id": "String",
|        |   "catalog_item_id": "String",
|        |   "delivery_time": "String",
|        |   "item_name": "String",
|        |   "localized_price": "String",
|        | }` |
Elements returned in the response body when two-step checkout is false (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;localized_recurring_price&quot;</td>
<td>&quot;localized_recurring_price&quot;: &quot;String&quot;, &quot;price&quot;: &quot;String&quot;, &quot;quantity&quot;: &quot;String&quot;, &quot;recurring_frequency&quot;: &quot;String&quot;, &quot;recurring_price&quot;: &quot;String&quot;, &quot;variables&quot;: (Object)</td>
</tr>
<tr>
<td>items.cart_item_id</td>
<td>Sys_id of the item in the cart. Data type: String</td>
</tr>
<tr>
<td>items.catalog_item_id</td>
<td>Sys_id of the catalog item in the cart. Data type: String</td>
</tr>
<tr>
<td>items.delivery_time</td>
<td>Amount of time it takes to deliver the item. Data type: String</td>
</tr>
<tr>
<td>items.item_name</td>
<td>Name of the item in the cart. Data type: String</td>
</tr>
<tr>
<td>items.localized_price</td>
<td>Price of the item in local currency. Data type: String</td>
</tr>
<tr>
<td>items.localized_recurring_price</td>
<td>Recurring price of the item in local currency. Data type: String</td>
</tr>
<tr>
<td>items.price</td>
<td>Price of the item. Data type: String</td>
</tr>
<tr>
<td>items.quantity</td>
<td>Number of the item in the cart. Data type: String</td>
</tr>
<tr>
<td>items.recurring_frequency</td>
<td>Recurring frequency of the item. Data type: String</td>
</tr>
<tr>
<td>items.recurring_price</td>
<td>Recurring price of the item. Data type: String</td>
</tr>
<tr>
<td>items.variables</td>
<td>Name-value pairs associated with the item in the cart.</td>
</tr>
</tbody>
</table>
Elements returned in the response body when two-step checkout is false (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent_id</td>
<td>If available, the sys_id of the parent record from which the request is created. For more information, see <a href="#">Configure a parent mapping for a request</a>.</td>
</tr>
<tr>
<td>parent_table</td>
<td>Name of the parent table (case sensitive).</td>
</tr>
<tr>
<td>request_number</td>
<td>Request record number.</td>
</tr>
<tr>
<td>request_id</td>
<td>Sys_id of the request.</td>
</tr>
<tr>
<td>subtotal_price</td>
<td>Cart subtotal.</td>
</tr>
<tr>
<td>subtotal_recurring_frequency</td>
<td>Subtotal of the current frequency block.</td>
</tr>
<tr>
<td>subtotal_recurring_frequency</td>
<td>Recurring frequency subtotal of the cart.</td>
</tr>
<tr>
<td>subtotal_recurring_price</td>
<td>Recurring frequency subtotal of the current frequency block.</td>
</tr>
<tr>
<td>subtotal_recurring_price</td>
<td>Recurring price subtotal of the current frequency block.</td>
</tr>
<tr>
<td>total_title</td>
<td>Title for total field on page.</td>
</tr>
</tbody>
</table>
Elements returned in the response body when two-step checkout is false (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>total_title</td>
<td>Title for total field on the current frequency block.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

```bash
curl "https://instance.servicenow.com/api/sn_sc/v1/servicecatalog/items/6690750f4f7b4200086eeed18110c761/checkout_guide" \
--request POST \
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--data "{"items":[{
  \"sys_id\":\"04b7e94b4f7b4200086eeed18110c7fd\",\n  \"variables\":{
    \"acrobat\":\"true\",\n    \"Additional_software_requirements\":\"MS Office 2007\"
  },\n  \"sysparm_quantity\":\"3\"
},
{
  \"sys_id\":\"e1be6dcb4f7b4200086eeed18110c74c\"
},
{
  \"sys_id\":\"186d917a6fab7980575967ddbb3ee4f2\",\n  \"variables\":{
    \"new_email\":\"abel.tuter@example.com\"
  }
},
{
  \"sys_id\":\"8b3ae7f0d1e004ece5c08239e522b\"
},
{
  \"sys_id\":\"962967674ff38200086eeed18110c7e7\",\n  \"variables\":{
    \"ergonomic_office\":\"Office chair should have lumbar support\"
  }
}]]" \
--user "username":"password"
```
<!--Case-1: if two-step is false:-->  

{  
  "result": {  
    "cart_id": "0d8c7a32c3211200d68d3b0ac3d3ae4f",  
    "subtotal_price": "$3,350.00",  
    "subtotal_recurring_frequency": ",",  
    "subtotal_recurring_price": "$300.00",  
    "total_title": "Total",  
    "monthly": {  
      "subtotal_price": "$50.00",  
      "subtotal_recurring_frequency": "Monthly",  
      "subtotal_recurring_price": "$0.00",  
      "total_title": "Total",  
      "items": [  
        {  
          "catalog_item_id": "90af095bcd38798071a208d1b64f",  
          "variables": {},  
          "quantity": "1",  
          "localized_price": "$50.00",  
          "price": "$50.00",  
          "recurring_frequency": "Monthly",  
          "localized_recurring_price": "$0.00",  
          "recurring_price": "$0.00",  
          "item_name": "Belkin iPad Mini Case",  
          "cart_item_id": "cb6e485f3211200d68d3b0ac3d3ae4f",  
          "delivery_time": "2 Days"  
        }  
      ],  
      "show_subtotal_price": "true",  
      "subtotal_title": "Subtotal"  
    },  
    "none": {  
      "subtotal_price": "-\n",  
      "subtotal_recurring_frequency": ",",  
      "subtotal_recurring_price": "$0.00",  
      "total_title": "Total",  
      "items": [  
        {  
          "catalog_item_id": "186d917a6fab7980575967d4bb33e4f",  
          "variables": {  
            "Preferred Email address": "abel.tuter@example.com"  
          },  
          "quantity": "1",  
        }  
      ]  
    }  
  }  
}
"localized_price": "$0.00",
"price": "$0.00",
"recurring_frequency": "",
"localized_recurring_price": "$0.00",
"recurring_price": "$0.00",
"item_name": "New Email Account",
"cart_item_id": "17c5241fc3211200d68d3b0ac3d3ae7f",
"delivery_time": "global.Workflow",
},
{
"catalog_item_id": "8b3ae7fedc1be1004ece5c08239e522b",
"variables": {},
"quantity": "1",
"localized_price": "$0.00",
"price": "$0.00",
"recurring_frequency": "",
"localized_recurring_price": "$0.00",
"recurring_price": "$0.00",
"item_name": "Corp VPN",
"cart_item_id": "97c5241fc3211200d68d3b0ac3d3ae7f",
"delivery_time": "2 Days",
},
{
"catalog_item_id": "e1be6dcb4f7b4200086eeed18110c74c",
"variables": {},
"quantity": "1",
"localized_price": "$0.00",
"price": "$0.00",
"recurring_frequency": "",
"localized_recurring_price": "$0.00",
"recurring_price": "$0.00",
"item_name": "External Monitor",
"cart_item_id": "d3c5241fc3211200d68d3b0ac3d3ae7f",
"delivery_time": "2 Days",
},
{
"catalog_item_id": "962967674ff38200086eeed18110c7e7",
"variables": {
"Please describe any ergonomic requirements\n\t\t": "Office chair should have lumbar support"
},
"quantity": "1",
"localized_price": "$0.00",
"price": "$0.00",

"recurring_frequency": 
"localized_recurring_price": "$0.00",
"recurring_price": "$0.00",
"item_name": "Desk Set Up",
"cart_item_id": "d7c5241fc3211200d68d3b0ac3d3ae7f",
"delivery_time": "5 Days"
]
],
"show_subtotal_price": "false",
"subtotal_title": "Subtotal"
},
"yearly": {
"subtotal_price": "$3,300.00",
"subtotal_recurring_frequency": "Annually",
"subtotal_recurring_price": "$300.00",
"total_title": "Total",
"items": [
{
"catalog_item_id": "04b7e94b4f7b4200086eeed18110c7fd",
"variables": {
"Adobe Photoshop": "",
"Adobe Acrobat": "true",
"Optional Software": "",
"Additional software requirements": "MS Office 2007"
},
"quantity": "3",
"localized_price": "$1,100.00",
"price": "$1,100.00",
"recurring_frequency": "Annually",
"localized_recurring_price": "$100.00",
"recurring_price": "$100.00",
"item_name": "Standard Laptop",
"cart_item_id": "9fc5241fc3211200d68d3b0ac3d3ae7e",
"delivery_time": "5 Days"
}
],
"show_subtotal_price": "true",
"subtotal_title": "Subtotal"
},
"show_subtotal_price": "true",
"subtotal_title": "Subtotal"
}
Case-2: if two-step is true:
{
  "result": {
    "request_number": "REQ0010001",
    "request_id": "82a7e89fc3211200d68d3b0ac3d3ae0a"
  }
}

Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url =
    'https://instance.servicenow.com/api/sn_sc/v1/servicecatalog/items/6690750f4f7b4200086eed18110c761/checkout_guide'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Content-Type':"application/json","Accept":"application/json"}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers ,data='{"items":[
    {"sys_id":"04b7e94b4f7b4200086eed18110c7fd","variables":{"acrobat":"true","Additional_software_requirements":"MS Office 2007"},"sysparm_quantity":"3"},
    {"sys_id":"e1be6dcb4f7b4200086eed18110c74c"},
    {"sys_id":"186d917a6fab798057967ddbb3ee4f2","variables":{"new_email":"abel.tuter@example.com"}}
]}
```
"sys_id":"8b3ae7fedc1be1004ece5c08239e522b"
},
{
"sys_id":"962967674ff38200086eeed18110c7e7",
"variables":{
"ergonomic_office":"Office chair should have lumbar support"
}
}}

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

Case-1: if two-step is false:
{
    "result": {
        "cart_id": "0d8c7a32c3211200d68d3b0ac3d3aeba",
        "subtotal_price": "$3,350.00",
        "subtotal_recurring_frequency": "",
        "subtotal_recurring_price": "$300.00",
        "total_title": "Total",
        "monthly": {
            "subtotal_price": "$50.00",
            "subtotal_recurring_frequency": "Monthly",
            "subtotal_recurring_price": "$0.00",
            "total_title": "Total",
            "items": [
                {
                    "catalog_item_id": "90af095bcd38798071a208d710d1b64f",
                    "variables": {},
                    "quantity": "1",
                    "localized_price": "$50.00",
                    "price": "$50.00",
                    "recurring_frequency": "Monthly",
                    "localized_recurring_price": "$0.00",
                    "recurring_price": "$0.00",
                    "item_name": "Belkin iPad Mini Case",
                    "cart_item_id": "cb6e485fc3211200d68d3b0ac3d3ae35"
                }
            ]
        }
    }
}
"delivery_time": "2 Days",
}
],
"show_subtotal_price": "true",
"subtotal_title": "Subtotal"
},
"none": {
"subtotal_price": "-",
"subtotal_recurring_frequency": "",
"subtotal_recurring_price": "$0.00",
"total_title": "Total",
"items": [
{
"catalog_item_id": "186d917a6fab7980575967d8bb3ee4f2",
"variables": {
"Preferred Email address": "abel.tuter@example.com"
},
"quantity": "1",
"localized_price": "$0.00",
"price": "$0.00",
"recurring_frequency": "",
"localized_recurring_price": "$0.00",
"recurring_price": "$0.00",
"item_name": "New Email Account",
"cart_item_id": "17c5241fc3211200d68d3b0ac3d3ae7f",
"delivery_time": "global.Workflow"
},
{
"catalog_item_id": "8b3ae77fedc1be1004e5e5c08239e522b",
"variables": {},
"quantity": "1",
"localized_price": "$0.00",
"price": "$0.00",
"recurring_frequency": "",
"localized_recurring_price": "$0.00",
"recurring_price": "$0.00",
"item_name": "Corp VPN",
"cart_item_id": "97c5241fc3211200d68d3b0ac3d3ae7f",
"delivery_time": "2 Days"
},
{
"catalog_item_id": "e1be6dcb4f7b4200086eedef18110c74c",
"variables": {},
"quantity": "1",
"localized_price": "$0.00",
"price": "$0.00",
"recurring_frequency": "",
"localized_recurring_price": "$0.00",
"recurring_price": "$0.00",
"item_name": "Corp VPN",
"cart_item_id": "97c5241fc3211200d68d3b0ac3d3ae7f",
"delivery_time": "2 Days"
}
"localized_price": "$0.00",
"price": "$0.00",
"recurring_frequency": "",
"localized_recurring_price": "$0.00",
"recurring_price": "$0.00",
"item_name": "External Monitor",
"cart_item_id": "d3c5241fc3211200d68d3b0ac3d3ae7f",
"delivery_time": "2 Days"
},
{
"catalog_item_id": "962967674ff38200086eeed18110c7e7",
"variables": {
    "Please describe any ergonomic requirements\n\n" : "Office chair should have lumbar support"
},
"quantity": "1",
"localized_price": "$0.00",
"price": "$0.00",
"recurring_frequency": "",
"localized_recurring_price": "$0.00",
"recurring_price": "$0.00",
"item_name": "Desk Set Up",
"cart_item_id": "d7c5241fc3211200d68d3b0ac3d3ae7f",
"delivery_time": "5 Days"
}
],
"show_subtotal_price": "false",
"subtotal_title": "Subtotal"
},
"yearly": {
"subtotal_price": "$3,300.00",
"subtotal_recurring_frequency": "Annually",
"subtotal_recurring_price": "$300.00",
"total_title": "Total",
"items": [
{
"catalog_item_id": "04b7e94b4f7b4200086eeed18110c7fd",
"variables": {
    "Adobe Photoshop": "",
    "Adobe Acrobat": "true",
    "Optional Software": "",
    "Additional software requirements": "MS Office 2007"
},
"quantity": "3",
"localized_price": "$0.00",
"price": "$0.00",
"recurring_frequency": "",
"localized_recurring_price": "$0.00",
"recurring_price": "$0.00",
"item_name": "External Monitor",
"cart_item_id": "d3c5241fc3211200d68d3b0ac3d3ae7f",
"delivery_time": "2 Days"
}
,"show_subtotal_price": "false",
"subtotal_title": "Subtotal"
],
"total_title": "Total",
"yearly": {
"subtotal_price": "$3,300.00",
"subtotal_recurring_frequency": "Annually",
"subtotal_recurring_price": "$300.00",
"total_title": "Total",
"items": [
{
"catalog_item_id": "04b7e94b4f7b4200086eeed18110c7fd",
"variables": {
    "Adobe Photoshop": "",
    "Adobe Acrobat": "true",
    "Optional Software": "",
    "Additional software requirements": "MS Office 2007"
}
,"quantity": "3",
"localized_price": "$0.00",
"price": "$0.00",
"recurring_frequency": "",
"localized_recurring_price": "$0.00",
"recurring_price": "$0.00",
"item_name": "Desk Set Up",
"cart_item_id": "d7c5241fc3211200d68d3b0ac3d3ae7f",
"delivery_time": "5 Days"
}]}
Case-2: if two-step is true:
{
    "result": {
        "request_number": "REQ0010001",
        "request_id": "82a7e89fc3211200d68d3b0ac3d3ae0a"
    }
}

**Service Catalog - POST /sn_sc/servicecatalog/items/{item_sys_id}/get_invalid_delegated_users**

Returns a list of users whose request for the specified item cannot be delegated (requested by another user.)

You can call this endpoint prior to calling an endpoint such as Service Catalog API - POST /sn_sc/servicecatalog/items/{sys_id}/add_to_cart or Service Catalog API - POST /sn_sc/servicecatalog/items/{sys_id}/order_now that allow the requisition of a service catalog item on behalf of another person. These calls fail if you pass in a user that is not authorized for the associated item. This endpoint allows you to verify your list of users prior to calling an endpoint that orders the item or tries to add the item to a cart.

**URL format**

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/items/{item_sys_id}/get_invalid_delegated_users
Default URL: /api/sn_sc/servicecatalog/items/{item_sys_id}/
get_invalid_delegated_users

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>item_sys_id</td>
<td>Sys_id of the service catalog item to verify whether the specified user has acquisition rights. Located in the Service Catalog [sc_catalog] table.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_also_request_for</td>
<td>Comma separated string of valid user sys_ids. The endpoint verifies that these users have acquisition rights to the specified service catalog item. Located in the Users [sys_user] table.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Request failed. Bad request given for processing. Refer to the message in the return results for additional information on the error.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errCode</td>
<td>Number associated with the error. Only returned if the endpoint fails.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>errMsg</td>
<td>Description of the error that the endpoint encountered when it failed.</td>
</tr>
<tr>
<td>invalidUsers</td>
<td>Names of the users that are not able to request the specified catalog item.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
</tbody>
</table>

### Example: cURL request

```bash
curl
  "https://instance.servicenow.com/api/sn_sc/servicecatalog/items/04b7e94b4f7b4200086eeed1810c7fd/get_invalid_delegated_users"
  --request POST
  --header "Accept:application/json"
  --header "Content-Type:application/json"
  --data
  "\"sysparm_also_request_for\":\"a8f98bb0eb32010045e1a5115206fe3a,62826bf03710200044e0bfc8bcbe5df1\""
  --user "username":"password"

{
  "result": {
    "invalidUsers": [
      "Abel Tuter"
    ]
  }
}
```

### Example: cURL request

```bash
curl
  "https://instance.servicenow.com/api/sn_sc/servicecatalog/items/04b7e94b4f7b4200086eeed18110c7fd/get_invalid_delegated_users"
  --request POST
  --header "Accept:application/json"
  --header "Content-Type:application/json"
  --data
  "\"sysparm_also_request_for\":\"a8f98bb0eb32010045e1a5115206fe3a,62826bf03710200044e0bfc8bcbe5df1\""
  --user "username":"password"
```
Service Catalog - POST /sn_sc/servicecatalog/items/{sys_id}/order_now

Orders the specified catalog item.

**URL format**

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/items/{sys_id}/order_now

Default URL: /api/sn_sc/servicecatalog/items/{sys_id}/order_now

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the service catalog item. Located in the Catalog Item [sc_cat_item] table. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_also_request_for</td>
<td>Comma-separated string of user sys_ids of other users for which to order the specified item. User sys_ids are located in the User [sys_user] table.</td>
</tr>
<tr>
<td></td>
<td>• You cannot specify more than 50 additional users to receive the item.</td>
</tr>
<tr>
<td></td>
<td>• If any person in the list doesn’t qualify for the item, the entire request is rejected.</td>
</tr>
<tr>
<td></td>
<td>• If the associated item does not have the requested_for variable set, the request is rejected.</td>
</tr>
<tr>
<td></td>
<td>• If the associated item has the requested_for variable set, but the Enable also Request for variable is not configured, or the calling user doesn’t have the proper roles, the request is rejected.</td>
</tr>
<tr>
<td>sysparm_quantity</td>
<td>Required. Quantity of the item. Cannot be a negative number.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>variables</td>
<td>Name-value pairs of all mandatory cart item variables. Mandatory variables are defined on the associated form.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json Or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Request headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Indicates that the quantity value is invalid and the request is not placed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>number</td>
<td>Number of the generated request.</td>
</tr>
<tr>
<td>parent_id</td>
<td>If available, the sys_id of the parent record from which the request is created. For more information, see Configure a parent mapping for a request. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>parent_table</td>
<td>If available, the name of the parent table from which the request is created. Data type: String</td>
</tr>
<tr>
<td>request_id</td>
<td>Sys_id of the order request.                                                Data type: String</td>
</tr>
<tr>
<td>request_number</td>
<td>Request number.                                                              Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the order.                                                        Data type: String</td>
</tr>
<tr>
<td>table</td>
<td>Table name of the request.                                                   Data type: String</td>
</tr>
</tbody>
</table>

**Example: Sample cURL request**

```bash
curl "https://instance.servicenow.com/api/sn_sc/servicecatalog/items/d82ea08510247200964f77ffeec6c4ee/order_now" \
    --request POST \
    --header "Accept:application/json" \
    --header "Content-Type:application/json" \
    --data "{
        sysparm_quantity: 1,
        variables: {
            replacement: 'Yes',
            originalnumber: '1640000',
            data_plan: '500MB'
        }
    }
"
    --user "username":"password"
```

```json
{
    "result": {
        "sys_id": "cf56a3fcdb3a2300e890f71fbf9619ac",
        "number": "REQ0010012",
        "request_number": "REQ0010012",
        "request_id": "cf56a3fcdb3a2300e890f71fbf9619ac",
        "table": "sc_request"
    }
}
```
Example: Sample Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_sc/servicecatalog/items/d82ea08510247200964f77ffe
c6c4ee/order_now'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {"Content-Type":"application/json","Accept":"application/json"}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers, data="{
    sysparm_quantity: 1,
    variables: {
        replacement: 'Yes',
        originalnumber: '1640000',
        data_plan: '500MB'
    }
}
"

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error
Response:',response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Service Catalog - POST /sn_sc/servicecatalog/items/{sys_id}/submit_producer

Creates a record and returns the Table API relative path and redirect URL to access the created record.

**URL format**

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/items/{sys_id}/submit_producer

Default URL: /api/sn_sc/servicecatalog/items/{sys_id}/submit_producer

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the table in which to create the record producer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_view</td>
<td>UI view for which to render the data. Determines the fields returned in the response. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• desktop</td>
</tr>
<tr>
<td></td>
<td>• mobile</td>
</tr>
<tr>
<td></td>
<td>• both</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If you also specify the <code>sysparm_fields</code> parameter, it takes precedent.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>variables</td>
<td>Name-value pairs of the variables of the producer record to define.</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
## Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Indicates successful request and returns the response object consisting of Table API and URL to access the created record.</td>
</tr>
<tr>
<td>400</td>
<td>Request is not processed due to incorrect request message.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Requested resource cannot be found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>number</td>
<td>Number of the ticket associated with the record producer. Data type: String</td>
</tr>
<tr>
<td>parent_id</td>
<td>If available, sys_id of the parent record from which the request is created. For more information, see Configure a parent mapping for a request. Data type: String</td>
</tr>
<tr>
<td>parent_table</td>
<td>Name of the parent table. Data type: String</td>
</tr>
<tr>
<td>record</td>
<td>Table API relative path of the newly created record producer. Data type: String</td>
</tr>
<tr>
<td>redirect_portal_url</td>
<td>URL to which to redirect the Service Portal. Data type: String</td>
</tr>
<tr>
<td>redirect_to</td>
<td>Redirect value. Data type: String</td>
</tr>
<tr>
<td>redirect_url</td>
<td>Redirect URL to access the created record producer. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the record created.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>table</td>
<td>Name of the table on which the request was made.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```
curl
"https://instance.servicenow.com/api/sn_sc/servicecatalog/items/3f1dd0320a0a0b99000a53f7604a2ef9/submit_producer" \
--request POST \--header "Accept:application/json" \--header "Content-Type:application/json" \--data "{'variables':"'urgency':2',"'comments':"This is sample incident'"}"
--user "username":"password"
```

```
{
  "result": {
    "sys_id": "6baf4e1ddba2300e890f71fbf961963",
    "number": "INC0010006",
    "parent_id": null,
    "record": "api/now/table/incident/6baf4e1ddba2300e890f71fbf961963",
    "redirect_portal_url": "",
    "parent_table": "task",
    "redirect_url": "incident.do?sys_id=6baf4e1ddba2300e890f71fbf961963&sysparm_view=ess",
    "table": "incident",
    "redirect_to": ""
  }
}
```

**Example: Python request**

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_sc/servicecatalog/items/3f1dd0320a0a0b99000a53f7604a2ef9/submit_producer'

# Set the user credentials
user = 'username'
```
pwd = 'password'

# Set the proper headers
headers = {"Content-Type":"application/json","Accept":"application/xml"}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers,
                          data="{"variables":{"urgency":"2","comments":"This is sample incident"}}")

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
          response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

Service Catalog - POST /sn/sc/servicecatalog/variables/{sys_id}/display_value
Returns the display value of the specified variable.

URL format

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/variables/{sys_id}/display_value
Default URL: /api/sn_sc/servicecatalog/variables/{sys_id}/display_value
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the variable for which to return the display value.</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_value</td>
<td>Sys_id of the record that contains the display value to return.</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Display value of the variable. Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

```
curl
  "https://instance.servicenow.com/api/sn_sc/servicecatalog/variables/2a3947a10a0a020e007f12648f8790c0/display_value" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --data "\"sysparm_value\":\"62826bf03710200044e0bfc8bcbe5df1\"" \
  --user "username":"password"
```
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url =
'https://instance.servicenow.com/api/sn_sc/servicecatalog/variables/2a3947a10a0a020e007f1248f8790c0/display_value'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Content-Type': 'application/json', 'Accept': 'application/json'}

# Make the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers,
data='{"sysparm_value": "62826bf03710200044e0bfc8bcbe5df1"}')

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": "Abel Tuter (Architect)"
}
```

Service Catalog - PUT /sn_sc/servicecatalog/cart/{cart_item_id}

Updates the specified item in the logged in user's cart.
URL format

Versioned URL: /api/sn_sc/{api_version}/servicecatalog/cart/{cart_item_id}
Default URL: /api/sn_sc/servicecatalog/cart/{cart_item_id}

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>api_version</td>
</tr>
<tr>
<td>cart_item_id</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_quantity</td>
<td>Required. Quantity of the item to update the cart.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>variables</td>
<td>Required if there are mandatory variables specified for the item. Object consisting of all mandatory variable names-value pairs.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Indicates that the request completed successfully and that the item is added to the cart.</td>
</tr>
</tbody>
</table>
| 400         | Following three errors can occur:  
  - *Invalid quantity value*: Indicates that either the `sysparm_quantity` parameter is not provided or contains invalid value.  
  - *Security constraints prevent ordering of Item*: Indicates that either the item ID specified in the path parameters is either invalid or the user does not have access to the item.  
  - *Mandatory Variables are required*: Indicates that one or more of the mandatory variable values is not provided in the request.  |
| 401         | Unauthorized. The user credentials are incorrect or have not been passed.  |
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| cart_id                     | Sys_id of the cart.  
Data type: String |
| <recurring_frequency>       | Describes an items in the cart having a specific recurring frequency.  
Possible values include:  
- daily  
- monthly  
- yearly  
- none (no recurring fees)  
Data type: Object |

```json
"<recurring_frequency>": {  
  "frequency_label": "String",  
  "items": [Array],  
  "show_subtotal_price": "String",  
  "subtotal_price": "String",  
  "subtotal_recurring_frequency": "String",  
  "subtotal_recurring_price": "String",  
  "subtotal_title": "String",  
  "total_title": "String"  
}
```

| <recurring_frequency>.frequency_label | Frequency of the associated recurring fee. If there are no recurring fees for the item, this contains "null".  
Data type: String |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;recurring_frequency&gt;.items</code></td>
<td>List of objects that describe each item associated with the specified recurring frequency category. Data type: Array</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.cart_item_id</code></td>
<td>Sys_id of the item in this cart. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.catalog_item_id</code></td>
<td>Catalog sys_id of the item. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.created.by</code></td>
<td>User that initially placed the item in the cart. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.created_on</code></td>
<td>Date on which the item was initially placed in the cart. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.delivery_time</code></td>
<td>Time that the specified item takes to be delivered. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.item_id</code></td>
<td>Item sys_id. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.item_name</code></td>
<td>Name of the specified item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.localized_price</code></td>
<td>Price of the item (in local currency). Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.localized_recurring_price</code></td>
<td>Recurring fee for the specified item (in local currency). Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.name</code></td>
<td>Item name. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.order_guide</code></td>
<td>Name of the order guide to which this item is associated. If this item is not associated with an order guide this parameter contains an empty string. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.picture</code></td>
<td>File name of the picture of the item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.price</code></td>
<td>Price of the item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.quantity</code></td>
<td>Number of the specified item in the cart. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.recurring_frequency</code></td>
<td>How often the recurring fee is charged for the specified item.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.recurring_price</code></td>
<td>Recurring fee for the specified item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.recurring_subtotal</code></td>
<td>Subtotal of the recurring fees for the item <code>(recurring_price * quantity)</code>. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.recurring_subtotal_price</code></td>
<td>Subtotal of the recurring fees for the item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.short_description</code></td>
<td>Short description of the item. Data type: String</td>
</tr>
</tbody>
</table>
| `<recurring_frequency>.items.show_price` | Flag that indicates whether the price of the item is shown in the cart. Possible values:  
  • true: Show the price of the item in the cart.  
  • false: Do not show the price of the item in the cart.  
  Data type: Boolean |
| `<recurring_frequency>.items.show.quantity` | Flag that indicates whether the item quantity should be shown in the cart. Possible values:  
  • true: Show the item quantity in the cart.  
  • false: Do not show the item quantity in the cart.  
  Data type: Boolean |
| `<recurring_frequency>.items.show_recurring_price` | Flag that indicates whether the recurring fees should appear in the cart. Possible values:  
  • true: Show the recurring fees in the cart.  
  • false: Do not show the recurring fees in the cart.  
  Data type: Boolean |
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;recurring_frequency&gt;.itemssubtotal</code></td>
<td>Subtotal of the number of items. Data type: Number</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.itemssubtotal_price</code></td>
<td>Subtotal for the item (price x quantity). Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.sys_class_name</code></td>
<td>System classification name of the item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.sys_id</code></td>
<td>Sys_id of the item. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.updated_by</code></td>
<td>User that updated the item after it was placed in the cart. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.updated_on</code></td>
<td>Last date/time on which the item was placed in the cart or updated. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.items.variables</code></td>
<td>Name-value pairs of the item variables. Data type: Object</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.showsubtotal_price</code></td>
<td>Flag that indicates whether to show a subtotal for the associated recurring fee category. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.subtotal_price</code></td>
<td>Subtotal of the cost of all items for the recurring frequency category. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.subtotal_recurring_frequency</code></td>
<td>Type of subtotal recurring frequency, such as daily, monthly, etc. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.subtotal_recurring_price</code></td>
<td>Subtotal of the recurring fees of the associated recurring frequency category. Data type: String</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.subtotal_title</code></td>
<td>If the showsubtotal_price is set to true, the text to display in the cart for the associated subtotal. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>&lt;recurring_frequency&gt;.total_title</code></td>
<td>Text to display for the total title for the associated recurring frequency category. Data type: String</td>
</tr>
</tbody>
</table>
| `show_subtotal_price`                     | Flag that indicates whether to show the subtotal price in the cart. Possible values:  
  • true: Show subtotals in the cart.  
  • false: Do not show subtotals in the cart.  
   Data type: String |
| `subtotal_price`                          | Subtotal of all items in the cart. Data type: String                      |
| `subtotal_recurring_frequency`            | Type of subtotal recurring frequency, such as daily, monthly, etc. Same as `<recurring_frequency>.subtotal_recurring_frequency`. Data type: String |
| `subtotal_recurring_price`                | Subtotal of the recurring frequency fees for all items in the cart. Data type: String |
| `subtotal_title`                          | Text to display in the cart for the subtotal field. Data type: String      |
| `total_title`                             | Text to display in the cart for the total field. Data type: String         |

**Example: cURL request**

```
curl 
  "https://instance.servicenow.com/api/sn_sc/v1/servicecatalog/cart/7d138fa21329e300027879d9614b0d56" \ 
  --request PUT \ 
  --header "Accept:application/json" \ 
  --data "{"sysparm_quantity": "2\"}" \ 
  --user "username":"password"

  "result": {
```
"cart_id": "0830db8413a56300397533e2e144b0ba",
"subtotal_price": "$2,397.00",
"subtotal_recurring_frequency": "",
"subtotal_recurring_price": "$0.00",
"total_title": "Total",
"none": {
  "subtotal_price": "$2,397.00",
  "subtotal_recurring_frequency": "",
  "subtotal_recurring_price": "$0.00",
  "total_title": "Total",
  "frequency_label": null,
  "items": [null]
  
  "updated_on": "2018-11-06 17:01:21",
  "catalog_item_id": "07f1666b0bf00300eba42da0d5673ab0",
  "short_description": "Apple iPhone 7",
  "recurring_subtotal_price": "$0.00",
  "delivery_time": "2 Days",
  "sys_class_name": "sc_cat_item",
  "sys_id": "a846872a1369e300027879d96144b07e",
  "price": "$799.00",
  "recurring_price": "$0.00",
  "subtotal_price": "$799.00",
  "variables": {},
  "quantity": "1",
  "item_id": "07f1666b0bf00300eba42da0d5673ab0",
  "recurring_price": "$0.00",
  "show_price": true,
  "item_name": "Apple iPhone 7",
  "cart_item_id": "a846872a1369e300027879d96144b07e",
  "recurring_subtotal": 0,
  "show_quantity": true,
  "created_by": "admin",
  "picture": "e823ee6b0bf00300eba42da0d5673a95.iix",
  "created_on": "2018-11-06 17:01:21",
  "localized_price": "$799.00",
  "subtotal": 799,
  "show_recurring_price": false,
  "name": "Apple iPhone 7",
  "updated_by": "admin",
  "localized_recurring_price": "$0.00",
  "order_guide": ""
},
]
"updated_on": "2018-11-06 17:21:08",
"catalog_item_id": "07f1666b0bf00300eba42da0d5673ab0",
"short_description": "Apple iPhone 7",
"recurring_subtotal_price": "$0.00",
"delivery_time": "2 Days",
"sys_class_name": "sc_cat_item",
"sys_id": "7d138fa21329e300027879d96144b065",
"price": "$799.00",
"recurring_frequency": null,
"subtotal_price": "$1,598.00",
"variables": {},
"quantity": "2",
"item_id": "07f1666b0bf00300eba42da0d5673ab0",
"recurring_price": "$0.00",
"show_price": true,
"item_name": "Apple iPhone7",
"cart_item_id": "7d138fa21329e300027879d96144b065",
"recurring_subtotal": 0,
"show_quantity": true,
"created_by": "admin",
"picture": "e823ee6b0bf00300eba42da0d5673a95.iix",
"created_on": "2018-11-06 16:47:30",
"localized_price": "$799.00",
"subtotal": 1598,
"show_recurring_price": false,
"name": "Apple iPhone7",
"updated_by": "admin",
"localized_recurring_price": "$0.00",
"order_guide": ""
}
},
"show_subtotal_price": "true",
"subtotal_title": "Subtotal"
},
"show_subtotal_price": "true",
"subtotal_title": "Subtotal"
}
}
# Set the request parameters
url =
'https://instance.servicenow.com/api/sn_sc/servicecatalog/cart/7d138fa21329e300027879d9614b065'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {
    "Content-Type": "application/json",
    "Accept": "application/json"
}

# Make the HTTP request
response = requests.put(url, auth=(user, pwd), headers=headers,
data="{"sysparm_quantity": \"1\" }"
)

# Check for HTTP codes other than 201
if response.status_code != 201:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

{
    "result": {
        "cart_id": "0830db8413a56300397533e2e144b0ba",
        "subtotal_price": "$2,397.00",
        "subtotal_recurring_frequency": ",",
        "subtotal_recurring_price": "$0.00",
        "total_title": "Total",
        "none": {
            "subtotal_price": "$2,397.00",
            "subtotal_recurring_frequency": ",",
            "subtotal_recurring_price": "$0.00",
            "total_title": "Total",
            "frequency_label": null,
            "items": [
                {
                    "updated_on": "2018-11-06 17:21:08",
                    "catalog_item_id": "07f1666b0bf003000e42da0d5673ab0",
                    "short_description": "Apple iPhone 7 ",
                }
            ]
        }
    }
}
Service Catalog - PUT /sn_sc/servicecatalog/items/{sys_id}/submit_guide
Retrieves a list of items based on the needs described for an order guide.

URL format
Versioned URL: /api/sn_sc/{api_version}/servicecatalog/items/{sys_id}/submit_guide
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the order guide whose items are to be retrieved. Located in the Order guide [sc_cat_item_guide] table. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>variables</td>
<td>Variables requested for an item. Each variable is a JSON object consisting of name-value pairs. Data type: Array</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>
Request headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Indicates that the request has completed successfully. Returns a list of items based on the passed-in variable list. If the result set does not contain any items, then an empty array is returned.</td>
</tr>
</tbody>
</table>
| 400         | Indicates that the request is invalid. Could be due to one of the following reasons:  
- User does not have access to the item.  
- Mandatory variables of one or more items have not been answered. (Only variables defined as mandatory are honored.) |
| 401         | Unauthorized. The user credentials are incorrect or have not been passed. |
| 500         | Internal error while calculating items depending on options chosen. |
## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalogs</td>
<td>List of objects the catalogs to which the item belongs and each catalog title and sys_id of the catalog.</td>
</tr>
<tr>
<td>catalogs.sys_id</td>
<td>Sys_id of the catalog.</td>
</tr>
<tr>
<td>catalogs.title</td>
<td>Catalog title.</td>
</tr>
<tr>
<td>category</td>
<td>Describes the category that contains this item.</td>
</tr>
<tr>
<td>category.sys_id</td>
<td>Sys_id of the category.</td>
</tr>
<tr>
<td>category.title</td>
<td>Category title</td>
</tr>
<tr>
<td>client_script</td>
<td>List of all the catalog client scripts defined on the catalog item.</td>
</tr>
</tbody>
</table>

**Data type examples:**

```
"catalogs": [{
  "sys_id": "String",
  "title": "String"
}]
```

```
"client_script": {
  "onChange": [Array],
  "onLoad": [Array],
  "onSubmit": [Array]
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>client_script.onChange</td>
<td>List of onChange client scripts associated with the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>client_script.onLoad</td>
<td>List of onLoad client scripts associated with the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>client_script.onSubmit</td>
<td>List of onSubmit client scripts associated with the item.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>data_lookup</td>
<td>Data lookups defined on the catalog item.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>description</td>
<td>Full description of catalog item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>icon</td>
<td>Path of the image that appears as an icon beside the catalog item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>name</td>
<td>Name of the catalog item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>short_description</td>
<td>Text on the service catalog homepage or search results page, or the title on the order form of the catalog item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>show_price</td>
<td>Indicates whether to show the price details. If true, returns the following details.</td>
</tr>
<tr>
<td></td>
<td>• local_currency</td>
</tr>
<tr>
<td></td>
<td>• localized_price</td>
</tr>
<tr>
<td></td>
<td>• localized_recurring_price</td>
</tr>
<tr>
<td></td>
<td>• price</td>
</tr>
<tr>
<td></td>
<td>• price_currency</td>
</tr>
<tr>
<td></td>
<td>• recurring_frequency</td>
</tr>
<tr>
<td></td>
<td>• recurring_price</td>
</tr>
<tr>
<td></td>
<td>• recurring_price_currency</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the catalog item.</td>
</tr>
<tr>
<td>type</td>
<td>Item type. Possible values: catalog item, order guide, record producer item</td>
</tr>
<tr>
<td>variables</td>
<td>Name-value pairs of all the variables associated with the catalog item.</td>
</tr>
<tr>
<td>ui_policy</td>
<td>List of all catalog UI policies defined on the catalog item.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl "https://instance.servicenow.com/api/sn_sc/v1/servicecatalog/items/6690750f4f7b4200086eeed18110c761/submit_guide" \
--request PUT \
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--data "{
   "variables":{
      "hiring_manager":"6816f79cc0a8016401c5a33be04be441",
      "hiring_group":"221f3db5c6112284009f4becd3039cc9",
      "remote":"No",
      "standard_package":"No",
      "laptop_type":"developer"
   }
}
--user "username":"password"
```

```json
{  
   "result": [  
```
The x1 Carbon is Lenovo's lightest ThinkPad yet. It provides a QHD display that fights glare and weighs less than three pounds. Ideal for most computing tasks, and highly mobile.

Technical Specs:
- Intel core i5 processor
- 512GB solid state drive (SSD)
- Backlit keyboard

show_price: true, recurring_price: "$100.00", type: "catalog_item", local_currency: "USD", sys_id: "04b7e94b4f7b4200086eeed18110c7fd", recurring_price_currency: "USD", localized_price: "$1,100.00", price: "$1,100.00", catalogs: [
  {
    "sys_id": "e0d08b13c3300100c8b837659bba8fb4",
    "title": "Service Catalog"
  }
], recurring_frequency: "Annually", name: "Standard Laptop", localized_recurring_price: "$100.00", category: {
  "sys_id": "d258b953c611227a0146101fb1be7c31",
  "title": "Hardware"
}, price_currency: "USD", show_quantity: false, quantity: "1", order: "100", variables: [
  {
    "label": "Optional Software",
    "type": 0,
    "show_price": true,
    "recurring_price": "$100.00",
    "type": "catalog_item",
    "local_currency": "USD",
    "sys_id": "1195c2084f889200086eeed18110c74a.iix",
    "description": "<p class="p1"><font size="3" color="#808080">x1 Carbon</font></p>
<p class="p1"><font size="2" color="#808080" face="arial,helvetica,sans-serif"><span class="s1">The x1 Carbon is Lenovo's lightest ThinkPad yet. It provides a QHD display that fights glare and weighs less than three pounds. Ideal for most computing tasks, and highly mobile. </span></font></p>
<p class="p2"><font size="2" color="#808080" face="arial,helvetica,sans-serif"><span class="s1">Technical Specs:</span></font></p>
<ul class="ul1"><li class="li3"><font size="2" color="#808080"><span class="s1">Intel core i5 processor</span></font></li><li class="li3"><font size="2" color="#808080"><span class="s1">512GB solid state drive (SSD) </span></font></li><li class="li3"><font size="2" color="#808080"><span class="s1">Backlit keyboard</span></font></li></ul>

show_quantity: false, quantity: "1", order: "100", variables: [
  {
    "label": "Optional Software",
    "type": 0,
"mandatory": false,
"displayvalue": "",
"friendly_type": "container_start",
"render_label": true,
"read_only": false,
"children": [
{
   "label": "Adobe Acrobat",
   "type": 7,
   "mandatory": false,
   "displayvalue": "false",
   "friendly_type": "check_box",
   "display_type": "CheckBox",
   "render_label": true,
   "read_only": false,
   "pricing_implications": false,
   "name": "acrobat",
   "attributes": "edge_encryption_enabled=true",
   "id": "90b72d4b4f7b420086eed18110c701",
   "value": false,
   "help_text": "",
   "max_length": 0
},
{
   "label": "Adobe Photoshop",
   "type": 7,
   "mandatory": false,
   "displayvalue": "false",
   "friendly_type": "check_box",
   "display_type": "CheckBox",
   "render_label": true,
   "read_only": false,
   "pricing_implications": false,
   "name": "photoshop",
   "attributes": "edge_encryption_enabled=true",
   "id": "a8b72d4b4f7b420086eed18110c701",
   "value": false,
   "help_text": "",
   "max_length": 0
}
]
The striking design of the LG Cinema Screen incorporates an ultra-narrow bezel, so virtually all you see is picture. The viewing experience is more immersive for your favorite movies and games. Make the most of your time, with an LG Full
HD LED monitor. <p><span style="color: #808080;">Item Specs:</span></p><ul><li>Screen Size: 27"
</li><li>Resolution: 1920 x 1080
</li><li>Aspect Ratio: 16:9
</li><li>Brightness: 250 cd/m²</li><li>Supports Split Screen</li></ul>
"category": {
  "sys_id": "109f0438c6112276003ae8ac13e7009d",
  "title": "Services"
},
"show_quantity": false,
"quantity": "1",
"order": "300",
"variables": [
  {
    "label": "Preferred Email address",
    "type": 6,
    "mandatory": true,
    "displayvalue": "",
    "friendly_type": "single_line_text",
    "display_type": "Single Line Text",
    "render_label": true,
    "read_only": false,
    "name": "new_email",
    "attributes": "edge_encryption_enabled=true",
    "id": "65865e474fbb4200086eeed18110c7dd",
    "value": "",
    "help_text": "",
    "max_length": 0
  }
],
"ui_policy": [],
"client_script": {
  "onChange": [],
  "onSubmit": [],
  "onLoad": []
},
"data_lookup": []
},
{
  "sys_id": "8b3ae7fedc1be1004ece5c08239e522b",
  "short_description": "Remote access to Internal Corporate Systems\n\n",
  "catalogs": [
    {
      "sys_id": "e0d08b13c3330100c8b837659bba8fb4",
      "title": "Service Catalog"
    }
  ],
  "name": "Corp VPN",
  "icon": "11dc8df74f778200086eeed18110c78e.iix",
  "description": "Remote access to Internal Corporate Systems\n\n"
Corp VPN Account

Corp VPN provides VPN (Virtual Private Network) access to all Internal Corporate Systems and to IP Restricted Customer Instances. By utilizing Corp VPN, users will be provided a similar experience to being within one of the Corporate office locations, while offsite.

Please reference the CORP VPN Installation and Support Documentation for instructions on installation, usage and FAQ’s. For all other questions, please contact IT Support.

The facilities organization will set up your new desk with the following:

- **Desk Set Up**

  The facilities organization will set up your new desk with the following:
configuration:</font></p></li><li><font size="3" face="arial, helvetica, sans-serif">Docking Station</font></li><li><font size="3" face="arial, helvetica, sans-serif">Power Station</font></li><li><font size="3" face="arial, helvetica, sans-serif">Phone</font></li><li><font size="3" face="arial, helvetica, sans-serif">Office Chair</font></li></ul>"},
"show_price": false,
"type": "catalog_item",
"category": {
  "sys_id": "",
  "title": null
},
"show_quantity": false,
"quantity": "1",
"order": "900",
"variables": [
  {
  "label": "Please describe any ergonomic requirements\n\n",
  "type": 2,
  "mandatory": false,
  "displayvalue": "",
  "friendly_type": "multi_line_text",
  "display_type": "Multi Line Text",
  "render_label": true,
  "read_only": false,
  "name": "ergonomic_office",
  "attributes": "edge_encryption_enabled=true",
  "id": "b0b9ab674ff38200086eeed18110c755",
  "value": "",
  "help_text": "",
  "max_length": 0
  }
],
"ui_policy": [],
"client_script": {
  "onChange": [],
  "onSubmit": [],
  "onLoad": []
}
]"}
Example: Python request

```python
# Install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/sn_sc/v1/servicecatalog/items/6690750f4f7b4200086eed18110c761/submit_guide'

# Set the user credentials
user = 'username'
pwd = 'password'

# Set the proper headers
headers = {'Content-Type':'application/json','Accept':'application/json'}

# Make the HTTP request
response = requests.put(url, auth=(user, pwd), headers=headers,
    data='{"variables":{"hiring_manager":"6816f79cc0a8016401c5a33be04be441","hiring_group":"221f3db5c6112284009f4becd3039cc9","remote":"No","standard_package":"No","laptop_type":"developer" }}')

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)
```

Same as above

---

Service Order Open API

The Service Order Open API provides a standardized mechanism for placing a service order with all of the necessary order parameters.

To access this API, the Order Management for Telecommunications (sn_ind_tmt_orm) plugin must be activated. For related information, see Service order management. This API is a ServiceNow® implementation of the Open API TMForum TMF641 Service Ordering Management API specification.

This API is provided within the sn_ind_tmt_orm namespace.
The calling user must have the sn_ind_tmt.orm.order_integrator role.

**Extending the Service Order Open API**

The Service Order Open API can be extended using the following script includes. The API should only be extended with an understanding of the consequences of the changes.

- **TMFOrderAPIConstants**: Contains constants and required parameter information.
- **TMFServiceOrderAPIUtil**: Contains functions to handle POST requests.
- **TMFServiceOrderGetAPIUtil**: Contains functions to handle GET requests.
- **ServiceOrderExtensionOOB**: Contains helper functions that support functions in TMFServiceOrderAPIUtil and TMFServiceOrderGetAPIUtil.
- **ServiceOrderProcessor**: An empty script include file. Use this file to define any functions that you want to override from ServiceOrderExtensionOOB.

Extend the Service Order Open API to make the following customizations.

**Required parameters**

To change which request body parameters are required or not required, update the TMFOrderAPIConstants script include.

- **TMFOrderAPIConstants.SCHEMA.CREATE_SERVICE_ORDER**: Sets the required request body parameters to create a service order.

**Request body validation**

To perform additional validation on the request body, override ServiceOrderExtensionOOB functions. ServiceOrderExtensionOOB contains the following four helper functions that return `true` by default.

- **validatePostRequest()** - Called by `processPostOrder()` in TMFServiceOrderAPIUtil.
- **validateServiceObj()** - Called by `processPostOrder()` in TMFServiceOrderAPIUtil.
- **validateRelatedPartyObj()** - Called by `processPostOrder()` in TMFServiceOrderAPIUtil.
- **validateGetRequest()** - Called by `processGetOrder()` in TMFServiceOrderGetAPIUtil.

If a helper function returns `false`, it stops the API operation. To apply custom validation, override ServiceOrderExtensionOOB helper functions by creating functions with identical
names and parameters in ServiceOrderProcessor. These new ServiceOrderProcessor functions will be called by TMFServiceOrderAPIUtil and TMFServiceOrderGetAPIUtil to replace the default ServiceOrderExtensionOOB helper functions. In this example, a function in ServiceOrderProcessor overrides a default function in ServiceOrderExtensionOOB to perform validation on the serviceOrderItem attribute.

```javascript
// ServiceOrderProcessor
var ServiceOrderProcessor = Class.create();
ServiceOrderProcessor.prototype =
Object.extendsObject(ServiceOrderExtensionOOB, {
  // Define overriding functions here
  // Function name and parameters must be identical to the function it overrides
  validatePostRequest: function(orderObject, details) {
    // Returning false terminates the POST request
    // Make sure to push error message in details array in case of error
    if (gs.nil(orderObject.serviceOrderItem)) {
      details.push(new TMFCommonOrderAPIUtil().getErrorDetailsObj(TMFOrderAPIConstants.MESSAGES.MISSING_ORDER_ITEM, '/'));
      return false;
    }
    return true;
  },

type: 'ServiceOrderProcessor'
});
```

**Additional REST operations**

To create additional operations beyond the existing GET and POST operations, create additional scripted REST resources for the Service Order Open API. The logic of the new scripted REST resources should be consistent with the existing operations. Define functions for the new operations in a new script include.

**Field mapping**

When creating records, the API maps request body parameters to record fields. When retrieving records, the API maps record fields to response object attributes.

ServiceOrderExtensionOOB contains the following functions to map a POST request body to a GlideRecord.
• transformOrderGr()
• transformOrdLineItemGr()
• transformCustLineItmContact()
• transformOrderItemChar()

ServiceOrderExtensionOOB contains the following functions to map a GlideRecord to a response object for GET or POST requests.

• transformPostOrderResponse()
• transformGetOrderResponse()
• transformServiceObj()
• transformRelatedPartyCustomerLineItem()
• transformOrderItemRelationship()
• transformGetOrdLineItmResponse()
• transformServiceCharacteristics()
• transformServiceSpecification()

Customize field mappings to add and retrieve data for additional fields, or to change the default mappings for fields. To customize the field mappings, override ServiceOrderExtensionOOB mapping functions by creating functions with identical names and parameters in ServiceOrderProcessor. These new ServiceOrderProcessor functions will be used by TMFServiceOrderAPIUtil and TMFServiceOrderGetAPIUtil to replace the default ServiceOrderExtensionOOB mapping functions. In this example, two functions in ServiceOrderProcessor override the default functions in ServiceOrderExtensionOOB to create mappings for the external_id and sys_id fields.

```javascript
// ServiceOrderProcessor
var ServiceOrderProcessor = Class.create();
ServiceOrderProcessor.prototype =
Object.extendsObject(ServiceOrderExtensionOOB, {
    // Define overriding functions here
    transformOrderGr: function(requestObject, orderGr) {
        orderGr.external_id = requestObject.externalId;
        return orderGr;
    }
});
```
Service Order Open – GET sn_ind_tmt_orm/serviceorder/{id}

Retrieves information for a specified service order.

This operation retrieves a customer order record with related records as dictated by query parameters and filter criteria.

This API retrieves service order information from the following related tables:

- Order [sn_ind_tmt_orm_order]
- Order Line Item [sn_ind_tmt_orm_order_line_item]
- Order Characteristic Value [sn_ind_tmt_orm_order_characteristic_value]
- Order Line Item Contact [sn_ind_tmt_orm_order_line_item_contact]

**URL format**

/api/sn_ind_tmt_orm/serviceorder/{id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Sys_id of the service order to retrieve. Located in the Order [sn_ind_tmt_orm_order] table. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>Comma-separated list of fields to return in the response. If this value is not passed in, all fields are returned. Possible values:</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>expectedCompletionDate</td>
<td></td>
</tr>
<tr>
<td>id</td>
<td></td>
</tr>
<tr>
<td>orderDate</td>
<td></td>
</tr>
<tr>
<td>ponr</td>
<td></td>
</tr>
<tr>
<td>serviceOrderItem</td>
<td></td>
</tr>
<tr>
<td>requestedCompletionDate</td>
<td></td>
</tr>
<tr>
<td>requestedStartDate</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. Can be for any of the following reasons:</td>
</tr>
<tr>
<td></td>
<td>- Missing query parameter</td>
</tr>
<tr>
<td></td>
<td>- Invalid URI</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>expectedCompletionDate</td>
<td>Date on which the order is to be completed.</td>
</tr>
<tr>
<td>id</td>
<td>Sys_id of the requested service order.</td>
</tr>
<tr>
<td>note</td>
<td>Array of objects that describe additional notes made by the customer when ordering.</td>
</tr>
<tr>
<td>note.author</td>
<td>User name of the person that authored the associated note.</td>
</tr>
<tr>
<td>note.date</td>
<td>Date that the note was created.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>note.text</td>
<td>Additional notes/comments made by the customer while ordering.</td>
</tr>
<tr>
<td>ponr</td>
<td>Flag that indicates whether the point of no return for the order has been reached. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Point of no return for the order has been reached. The order can't be modified or canceled while fulfillment is in progress.</td>
</tr>
<tr>
<td></td>
<td>• false: Point of no return for the order has not been reached. The order can be modified or canceled.</td>
</tr>
<tr>
<td>relatedParty</td>
<td>Array of objects that describe the role linked to an OrderLineItemContact. Data type: Array</td>
</tr>
<tr>
<td>relatedParty.id</td>
<td>Sys_id of the account or customer contact associated with the order. Located in the account [customer_account], contact [customer_contact], or Order line item contact [sn_ind_tmt_orm_order_line_item_contact] table. Data type: String</td>
</tr>
<tr>
<td>relatedParty.name</td>
<td>Name of the account or customer. Data type: String</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| relatedParty.@referredType  | Type of customer. Possible values:  
- Customer  
- Customer Contact  
Data type: String                                                                                                                                                                                                                                                                                                     |
| relatedParty.@type          | Part of TMF Open API standard. Annotation for order line item contact. This value is always `RelatedParty`. This information is not stored.  
Data type: String                                                                                                                                                                                                                                                                                                         |
| requestedCompletionDate     | Delivery date requested by the customer.  
Data type: String                                                                                                                                                                                                                                                                                                                             |
| requestedStartDate          | Order start date requested by the customer.  
Data type: String                                                                                                                                                                                                                                                                                                                               |
| serviceOrderItem            | Array of objects that describe items associated with the service order and their associate action.  
Data type: Array  

```json
"serviceOrderItem": [  
  {  
    "action": "String",  
    "id": "String",  
    "orderRelationship": [Array],  
    "place": {Object},  
    "ponr": Boolean,  
    "relatedParty": {Object},  
    "service": {Object},  
    "state": "String",  
    "@type": "String",  
    "version": "String"  
  }  
]
```

serviceOrderItem.action  | Action to carry out on the service. Actions are defined on the Choice List tab.   

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>in the Action Dictionary Entry field of the sn_ind_tmt_orm_order_line_item table. Possible values: add, change, delete. For details, see Action types. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.id</td>
<td>Unique identifier of the line item. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.orderRelationship</td>
<td>Array of objects that describe the parent/child relationship between order items. Data type: Array.</td>
</tr>
<tr>
<td>serviceOrderItem.orderRelationship.id</td>
<td>Same as the serviceOrderItem.id. Used for parent/child relationship. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.orderRelationship.relationshipType</td>
<td>Type of relationship between the two line items. This information is used to identify relationship hierarchy. Possible values: HasChild, HasParent. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.place</td>
<td>Maps of the locations on which to install the service. Data type: Object</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>serviceOrderItem.place.id</td>
<td>Sys_id of the associated location record in the Location [cmn_location] table. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.place.@type</td>
<td>Part of TMF Open API standard. Annotation for order line item contact. This value is always <code>Place</code>. This information is not stored. Data type: String</td>
</tr>
</tbody>
</table>
| serviceOrderItem.ponr                    | Flag that indicates whether the point of no return for the order line item has been reached. Valid values:  
  - true: Point of no return for the order line item has been reached. The order line item can't be modified or cancelled while fulfillment is in progress.  
  - false: Point of no return for the order line item has not been reached. The order line item can be modified or cancelled.  
Data type: Boolean                                                                             |
| serviceOrderItem.relatedParty            | Array of objects that describe the role linked to an OrderLineItemContact. Data type: Array                                                                                                                                                                                                                                           |

```json
"relatedParty": {
  "email": "String",
  "firstName": "String"
  "id": "String",
  "lastName": "String",
  "phone": "String",
  "@referredType": "String",
}
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceOrderItem.relatedParty.email</td>
<td>Email address of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.relatedParty.firstName</td>
<td>First name of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.relatedParty.id</td>
<td>Sys_id of the account or customer contact associated with the order. Located in the account [customer_account], contact [customer_contact], or Order line item contact [sn_ind_tmt_orm_order_line_item_table].Order Line Item Contact [sn_ind_tmt_orm_order_line_item_table].</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.relatedParty.lastName</td>
<td>Last name of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.relatedParty.phone</td>
<td>Business phone number of the contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.relatedParty.@referredType</td>
<td>Required. Type of customer. Possible values: OrderedLineItemContact</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.relatedParty.@type</td>
<td>Part of TMF Open API standard. Annotation for order line item contact. This value is always RelatedParty. Information is not stored.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service</td>
<td>Description of the instance details of the service purchased by the customer</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;service:&quot;</td>
<td>{</td>
</tr>
<tr>
<td>&quot;serviceCharacteristic&quot;: [Array],</td>
<td>Only service characteristics with a current value different from the previous value are returned.</td>
</tr>
<tr>
<td>&quot;serviceSpecification&quot;: (Object),</td>
<td>Data type: Array</td>
</tr>
<tr>
<td>&quot;serviceType&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;state&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;@type&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceCharacteristic</td>
<td>Array of objects that describe the characteristics of the associated service.</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceCharacteristic.value</td>
<td>Characteristic option values associated with the service. For additional information on characteristic option values, see Product catalog data model.</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceCharacteristic.valueType</td>
<td>Type of characteristic value.</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceSpecification</td>
<td>Description of the service specification associated with the service.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceOrderItem.service.serviceSpecification.id</td>
<td>Sys_id or of the associated service specification. Located in the sys_id or external_id field of the Service Specification [sn_prd_pm_service_specification] table. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceSpecification.name</td>
<td>Name of the service specification to associate with the service. Located in the Service Specification [sn_prd_pm_service_specification] table. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceSpecification.@type</td>
<td>Part of TMF Open API standard. Annotation for order line item contact. This value is always ServiceSpecificationRef. This information is not stored. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceSpecification.version</td>
<td>Service specification version. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceType</td>
<td>Service specification type. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.state</td>
<td>State of the sold product. For example, new. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.state</td>
<td>State of the associated service order item. For information, see Service order states. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.@type</td>
<td>Part of TMF Open API standard. Annotation for the service. This value is always ServiceOrderItem. This information is not stored. Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

The following example shows how to specify a service order to retrieve data.

curl
  "https://instance.service-now.com/api/sn_ind_tmt_orm/serviceorder/5e3dee59db74f0504ae3c9db13961944" \
  --request GET \
  --header "Accept:application/json" \
  --user 'username':'password'

The response body provides details of the specified service order.

```json
{
  "id": "4f2af65ac3a030106e2473ce3640ddcf",
  "ponr": "false",
  "expectedCompletionDate": "2018-01-15T09:37:40.000Z",
  "requestedCompletionDate": "2018-01-15T09:37:40.000Z",
  "requestedStartDate": "2018-01-15T09:37:40.000Z",
  "note": [  
    {  
      "author": "System Administrator",
      "date": "2021-06-07T14:53:04.000Z",
      "text": "This is a TMF service order illustration no 2"
    },
    {  
      "author": "System Administrator",
      "date": "2021-06-07T14:53:03.000Z",
      "text": "This is a TMF service order illustration"
    }
  ],
  "serviceOrderItem": [  
    {  
      "id": "100",
      "ponr": "false",
      "action": "add",
      "service": {  
        "@type": "Service",
        "state": "",
        "serviceType": "cfs"
      }
    }
  ]
}```
"serviceCharacteristic": [ 
{
   "name": "Firewall coverage",
   "valueType": "choice",
   "value": "Premium(Up to 50 sites)",
   "previousValue": ""
},
{
   "name": "Firewall Security",
   "valueType": "choice",
   "value": "Standard",
   "previousValue": ""
},
{
   "name": "Administration Support",
   "valueType": "choice",
   "value": "Basic( Customer Managed)",
   "previousValue": ""
}
],
"serviceSpecification": {
   "id": "f99546ff07266010a7955b7e0ad300a8",
   "name": "Managed Firewall Service",
   "version": "1",
   "@type": "ServiceSpecificationRef"
},
"place": {
   "id": "920cf6ac73d423002728660c4cf6a799",
   "@type": "Place"
},
"relatedParty": [ 
{
   "id": "972af65ac3a030106e2473ce3640ddd1",
   "firstName": "Mike",
   "lastName": "Hudson",
   "email": "mike@example.com",
   "phone": "1234567890",
   "@type": "RelatedParty",
   "@referredType": "OrderLineItemContact"
}
],
"state": "new",
"version": "1"
Service Order Open – POST sn_ind_tmt_orm/serviceorder

Enables creating, updating, or deleting a customer order or customer order line item for service orders.

Use this API to create, update, or disconnect a customer order.

When an add action is received and accepted, a new record is created in the following tables:

- Order [sn_ind_tmt_orm_order]
- Order Line Item [sn_ind_tmt_orm_order_line_item]
- Order Characteristic Value [sn_ind_tmt_orm_order_order_characteristic_value]
- Order Line Item Contact [sn_ind_tmt_orm_order_line_item_line_item_contact]

URL format

/api/sn_ind_tmt_orm/serviceorder
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| externalId | Unique order number for the external service order.  
                   Data type: String  
                   Table/field updated: sn_ind_tmt_orm_order |
| note     | Array of objects that describe notes made by the customer when ordering.  
                   Data type: Array  
                   Table/field updated: sn_ind_tmt_orm_order |
| note.text | Required. Additional notes/made by the customer while ordering.  
                   Data type: String  
                   Table/field updated: sn_ind_tmt_orm_order/comments  
                   Default: Blank string |
| orderDate | Date of the customer order request.  
                   Data type: String |
| relatedParty | Array of objects that describe role linked to an OrderLineItem |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.  
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
## Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relatedParty.id</td>
<td>Required. Sys_id of the account or customer contact associated with the order. Located in the Account [customer_account] or Contact [customer_contact] table. Data type: String Table updated: sn_ind_tmt_orm_order</td>
</tr>
<tr>
<td>relatedParty.name</td>
<td>Name of the account or customer. Data type: String</td>
</tr>
</tbody>
</table>
| relatedParty.@referredType | Type of customer. Possible values:  
  - Customer  
  - CustomerContact Data type: String                                                                                                              |
<p>| relatedParty.@type    | Part of TMF Open API standard. Annotation for order line item contact. This value is always RelatedParty. This information is not stored. Data type: String                                                                 |
| requestedCompletionDate | Delivery date requested by the customer. Data type: String Table/field updated: sn_ind_tmt_orm_order/expected_end_date Default: Blank string                                                                 |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestedStartDate</td>
<td>Order start date requested by the customer.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Table/field updated: sn_ind_tmt_orm_order/expected_start_date</td>
</tr>
<tr>
<td></td>
<td>Default: Blank string</td>
</tr>
<tr>
<td>serviceOrderItem</td>
<td>Required. Array of objects that describe items associated with the service order and their associate action.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;serviceOrderItem&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;action&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;orderRelationship&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;place&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;relatedParty&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;service&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;@type&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;version&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td>Table/field updated: sn_ind_tmt_orm_order_line_item</td>
</tr>
<tr>
<td>serviceOrderItem.action</td>
<td>Required. Action to carry out on the service. Possible actions defined on the Choice List tab in the Action Dictionary Entry field of the sn_ind_tmt_orm_order_line_item table.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• add</td>
</tr>
<tr>
<td></td>
<td>• change</td>
</tr>
<tr>
<td></td>
<td>• delete</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceOrderItem.id</td>
<td>Required. Unique identifier of the service provided. Located in the <code>sn_ind_tmt.orm_external_id</code> field of the Service Inventory table.</td>
</tr>
<tr>
<td>serviceOrderItem.orderRelationship</td>
<td>Required. Array of objects that describe the parent/child relationship between order items.</td>
</tr>
<tr>
<td>serviceOrderItem.orderRelationship.id</td>
<td>Required. Same as the <code>productOrderItem.id</code> value. Used for parent/child relationship.</td>
</tr>
</tbody>
</table>

For details, see [Action types](#)  
Data type: String  
Table/field updated: `sn_ind_tmt.orm_order_line_item/action`  

"orderRelationship": {  
  
  "id": "String",  
  "relationshipType": "String"  
}  

Table/field updated: `sn_ind_tmt.orm_order_line_item/orderRelationship.id`  

Default: Blank string
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceOrderItem.orderRelationship.relationshipType</td>
<td>Required. Type of relationship between the two line items. This information is used to identify relationship hierarchy. Possible values:</td>
</tr>
</tbody>
</table>
|                                                | • HasChild  
|                                                | • HasParent  
|                                                | Data type: String                                                                                                                          |
| serviceOrderItem.place                         | Maps of the locations on which the service is to be performed.  
|                                                | Data type: Object  
|                                                | "place": {  
|                                                |   "id": "String",  
|                                                |   "@type": "String"  
|                                                | }                                                                                                                                          |
| serviceOrderItem.place.id                     | Required. Sys_id of the associated location record in the [cmn_location] table.  
|                                                | Data type: String  
|                                                | Table/field updated:  
|                                                | sn_ind_tmt_orm_order_line_item  
|                                                | Default: Blank string                                                                                                                      |
| serviceOrderItem.place.@type                  | Part of TMF Open API standard. Annotation for order line item place.  
|                                                | Data type: String                                                                                                                          |
| serviceOrderItem.relatedParty                 | Array of objects that describe the role linked to an OrderLineItemContact.  
|                                                | Data type: Array  
|                                                | "relatedParty": {  
|                                                |   "email": "String",  
|                                                |   "firstName": "String"  
|                                                |   "id": "String",  
|                                                |   "lastName": "String"  
<p>|                                                | }                                                                                                                                          |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceOrderItem.relatedParty.email</td>
<td>Email address of the contact.</td>
</tr>
<tr>
<td>serviceOrderItem.relatedParty.firstName</td>
<td>First name of the contact.</td>
</tr>
<tr>
<td>serviceOrderItem.relatedParty.id</td>
<td>Required. Sys_id of the account or customer contact associated with the order. Located in the Order Line Item Contact [sn_ind_tmt_orm_order_line_item_contact] table.</td>
</tr>
<tr>
<td>serviceOrderItem.relatedParty.lastName</td>
<td>Last name of the contact.</td>
</tr>
<tr>
<td>serviceOrderItem.relatedParty.phone</td>
<td>Business phone number of the contact.</td>
</tr>
</tbody>
</table>
| serviceOrderItem.relatedParty.@referredType | Required. Type of customer. Possible values:
• OrderLineItemContact |
| serviceOrderItem.relatedParty.@type      | Part of TMF Open API standard. Annotation for order line item. This value is always RelatedParty. Information is not stored. |
| serviceOrderItem.service                 | Required. Description of the details of the service purchased by the customer. |
## Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Data type:</strong> Object</td>
</tr>
<tr>
<td></td>
<td>&quot;service&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;serviceCharacteristic&quot;: [Array</td>
</tr>
<tr>
<td></td>
<td>&quot;serviceSpecification&quot;: {Object</td>
</tr>
<tr>
<td></td>
<td>@type&quot;: &quot;String&quot; }</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td><strong>Table/field updated:</strong></td>
<td>sn_ind_tmt orm_order_line_item</td>
</tr>
<tr>
<td>serviceOrderItem.service.id</td>
<td>Unique identifier of the service sold. This value can be the sys_id or external ID in the Product Inventory [sn_ind_tmt orm_product_inventory] table. This value is only used for change orders.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> String</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceCharacteristic</td>
<td>Array of objects that describe characteristics of the associated service. Only service characteristics with a current value different from the previousValue are returned.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> Array</td>
</tr>
<tr>
<td></td>
<td>&quot;serviceCharacteristic&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;previousValue&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceCharacteristic.name</td>
<td>Name of the characteristic record to associate with the service. Located in the Characteristic [sn_prd_pm_characteristic] table.</td>
</tr>
<tr>
<td></td>
<td><strong>Data type:</strong> String</td>
</tr>
</tbody>
</table>
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceOrderItem.service.serviceCharacteristic.previousValue</td>
<td>Previous characteristic option values if the update is for change order. The request is a change order if the <code>serviceOrderItem.action</code> parameter is other than <code>add</code>. For additional information on characteristic option values, see Product catalog data model.</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceCharacteristic.value</td>
<td>Characteristic option values associated with the service. For additional information on characteristic option values, see Product catalog data model.</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceSpecification</td>
<td>Required. Description of the service specification associated with the service.</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceSpecification.id</td>
<td>Sys_id or of the associated service specification. Located in the sys_id or external_id field of the Service Specification [sn_prd_pm_service_specification].</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceSpecification.name</td>
<td>Required. Name of the service specification to associate with. Located in the Service Specification [sn_prd_pm_service_specification].</td>
</tr>
</tbody>
</table>
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceOrderItem.service.serviceSpecification.@type</td>
<td>Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.@type</td>
<td>Data type: String</td>
</tr>
<tr>
<td>@type</td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
| 201         | Successful. If there are any issues with the characteristics or characteristics option information, the endpoint stores the following comments in the work notes fields of the associated Customer Order Line Item record:  
  - The following Order Item characteristics does not exist: Review specification `<specification.name>` and correct the characteristic and characteristic option in the order line item prior to approving the order.  
  - Order Item characteristic: `<characteristic.name>` with characteristic value: `<characteristic.value>` is invalid. Correct the characteristic values before approving the order. |
| 400         | Bad Request. Could be any of the following reasons:  
  - Invalid payload: Request body missing. - Payload was not passed in the request body.  
  - Invalid payload: serviceOrderItem missing. - Service order line item object or JSON is missing.  
  - Invalid payload: serviceOrderItem ID is missing. – The ID is missing from the service order line item.  
  - Invalid payload: serviceOrderItem action is missing. – The service object or JSON is missing from the payload.  
  - Invalid payload: serviceOrderItem service is missing. – The service property is missing from the service order line item. |
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>•</td>
<td>Invalid payload: serviceSpecification is missing. - The service specification object or JSON in the service order line item is missing from the payload.</td>
</tr>
<tr>
<td>•</td>
<td>Invalid payload: serviceSpecification ID is missing. - The id parameter in the service order line item of the service specification object is missing from the payload.</td>
</tr>
<tr>
<td>•</td>
<td>Invalid payload: Service specification does not exist. - The service specification in the service order line item is not valid.</td>
</tr>
<tr>
<td>•</td>
<td>Invalid payload: Product Inventory does not exist. - In a change or delete order (i.e., action = change or delete), the sold service sent is not present in the system.</td>
</tr>
<tr>
<td>•</td>
<td>Invalid payload: Product inventory ID is missing. – In a change or delete order, the Sold Product ID is missing from the payload.</td>
</tr>
<tr>
<td>•</td>
<td>Invalid payload: Sold Product is inactive. – In a change or delete order, the Sold Product sent in the payload is not in active state.</td>
</tr>
<tr>
<td>•</td>
<td>Invalid payload: Customer Account is missing. – In the payload, related party customer object is missing.</td>
</tr>
<tr>
<td>•</td>
<td>Invalid payload: Customer Account does not exist. – The related party customer provided in payload is not present in the system.</td>
</tr>
<tr>
<td>•</td>
<td>Invalid payload: Order creation failed. - Not able to create the requested order.</td>
</tr>
</tbody>
</table>

| 404         | Not found. The requested item was not found. |

---

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>externalId</td>
<td>Unique order number for the service order. Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>Sys_id of the service order. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>note</td>
<td>Array of objects that describe additional notes made by the customer when ordering. Data type: Array</td>
</tr>
<tr>
<td>note.text</td>
<td>Additional notes/comments made by the customer while ordering. Data type: String</td>
</tr>
<tr>
<td>relatedParty</td>
<td>Array of objects that describe the party role linked to an OrderLineItemContact. Data type: Array</td>
</tr>
<tr>
<td>relatedParty.id</td>
<td>Sys_id of the account or customer contact associated with the order. Located in the account [customer_account], contact [customer_contact], or Order line item contact [sn_ind_tmt_orm_order_line_item_contact] table. Account [customer_account], Contact [customer_contact]. Data type: String</td>
</tr>
<tr>
<td>relatedParty.name</td>
<td>Name of the account or customer. Data type: String</td>
</tr>
<tr>
<td>relatedParty.@referredType</td>
<td>Type of customer. Possible values: ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
|                          | • Customer  
                        • CustomerContact  
                          Data type: String                                                                 |
| relatedParty.@type       | Part of TMF Open API standard. Annotation for order line item. This value is always `RelatedParty`. Information is not stored.  
                          Data type: String                                                                 |
| requestedCompletionDate   | Delivery date requested by the customer.  
                          Data type: String                                                                 |
| requestedStartDate       | Order start date requested by the customer.  
                          Data type: String                                                                 |
| serviceOrderItem         | Array of objects that describe the items associated with the service order and their associate action.  
                          Data type: Array  
                          
                          "serviceOrderItem": [  
                          {  
                          "action": "String",  
                          "id": "String",  
                          "orderRelationship": [Array],  
                          "place": {Object},  
                          "relatedParty": {Object},  
                          "service": {Object},  
                          "@type": "String",  
                          "version": "String"  
                          }  
                          ]  
<pre><code>                      |
</code></pre>
<p>| serviceOrderItem.action  | Action to carry out on the service. Possible actions are defined on the Choice List tab in the Action Dictionary Entry of the sn_ind_tmt_or... |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• add</td>
<td></td>
</tr>
<tr>
<td>• change</td>
<td></td>
</tr>
<tr>
<td>• delete</td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>serviceOrderItem.id</td>
<td>Unique identifier of the line item.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>serviceOrderItem.orderRelationship</td>
<td>Array of objects that describe the parent/child relationship between order items.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td>&quot;orderRelationship:&quot; [</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;id&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;relationshipType&quot;: &quot;String&quot; }</td>
<td></td>
</tr>
<tr>
<td>]</td>
<td></td>
</tr>
<tr>
<td>serviceOrderItem.orderRelationship.id</td>
<td>Same as the serviceOrderItem.id value. Used for parent/child relationship.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>serviceOrderItem.orderRelationship.relationshipType</td>
<td>Type of relationship between order items. This information is used to identify relationship hierarchy. Possible values:</td>
</tr>
<tr>
<td>• HasChild</td>
<td></td>
</tr>
<tr>
<td>• HasParent</td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>serviceOrderItem.place</td>
<td>Maps of the locations on which to install the service.</td>
</tr>
<tr>
<td>Data type: Object</td>
<td></td>
</tr>
<tr>
<td>&quot;place&quot;: {</td>
<td></td>
</tr>
<tr>
<td>&quot;id&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>@type&quot;: &quot;String&quot; }</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>serviceOrderItem.place.id</td>
<td>Sys_id of the associated location record in the Location [cmn_location] table. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.place.@type</td>
<td>Part of TMF Open API standard. Annotation for order line item contact. This value is always <code>Place</code>. This information is not stored. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.relatedParty</td>
<td>Array of objects that describe the role linked to an OrderLineItem contact. Data type: Array</td>
</tr>
<tr>
<td>serviceOrderItem.service.relatedParty.email</td>
<td>Email address of the contact. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.relatedParty.firstName</td>
<td>First name of the contact. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.relatedParty.id</td>
<td>Sys_id of the account or customer contact associated with the order. Located in the account [customer_account], contact [customer_contact], or Order line item contact [sn_ind_tmt_orm_order_line_item_contact] table. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.relatedParty.lastName</td>
<td>Last name of the contact. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>serviceOrderItem.service.relatedParty.phone</td>
<td>Business phone number of the contact. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.relatedParty.@referredType</td>
<td>Required. Type of customer. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• OrderLineItemContact</td>
</tr>
<tr>
<td>serviceOrderItem.service.relatedParty.@type</td>
<td>Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service</td>
<td>Description of the instance details of the service purchased by the customer</td>
</tr>
<tr>
<td></td>
<td>&quot;service&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;serviceCharacteristic&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;serviceSpecification&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>'@type': &quot;String&quot;</td>
</tr>
<tr>
<td>serviceOrderItem.service.id</td>
<td>Unique identifier of the service sold. This value can be the sys_id or</td>
</tr>
<tr>
<td></td>
<td>external ID in the Product Inventory table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceCharacteristic</td>
<td>Array of objects that describe characteristics of the associated service.</td>
</tr>
<tr>
<td></td>
<td>Only service characteristics with a current value different from the</td>
</tr>
<tr>
<td></td>
<td>previous value are returned.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceOrderItem.service.serviceCharacteristic.name</td>
<td>Name of the characteristic record to associate with the service. Located in the Characteristic [sn_prd_pm_characteristic] table. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceCharacteristic.previousValue</td>
<td>Previous characteristic option values if the update is for a change order. The request is a change order if the serviceOrderItem.action parameter is other than add. For additional information on characteristic option values, see Product catalog data model. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceCharacteristic.value</td>
<td>Characteristic option values associated with the service. For additional information on characteristic option values, see Product catalog data model. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceSpecification</td>
<td>Required. Description of the service specification associated with the service. Data type: Object</td>
</tr>
</tbody>
</table>

Updates the service_specification field in the Order Line Item [sn_ind_tmt_orm_order_line_item].
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceOrderItem.service.serviceSpecification.id</td>
<td>Sys_id or of the associated service specification. Located in the sys_id or external_id field of the Service Specification [sn_prd_pm_service_specification] table. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceSpecification.name</td>
<td>Name of the service specification to associate with the service. Located in the Service Specification [sn_prd_pm_service_specification] table. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.service.serviceSpecification.@type</td>
<td>Part of TMF Open API standard. Annotation for order line item contact. This value is always ServiceSpecificationRef. This information is not stored. Data type: String</td>
</tr>
<tr>
<td>serviceOrderItem.@type</td>
<td>Part of TMF Open API standard. Annotation for the service. This value is always ServiceOrder. This information is not stored. Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>State of the associated service order. For information, see Service order states. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

The following example shows how to create a customer service order.

```bash
curl "https://instance.service-now.com/api/sn_ind_tmt_orm/serviceorder" \
  --request POST \
  --header "Accept: application/json" \
  --header "Content-Type: application/json" \
  --data "{
    "externalId": "BSS748",
    "requestedStartDate": "2018-01-15T09:37:40.508Z",
    "requestedCompletionDate": "2018-01-15T09:37:40.508Z",
    "@type": "ServiceOrder"
  }"
```
"note": [
  {
    "text": "This is a TMF service order illustration"
  },
  {
    "text": "This is a TMF service order illustration no 2"
  }
],
"serviceOrderItem": [
  {
    "id": "100",
    "action": "add",
    "@type": "ServiceOrderItem",
    "service": {
      "@type": "CFS",
      "serviceCharacteristic": [
        {
          "name": "Firewall Security",
          "value": "Standard"
        },
        {
          "name": "Firewall coverage",
          "value": "Premium(Up to 50 sites)"
        },
        {
          "name": "Administration Support",
          "value": "Basic( Customer Managed)"
        }
      ],
      "serviceSpecification": {
        "id": "f99546ff07266010a7955b7e0ad300a8",
        "name": "Managed Firewall Service",
        "@type": "Managed Firewall Service"
      }
    },
    "relatedParty": [
      {
        "firstName": "Mike",
        "lastName": "Hudson",
        "email": "mike@example.com",
        "phone": "1234567890",
        "@type": "RelatedParty",
        "@referredType": "OrderLineItemContact"
      }
    ]
  }
]
The response body displays results of the created service order.

```json
{
  "externalId": "BSS748",
  "requestedStartDate": "2018-01-15T09:37:40.508Z",
  "requestedCompletionDate": "2018-01-15T09:37:40.508Z",
  "@type": "ServiceOrder",
  "note": [
    {
      "text": "This is a TMF service order illustration"
    },
    {
      "text": "This is a TMF service order illustration no 2"
    }
  ],
  "serviceOrderItem": [
    {
      "id": "100",
      "action": "add",
      "@type": "ServiceOrderItem",
      "relatedParty": [
        {
          "id": "eaf68911c35420105252716b7d40ddde",
          "name": "Sally Thomas",
          "@type": "RelatedParty",
          "@referredType": "CustomerContact"
        },
        {
          "id": "ffc68911c35420105252716b7d40dd55",
          "name": "Funco Intl",
          "@type": "RelatedParty",
          "@referredType": "Customer"
        }
      ]
    }
  ]
}
```
"service": {
    "@type": "CFS",
    "serviceCharacteristic": [
        {
            "name": "Firewall Security",
            "value": "Standard"
        },
        {
            "name": "Firewall coverage",
            "value": "Premium (Up to 50 sites)"
        },
        {
            "name": "Administration Support",
            "value": "Basic (Customer Managed)"
        }
    ],
    "serviceSpecification": {
        "id": "f99546ff07266010a7955b7e0ad300a8",
        "name": "Managed Firewall Service",
        "@type": "Managed Firewall Service"
    },
    "relatedParty": [
        {
            "firstName": "Mike",
            "lastName": "Hudson",
            "email": "mike@example.com",
            "phone": "1234567890",
            "@type": "RelatedParty",
            "@referredType": "OrderLineItemContact"
        }
    ],
    "place": {
        "id": "920cf6ac73d423002728660c4cf6a799",
        "@type": "Place"
    },
    "state": "new"
},
"relatedParty": [
    {
        "id": "eaf68911c35420105252716b7d40ddde",
        "name": "Sally Thomas",
        "@type": "RelatedParty",
        "@referredType": "OrderLineItemContact"
    }
]
Example: cURL request

The following example shows how to update a customer service order by updating properties and setting the serviceOrderItem.action property to change.

curl "https://instance.service-now.com/api/sn_ind_tmt_orm/serviceorder" \
    --request POST \
    --header "Accept:application/json" \
    --header "Content-Type:application/json" \
    --data "{
        "externalId": "BSS748",
        "requestedStartDate": "2018-01-15T09:37:40.508Z",
        "requestedCompletionDate": "2018-01-15T09:37:40.508Z",
        "@type": "ServiceOrder",
        "note": [
            { "text": "This is a TMF service order illustration" },
            { "text": "This is a TMF service order illustration no 2" }
        ],
        "serviceOrderItem": [
            { "id": "100", "action": "change", "@type": "ServiceOrderItem", "service": { "@type": "CFS", "serviceCharacteristic": [ ] } }
        ]
    }"
The response body displays results of the changed service order.

```json
{
    "externalId": "BSS748",
    "name": "Firewall Security",
    "value": "Premium",
    "previousValue": "Standard"
}
```
"requestedStartDate": "2018-01-15T09:37:40.508Z",
"requestedCompletionDate": "2018-01-15T09:37:40.508Z",
"@type": "ServiceOrder",
"note": [

  {
    "text": "This is a TMF service order illustration"
  },

  {
    "text": "This is a TMF service order illustration no 2"
  }
],
"serviceOrderItem": [

  {
    "id": "100",
    "action": "change",
    "@type": "ServiceOrderItem",
    "service": {
      "@type": "CFS",
      "serviceCharacteristic": [

        {
          "name": "Firewall Security",
          "value": "Premium",
          "previousValue": "Standard"
        }
      ],
      "id": "4b5072aec3a83010abc8b5183c40dd42"
    },
    "relatedParty": [

      {
        "firstName": "Mike",
        "lastName": "Hudson",
        "email": "mike@example.com",
        "phone": "1234567890",
        "@type": "RelatedParty",
        "@referredType": "OrderLineItemContact"
      }
    ],
    "place": {
      "id": "920cf6ac73d423002728660c4cf6a799",
      "@type": "Place"
    },
    "state": "new"
  }
]}
Example: cURL request

The following example shows how to remove a customer service order by setting the serviceOrderItem.action property to delete.

curl "https://instance.service-now.com/api/sn_ind_tmt_orm/serviceorder" \
--request POST \ 
--header "Accept:application/json" \ 
--header "Content-Type:application/json" \ 
--data "{
   "externalId": "BSS748",
   "requestedStartDate": "2018-01-15T09:37:40.508Z",
   "requestedCompletionDate": "2018-01-15T09:37:40.508Z",
   "@type": "ServiceOrder",
   "note": [ 
   {  
   "text": "This is a TMF service order illustration"
   },
   {  
   "text": "This is a TMF service order illustration no 2"
   }
   ],
   "serviceOrderItem": [ 
   { 
   "id": "100",
   "action": "delete",
   "@type": "ServiceOrderItem",
   "@type": "RelatedParty",
   "@referredType": "Customer
   }
   ,
   { 
   "id": "ff68911c35420105252716b7d40dd55",
   "name": "Funco Intl",
   "@type": "RelatedParty",
   "@referredType": "Customer
   }
   ]
   }"
"service": {  
  "@type": "CFS",  
  "serviceCharacteristic": [  
    {  
      "name": "Firewall Security",  
      "value": "Premium",  
      "previousValue": "Standard"  
    }  
  ],  
  "id": "4b5072aec3a83010abc8b5183c40dd42"  
},  
"relatedParty": [  
  {  
    "firstName": "Mike",  
    "lastName": "Hudson",  
    "email": "mike@example.com",  
    "phone": "1234567890",  
    "@type": "RelatedParty",  
    "@referredType": "OrderLineItemContact"  
  }  
],  
"place": {  
  "id": "920cf6ac73d423002728660c4cf6a799",  
  "@type": "Place"  
}  
],  
"relatedParty": [  
  {  
    "id": "eaf68911c35420105252716b7d40ddde",  
    "name": "Sally Thomas",  
    "@type": "RelatedParty",  
    "@referredType": "CustomerContact"  
  },  
  {  
    "id": "ffc68911c35420105252716b7d40dd55",  
    "name": "Funco Intl",  
    "@type": "RelatedParty",  
    "@referredType": "Customer"  
  }  
]}
--user 'username': 'password'
The response body displays information of the deleted service order.

```json
{
  "externalId": "BSS748",
  "requestedStartDate": "2018-01-15T09:37:40.508Z",
  "requestedCompletionDate": "2018-01-15T09:37:40.508Z",
  "@type": "ServiceOrder",
  "note": [
    {
      "text": "This is a TMF service order illustration"
    },
    {
      "text": "This is a TMF service order illustration no 2"
    }
  ],
  "serviceOrderItem": [
    {
      "id": "100",
      "action": "delete",
      "@type": "ServiceOrderItem",
      "service": {
        "@type": "CFS",
        "serviceCharacteristic": [
          {
            "name": "Firewall Security",
            "value": "Premium",
            "previousValue": "Standard"
          }
        ],
        "id": "4b5072aec3a83010abc8b5183c40dd42"
      },
      "relatedParty": [
        {
          "firstName": "Mike",
          "lastName": "Hudson",
          "email": "mike@example.com",
          "phone": "1234567890",
          "@type": "RelatedParty",
          "@referredType": "OrderLineItemContact"
        }
      ],
      "place": {
        "id": "920cf6ac73d423002728660c4cf6a799",
        "@type": "Place"
      }
    }
  ]
}
```
SG Services API

Creates and manages application services and upstream relationships between them.

Request apps on the Store

Visit the ServiceNow Store website to view all the available apps and for information about submitting requests to the store. For cumulative release notes information for all released apps, see the ServiceNow Store version history release notes.

This API can only be used when the CMDB Application API and CLI (sn_service_graph) plugin is activated. This API is used within the sn_service_graph namespace.

Using this API does not require details regarding source tables or relationship types.

To script critical operations which support automation across the enterprise, you can leverage APIs or run command line operations that the CMDB Application CLI and API store app provide instead of using the user interface. The CMDB Application CLI and API store app provides a robust framework
which consolidates all the APIs that are related to application services and the command lines that let you access the interface to those APIs.

CMDB Application CLI and API commands enable the following tasks:

• Registering and creating an application service and establishing upstream relationships
• Getting details of a given application service and its upstream relationships
• Connecting higher level constructs such as business applications and business service offerings
• Populating an application service with a given population type
• Changing the state of an application service

For the command line solution, see CMDB Application CLI and API available commands.

**SG Services – POST – `/sg_services/app_service/convert`**

Converts a manual or empty type application service to a calculated application service. During conversion, the application service record moves into the `[cmdb_ci_service_calculated]` table with the newly assigned class.

The following properties for identifying a CI take precedence as follows:

1. `sys_id` – If `sys_id`, the system only uses the `sys_id` and ignores any additional values.
2. `number` – If provided without the `sys_id`, the system only uses the number and ignores any additional values.
3. `<IRE field name>` – The system only uses these values if the `sys_id` or `number` are not provided.

**URL format**

Versioned URL: `/api/sn_service_graph/{api_version}/sg_services/app_service/convert`

Default URL: `/api/sn_service_graph/sg_services/app_service/convert`
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;IRE field name&gt;</td>
<td>One or more IRE fields identifying the application service. For example, name or version.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>levels</td>
<td>Number of levels to include in the conversion.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or the user does not have the app_service_admin role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>Indicates success or failure. Data type: String</td>
</tr>
</tbody>
</table>

### Example: cURL request

The following example shows how to convert an application service type.

```bash
curl
"https://instance.service-now.com/api/sn_service_graph/sg_services/app_service/convert"
--request POST
```
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--data "{
  "name": "Test Register",
  "environment": "Test",
  "version": "1.0",
  "levels": 8
}" \
--user 'username':'password'

Results indicating a successful conversion to a calculated application service.

```
{
  "result": {
    "status": "success"
  }
}
```

SG Services – POST – /sg_services/app_service/delete

Deletes an application service.

The following properties for identifying a CI take precedence as follows:

1. sys_id – If sys_id, the system only uses the sys_id and ignores any additional values.
2. number – If provided without the sys_id, the system only uses the number and ignores any additional values.
3. <IRE field name> – The system only uses these values if the sys_id or number are not provided.

URL format

Versioned URL: /api/sn_service_graph/{api_version}/sg_services/app_service/delete

Default URL: /api/sn_service_graph/sg_services/app_service/delete

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
</tbody>
</table>
Path parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;IRE field name&gt;</td>
<td>One or more IRE fields identifying the application service. For example, name or version. Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or the user does not have the app_service_admin role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>Indicates success or failure. Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example shows how to delete an application service.

curl "https://instance.service-now.com/api/sn_service_graph/sg_services/app_service/delete"
   --request POST
   --header "Accept:application/json"
   --header "Content-Type:application/json"
   --data "{
       "name": "Test Register",
       "environment": "Test",
       "version": "1.0"
   }"
   --user 'username': 'password'

Results indicating a successful removal of an application service.

{ "result": { |
SG Services – POST – /sg_services/app_service/find

Finds the details of a given application service and its upstream relationships.

Users with the app_service_user role can use this API, but results are restricted to application services in Operational status. The app_service_admin role provides unlimited viewing of application services.

The following properties for identifying a CI take precedence as follows:

1. sys_id – If sys_id, the system only uses the sys_id and ignores any additional values.
2. number – If provided without the sys_id, the system only uses the number and ignores any additional values.
3. <IRE field name> – The system only uses these values if the sys_id or number are not provided.

**URL format**

Versioned URL: /api/sn_service_graph/{api_version}/sg_services/app_service/find

Default URL: /api/sn_service_graph/sg_services/app_service/find

**Supported request parameters**

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;IRE fields&gt;</td>
<td>One or more IRE fields identifying the application service. For example, name or version.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or the user does not have the app_service_admin role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;IRE field name&gt;</code></td>
<td>One or more IRE fields identifying the application service. For example, name or version.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>operational_status</td>
<td>Operational status of the application service. For example, active.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>relationships</td>
<td>List of objects defining the application service's upstream relationships.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;relationships&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;number&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sys_id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;class_name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;relationship&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>relationships.class_name</td>
<td>Name of the class that contains the application service.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>relationships.name</td>
<td>Name of the relationship.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>relationships.number</td>
<td>Unique number of the relationship.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>relationships.relationship</td>
<td>The relationship rule.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>relationships.sys_id</td>
<td>Sys_id of the relationship.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

The following example shows how to find details of an application service.

```
curl "https://instance.service-now.com/api/sn_service_graph/sg_services/app_service/find" \
--request POST \ 
--header "Accept:application/json" \ 
--header "Content-Type:application/json" \ 
--data "{
   "name": "Test App Service1"
}" \ 
--user 'username':'password'
```

The response body includes the application service and relationship information.

```json
{
   "result": {
      "aliases": null,
      "asset": null,
      "asset_tag": null,
      "assigned": "",
      "assigned_to": null,
      "assignment_group": null,
      "attestation_score": null,
      "attested": "0",
      "attested_by": null,
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
SG Services – POST – /sg_services/app_service/populate

Populates an application service with a service population method.

The following properties for identifying a CI take precedence as follows:

- **Name**
- **System ID**
- **Number**
- **Class Name**
- **Relationship**
1. sys_id – If sys_id, the system only uses the sys_id and ignores any additional values.
2. number – If provided without the sys_id, the system only uses the number and ignores any additional values.
3. <IRE field name> – The system only uses these values if the sys_id or number are not provided.

**URL format**

Versioned URL: /api/sn_service_graph/{api_version}/sg_services/app_service/populate

Default URL: /api/sn_service_graph/sg_services/app_service/populate

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;IRE field name&gt;</td>
<td>One or more IRE fields identifying the application service. For example, name or version. Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service. Data type: String</td>
</tr>
</tbody>
</table>
## Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>population_method</td>
<td>Required. Identifies the population method and its accompanying property to identify the content for population. Only one accompanying object is valid per type. Data type: Object</td>
</tr>
<tr>
<td>population_method.group_id</td>
<td>Group ID of the CMDB group configured with the cmdb_group population type. Data type: &quot;String&quot;</td>
</tr>
<tr>
<td>population_method.levels</td>
<td>Number of levels to use in building the service. If the level value is not provided, the system checks the sys_property for the value. If svc.manual.convert.levels.default_value is not populated, a default value of 3 is used. Data type: Number</td>
</tr>
</tbody>
</table>

```
"population_method": {
  "group_id": "String",
  "type": "cmdb_group"
}
```

Associated population type: cmdb_group

```
"population_method": {
  "levels": Number,
  "type": "dynamic_service"
}
```

Associated population type: dynamic_service

Default: 3 if no level value is set for the sys_property
### Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>population_method.service_candidate</td>
<td>Unique identifier of the service candidate. Data type: String</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>population_method.service_relations</td>
<td>List of objects containing hierarchy data for the CIs within the application</td>
</tr>
<tr>
<td></td>
<td>service. All CIs form pairs with a parent and child CI. The top-level CI,</td>
</tr>
<tr>
<td></td>
<td>referred to as the entry point of an application service, does not have a</td>
</tr>
<tr>
<td></td>
<td>parent CI. Data type: Array</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>population_method.service_relations.child</td>
<td>Name of a child CI related to the CI. Data type: String</td>
</tr>
<tr>
<td>population_method.service_relations.parent</td>
<td>Name of a parent CI related to the CI. Data type: String</td>
</tr>
</tbody>
</table>
### Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>population_method.tags</td>
<td>List of objects containing tags to associate with the CI. This information is located in the Key Values [cmdb_key_value] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&quot;population_method&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;tags&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;tag&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>},</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: &quot;tag_list&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>Associated population type: tag_list</td>
</tr>
<tr>
<td>population_method.tags.tag</td>
<td>Tag name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>population_method.tags.value</td>
<td>Tag value.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>population_method.type</td>
<td>Required. Population type to add to the application service.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• cmdb_group</td>
</tr>
<tr>
<td></td>
<td>• service_hierarchy</td>
</tr>
<tr>
<td></td>
<td>• dynamic_service</td>
</tr>
<tr>
<td></td>
<td>• tag_list</td>
</tr>
<tr>
<td></td>
<td>• tag_based_service_family</td>
</tr>
</tbody>
</table>
Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or the user does not have the app_service_admin role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>
Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>Indicates success or failure.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

The following example shows how to populate an application service with a dynamic_service type.

```bash
curl
  "https://instance.service-now.com/api/sn_service_graph/sg_services/app_service/populate" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --data "{
    "name": "Test Register",
    "environment": "Test",
    "version": "1.0",
  }"

  \"population_method\": {  
    \"type\": \"dynamic_service\",
    \"levels\": 8
  }"

  --user 'username': 'password'
```

Results indicating a successful population of an application service.

```json
{
  "result": {
    "status": "success"
  }
}
```

**SG Services – POST – /sg_services/app_service/register**

Creates an application service, tags and constructs upstream relationships such as business applications, business service offerings, and other application services.

The following properties for identifying a CI take precedence as follows:
1. sys_id – If sys_id, the system only uses the sys_id and ignores any additional values.
2. number – If provided without the sys_id, the system only uses the number and ignores any additional values.
3. <IRE field name> – The system only uses these values if the sys_id or number are not provided.

**URL format**

Versioned URL: `/api/sn_service_graph/{api_version}/sg_services/app_service/register`

Default URL: `/api/sn_service_graph/sg_services/app_service/register`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;IRE field name&gt;</td>
<td>One or more IRE fields identifying the application service. For example, name or version. Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service. Data type: String</td>
</tr>
</tbody>
</table>
## Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relationships</td>
<td>Upstream relationships categorized by type. Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;relationships&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;business_app&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;business_service_offering&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;parent_app_service&quot;: [Array],</td>
</tr>
<tr>
<td></td>
<td>&quot;technical_service_offering&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Maximum number of relationships is 25.</td>
</tr>
<tr>
<td>relationships.business_app</td>
<td>List of objects representing Business Application relationship types. These</td>
</tr>
<tr>
<td></td>
<td>values can be defined using one of the following items as key-value pairs.</td>
</tr>
<tr>
<td></td>
<td>• &lt;IRE field name&gt;</td>
</tr>
<tr>
<td></td>
<td>• number</td>
</tr>
<tr>
<td></td>
<td>• sys_id</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>relationships.business_service_offering</td>
<td>List of objects representing Business Service Offering relationship types.</td>
</tr>
<tr>
<td></td>
<td>These values can be defined using the following items as key-value pairs.</td>
</tr>
<tr>
<td></td>
<td>• &lt;IRE field name&gt;</td>
</tr>
<tr>
<td></td>
<td>• number</td>
</tr>
<tr>
<td></td>
<td>• sys_id</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>relationships.parent_app_service</td>
<td>List of objects representing Application Service relationship types.</td>
</tr>
<tr>
<td></td>
<td>These values can be defined using the following items as key-value pairs.</td>
</tr>
<tr>
<td></td>
<td>• &lt;IRE field name&gt;</td>
</tr>
<tr>
<td></td>
<td>• number</td>
</tr>
<tr>
<td></td>
<td>• sys_id</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;IRE field name&gt;</code></td>
</tr>
<tr>
<td></td>
<td>• number</td>
</tr>
<tr>
<td></td>
<td>• sys_id</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>relationships.technical_service_offering</td>
<td>List of objects representing Technical Service Offering relationship types. These values can be defined using the following items as key-value pairs.</td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;IRE field name&gt;</code></td>
</tr>
<tr>
<td></td>
<td>• number</td>
</tr>
<tr>
<td></td>
<td>• sys_id</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>tags</td>
<td>List of objects containing tag definitions as key-value pairs.</td>
</tr>
<tr>
<td></td>
<td>&quot;tags&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;key&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>tags.key</td>
<td>Tag category name.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>tags.value</td>
<td>Tag value.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>cHeader</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
</tr>
</tbody>
</table>

Status codes
The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or the user does not have the app_service_admin role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>cHeader</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app_service</td>
<td>Application service details.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>cHeader</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>app_service</code>: {}</td>
<td><strong>name</strong>: String, <strong>number</strong>: String, <strong>sys_id</strong>: String</td>
</tr>
<tr>
<td>app_service.name</td>
<td>Name of the application service. Data type: String</td>
</tr>
<tr>
<td>app_service.number</td>
<td>Unique number that identifies the application service. Data type: String</td>
</tr>
<tr>
<td>app_service.sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table. Data type: String</td>
</tr>
<tr>
<td>message</td>
<td>Message describing the status. Possible values: • Service already exists • Service registered successfully Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>Status indicating whether the service has been registered. Possible values: • Insert: The application service was successfully created. • No action: The application service already exists. No action taken. Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

The following example shows how to register an application service.

```bash
curl "instance.service-now.com/api/sn_service_graph/sg_services/app_service/register"
  
  "--request POST 
  "--header "Accept:application/json" 
  "--header "Content-Type:application/json" 
  "--data "{
"name": "Test Register",
"environment": "Test",
"version": "1.0",
"number": " SNSVC0001014",
"relationships": {
  "business_application": [
    {
      "sys_id": "0250a94040697410f87713b656474250"
    },
    {
      "number": "APM0001002"
    },
    {
      "name": "Test Biz App1"
    }
  ],
  "business_service_offering": [
    {
      "sys_id": "ed32e98040697410f87713b656474259"
    }
  ],
  "technical_service_offering": [
    {
      "sys_id": "80e12d8040697410f87713b65647421c"
    },
    {
      "number": "BSN0001005"
    },
    {
      "name": "Tech Service Offering2"
    }
  ],
  "parent_app_service": [
    {
      "sys_id": "a2f0618040697410f87713b656474255"
    }
  ]
},
"tags": [
  {
    "key": "key1",
    "value": "value1"
  }
]
The response body includes ID and status information.

```json
{
  "result": {
    "app_service": {
      "sys_id": "5780cb604061f410f87713b656474271",
      "name": "Test Register",
      "number": " SNSVC0001014"
    },
    "message": "Service registered successfully",
    "status": "INSERT"
  }
}
```

**SG Services – POST – /sg_services/app_service/relationship/create**

Constructs upstream relations such as business applications, business service offerings, and other application services.

This API creates a relationship, taking input with a single parent and a corresponding child object.

The following properties for identifying a CI take precedence as follows:

1. sys_id – If sys_id, the system only uses the sys_id and ignores any additional values.
2. number – If provided without the sys_id, the system only uses the number and ignores any additional values.
3. <IRE field name> – The system only uses these values if the sys_id or number are not provided.

**URL format**

*Versioned URL:* /api/sn_service_graph/{api_version}/sg_services/app_service/relationship/create

*Default URL:* /api/sn_service_graph/sg_services/app_service/relationship/create
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>child</td>
<td>Information identifying the child application service with which to create a relationship. The child is located in the Application Service [cmdb_ci_service_auto] table. A dynamic CI group can be added as a child but cannot be parent. Data type: Object</td>
</tr>
<tr>
<td>child.&lt;service_app_identifier&gt;</td>
<td>Details identifying the child application service with which to create a relationship. Only one option is required. Each option is listed by processing precedence:</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
### Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• <code>sys_id</code> – Sys_id of the child application service.</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td>• <code>number</code> – Unique number that identifies the child application service.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;IRE field name&gt;</code> – IRE fields identifying the application service. For example, name or version.</td>
<td></td>
</tr>
<tr>
<td>parent</td>
<td>Details identifying the parent application service with which to create a relationship.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td>Only one option is required. Each option is listed by processing precedence:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• <code>sys_id</code> – Sys_id the application service listed in the Application Service [cmdb_ci_service_auto].</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• <code>number</code> – Unique number that identifies the application service.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• <code>&lt;IRE field name&gt;</code> – One or more IRE fields identifying the application service. For example, name or version.</td>
<td></td>
</tr>
<tr>
<td>parent.&lt;service_app_identifier&gt;</td>
<td>Information identifying the application service.</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td>The parent class name should be from one of the following tables:</td>
<td></td>
</tr>
<tr>
<td>parent.class_name</td>
<td>Name of the class that contains the application service.</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• cmdb_ci_service_auto</td>
<td></td>
</tr>
<tr>
<td>• cmdb_ci_service_discovered</td>
<td></td>
</tr>
<tr>
<td>• cmdb_ci_service_by_tags</td>
<td></td>
</tr>
<tr>
<td>• cmdb_ci_service_calculated</td>
<td></td>
</tr>
<tr>
<td>• service_offering</td>
<td></td>
</tr>
<tr>
<td>• cmdb_ci_business_app</td>
<td></td>
</tr>
<tr>
<td>Default: cmdb_ci_service_auto</td>
<td></td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or the user does not have the app_service_admin role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| status | Indicates success or failure.  
Data type: String |

Example: cURL request

The following example shows how create a relationship from an application service.

```c
curl  
"https://instance.service-now.com/api/sn_service_graph/sg_services/app_service/relationship/create"  
--request POST  
--header "Accept:application/json"  
--header "Content-Type:application/json"  
--data 
"{  
"child": {  
"name": "wdfsdf",  
"environment": "Test",  
"version": "1.0"  
},  
"parent": {  
"sys_id": "abcdefg",  
"name": "business App1",  
"class_name": "service_offering" 
} 
}"  
--user 'username':'password'
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Results indicating a successful creation of an application service relationship.

```
{
  "result": {
    "status": "success"
  }
}
```

**SG Services – POST – /sg_services/app_service/relationship/delete**

Deletes an application service upstream relationship.

The following properties for identifying a CI take precedence as follows:

1. `sys_id` – If `sys_id`, the system only uses the `sys_id` and ignores any additional values.
2. `number` – If provided without the `sys_id`, the system only uses the number and ignores any additional values.
3. `<IRE field name>` – The system only uses these values if the `sys_id` or number are not provided.

**URL format**

*Versioned URL:* `/api/sn_service_graph/{api_version}/sg_services/app_service/relationship/delete`

*Default URL:* `/api/sn_service_graph/sg_services/app_service/relationship/delete`

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

| Name   | Description |
|--------|-------------|-------------|
| None   |             |             |
### Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| child                          | Information describing the child relationship to be deleted from the service application.  
|                                | Data type: Object                                                                                                                          |
|                                | "child": {  
|                                |   "<IRE field name>": "String",  
|                                |   "number": "String",  
|                                |   "sys_id": "String"  
|                                | }                                                                                                                                 |
| child.<IRE field name>         | One or more IRE fields identifying the child application service. For example, name or version.                                           |
|                                | Data type: String                                                                                                                          |
| child.number                   | Unique number that identifies the child application service.                                                                               |
|                                | Data type: String                                                                                                                          |
| child.sys_id                   | Sys_id of the child application service listed in the Application Service [cmdb_ci_service_auto].                                           |
|                                | Data type: String                                                                                                                          |
| parent                         | Details identifying the parent application service from which to remove a relationship.                                                    |
|                                | Data type: Object                                                                                                                          |
|                                | "parent": {  
|                                |   "<IRE field name>": "String",  
|                                |   "number": "String",  
|                                |   "sys_id": "String",  
|                                |   "class_name": "String"  
|                                | }                                                                                                                                 |
| parent.<IRE field name>        | One or more IRE fields identifying the application service. For example, name or version.                                                   |
|                                | Data type: String                                                                                                                          |
| parent.number                  | Unique number that identifies the application service.                                                                                    |
|                                | Data type: String                                                                                                                          |
### Request body parameters (JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent.sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table. Data type: String</td>
</tr>
</tbody>
</table>
| parent.class_name| Name of the class that contains the application service. The parent class name should be from one of the following tables:  
  • cmdb_ci_service_auto  
  • cmdb_ci_service_discovered  
  • cmdb_ci_service_by_tags  
  • cmdb_ci_service_calculated  
  • service_offering  
  • cmdb_ci_business_app  
  Default: cmdb_ci_service_auto  
  Data type: String |

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or the user does not have the app_service_admin role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>Indicates success or failure.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

Example: cURL request

The following example shows how to remove a relationship from an application service.

```
curl
  "https://instance.service-now.com/api/sn_service_graph/sg_services/app_service/relationship/delete" 
--request POST 
--header "Accept:application/json" 
--header "Content-Type:application/json" 
--data "{
    "child": {
      "name": "Test Register",
      "environment": "Test",
      "version": "1.0"
    },

    "parent": {
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Results indicating a successful removal of an application service relationship.

```json
{
   "result": {
      "status": "success"
   }
}
```

**SG Services – POST – /sg_services/app_service/state**

Changes the application service lifecycle state to activate, deactivate, or retire.

The following properties for identifying a CI take precedence as follows:

1. `sys_id` – If `sys_id`, the system only uses the `sys_id` and ignores any additional values.
2. `number` – If provided without the `sys_id`, the system only uses the `number` and ignores any additional values.
3. `<IRE field name>` – The system only uses these values if the `sys_id` or `number` are not provided.

**URL format**

Versioned URL: `/api/sn_service_graph/{api_version}/sg_services/app_service/state`

Default URL: `/api/sn_service_graph/sg_services/app_service/state`

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>api_version</td>
</tr>
</tbody>
</table>
## Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

## Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;IRE field name&gt;</code></td>
<td>One or more IRE fields identifying the application service. For example, name or version. Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service. Data type: String</td>
</tr>
<tr>
<td>state</td>
<td>Required. Lifecycle state of the application service. These values are updated in the Application Services [cmdb_ci_service_auto] table. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• ACTIVATE – Life cycle is operational and in use.</td>
</tr>
<tr>
<td></td>
<td>◦ operational_status=Operational</td>
</tr>
<tr>
<td></td>
<td>◦ life_cycle_stage=Operational</td>
</tr>
<tr>
<td></td>
<td>◦ life_cycle_stage_status=In Use</td>
</tr>
<tr>
<td></td>
<td>• DEACTIVATE – Life cycle is not operational and is in the design stage.</td>
</tr>
<tr>
<td></td>
<td>◦ operational_status=Non-Operational</td>
</tr>
<tr>
<td></td>
<td>◦ life_cycle_stage=Design</td>
</tr>
<tr>
<td></td>
<td>◦ life_cycle_stage_status=Build</td>
</tr>
<tr>
<td></td>
<td>• RETIRE – End of life.</td>
</tr>
<tr>
<td></td>
<td>◦ operational_status=Retired</td>
</tr>
<tr>
<td></td>
<td>◦ life_cycle_stage=End Of Life</td>
</tr>
<tr>
<td></td>
<td>◦ life_cycle_stage_status=Retired</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table. Data type: String</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Headers
The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes
The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or the user does not have the app_service_admin role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>Indicates success or failure.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
Example: cURL request

The following example shows how to change an application service lifecycle state to activate.

curl "https://instance.service-now.com/api/sn_service_graph/sg_services/app_service/state"
  \   
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --data "{
   \"name\": \"Test Register\",\n   \"environment\": \"Test\",\n   \"version\": \"1.0\",\n   \"state\": \"activate\"
   }
} \
--user 'username':'password'

Results indicating a successful operation.

{
  "result": {
    "status": "success"
  }
}

SG Services – POST – /sg_services/app_service/update

Updates an existing application service provided and creates tags for the given application service.

The following properties for identifying a CI take precedence as follows:

1. sys_id – If sys_id, the system only uses the sys_id and ignores any additional values.
2. number – If provided without the sys_id, the system only uses the number and ignores any additional values.
3. <IRE field name> – The system only uses these values if the sys_id or number are not provided.

URL format

Versioned URL: /api/sn_service_graph/{api_version}/sg_services/app_service/update
Default URL: /api/sn_service_graph/sq_services/app_service/update

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

Request body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;fields or tags to update&gt;</td>
<td>Use key-value pairs to identify each field or tag to be updated. Only basic information can be updated, no upstream relationships can be updated. Data type: String</td>
</tr>
<tr>
<td>&lt;IRE field name&gt;</td>
<td>One or more IRE fields identifying the application service. For example, name or version. You can send the sys_id, number or IRE to identify an application service; however, none of these fields can be updated when used as an identifier. To update the IRE fields, input should include the sys_id or number as an identifier. Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or the user does not have the app_service_admin role.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;IRE field name&gt;</td>
<td>One or more IRE fields identifying the application service. For example, name or version. Data type: String</td>
</tr>
<tr>
<td>number</td>
<td>Unique number that identifies the application service. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the application service listed in the Application Service [cmdb_ci_service_auto] table.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&lt;updated fields&gt;</td>
<td>If the update is successful, each modified field sent in the payload is listed in the response body.</td>
</tr>
</tbody>
</table>

**Example: cURL request**

The following example shows how to update an application service, using name as an IRE field.

```bash
curl "https://instance.service-now.com/api/sn_service_graph/sg_services/app_service/update"
  
--request POST 
--header "Accept:application/json"
--header "Content-Type:application/json"
--data "{
   
   
   "name": "Test Register",
   "environment": "Test",
   "version": "1.0"
   
   }
   
--user 'username':'password'
```

Response includes application service identification information and updated fields.

```json
{  
"result": {  
  "sys_id": "123456",
  "number": "SVCKji0w9e",
  "name": "Test Register",
  "environment": "Test",
  "version": "1.0"
}
}
```

**Syntax Editor API**

The Syntax Editor API enables you to access context menu options in the JavaScript syntax editor. Use this API to retrieve type definitions for script includes, find references to script includes or tables, and retrieve a list of cached tokens.

For more information about the syntax editor, see JavaScript syntax editor.
This API is available by default.

**Syntax Editor - GET /syntax_editor/cache/{token_type}**
Retrieves a list of cached tokens by token type.

**URL format**

Versioned URL: /api/now/{api_version}/syntax_editor/cache/{token_type}
Default URL: /api/now/syntax_editor/cache/{token_type}

**Supported request parameters**

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>api_version</td>
</tr>
</tbody>
</table>
| token_type | Type of token. Valid values:  
- sys_db_object. The table containing a record of each table in the database.  
- sys_script_include. The table containing script includes. Data type: String |

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>name</td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Result of the request. An object containing the flush time for the cache, and a list of cached tokens based on the requested token type. The token type can be either sys_script_include or sys_db_object.</td>
</tr>
</tbody>
</table>

```json
"result": {
  "flushTime": Number,
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;token_type&gt;: {Object}</td>
<td></td>
</tr>
<tr>
<td>Data type: Object</td>
<td></td>
</tr>
</tbody>
</table>
| result.flushTime | Time since the cache was last flushed.  
Data type: Number  
Unit: milliseconds  |
| result.sys_script_include | List of all the cached script includes. Each list item has the name of the script include and its sys_id.  
"sys_script_include": {  
  <name>: <sys_id>  
}  
Data type: Object | |
| result.sys_script_include.<name> | Name of the script include and its sys_id.  
Data type: String | |
| result.sys_db_object | List of all the cached database objects. Each list item has the name of the database object and a Boolean indicating whether it's cached.  
"sys_db_object": {  
  <name>: Boolean  
}  
Data type: Object | |
| result.sys_db_object.<name> | Name of the database object with a flag indicating whether it's cached.  
Possible values:  
• true. The object is cached.  
• false. The object is not cached.  
Data type: Boolean | |

**Example: cURL request**

This example retrieves a list of cached tokens with a sys_script_include token type.
curl "https://instance.servicenow.com/api/now/syntax_editor/cache/sys_script_include" \
--request GET \n--header "Accept:application/json" \n--user 'username':'password'

{
    "result": {
        "flushTime":1611616020213,
        "sys_script_include": {
            "sn_app.TableFinder":"fa039df3c3330200c409914523aa5133",
            "global.TestRunner":"fb4aa3200000302207e7829cac2dc3488",
            "global.IndexCreator":"017681d20a2581030070eb488bee3fb8",
            "sn_srt.NewFields":"67f60c51cb103355439b78d5634c9c74"
        }
    }
}

Syntax Editor - POST /syntax_editor/getReferences

Returns a list of the files that reference a specific script include or table.

URL format

Versioned URL: /api/now/{api_version}/syntax_editor/getReferences

Default URL: /api/now/syntax_editor/getReferences

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ignoredTables</td>
<td>Tables to ignore when searching for references to a specific script_include or table. Use either ignoredTables or searchTables, not both.</td>
</tr>
<tr>
<td></td>
<td>&quot;ignoredTables&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>searchTables</td>
<td>Tables to include when searching for references to a specific script include or table. Use either ignoredTables or searchTables, not both.</td>
</tr>
<tr>
<td></td>
<td>&quot;searchTables&quot;: [Array]</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>searchWord</td>
<td>Required. Name of the script include or table to find references to.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>searchWordType</td>
<td>Required. Token type of the searchWord. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• script_include</td>
</tr>
<tr>
<td></td>
<td>• table</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Request headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Result of the request, presented as an array of objects. Each object contains all the references found in a single file.</td>
</tr>
</tbody>
</table>

```
"result": [ 
    {
        "name": "String",
        "sysId": "String",
        "tableName": "String",
        "tableDisplayName": "String",
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result.name</td>
<td>Name of the file containing the references.</td>
</tr>
<tr>
<td>result.sysId</td>
<td>Sys_id of the file containing the references.</td>
</tr>
<tr>
<td>result.tableName</td>
<td>Name of the table containing the file that has the references.</td>
</tr>
<tr>
<td>result.tableDisplayName</td>
<td>Label of the table containing the file that has the references.</td>
</tr>
<tr>
<td>result.applicationLabel</td>
<td>Application containing the file that has the references.</td>
</tr>
<tr>
<td>result.fields</td>
<td>Fields containing the references. An array of objects where each object has a field name and an array of references found in that field.</td>
</tr>
<tr>
<td>result.fields.name</td>
<td>Name of the field that contains the references.</td>
</tr>
<tr>
<td>result.fields.references</td>
<td>List of references, presented as an array of objects. Each object is one reference, with a line</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>number specifying its location in the file and a code snippet that shows the reference.</td>
</tr>
<tr>
<td></td>
<td>&quot;references&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;lineNo&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;code&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td>result.fields.references.lineNo</td>
<td>Line number of the reference.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>result.fields.references.code</td>
<td>Code snippet that contains the reference.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

This example searches the sys_script_include table for all references to the ListFinder script include.

```bash
curl "https://instance.servicenow.com/api/now/syntax_editor/getReferences" \
   --request POST \
   --header "Accept:application/json" \
   --header "Content-Type:application/json" \
   --data "{
   "searchWord":"ListFinder",
   "searchWordType":"script_include",
   "searchTables":["sys_script_include"]
}" \
   --user 'username':'password
```

```json
{
   "result":{
      {
         "name":"SysList",
         "sysId":"7d8f32c3c0a8016400e609be97b11s98",
         "tableName": "sys_script_include",
         "tableDisplayName": "Script Include",
         "applicationLabel": "Global",
         "fields":{
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Syntax Editor - POST /syntax_editor/intellisense/{tableName}

Returns the type definition for a script include.

**URL format**

Versioned URL: /api/now/{api_version}/syntax_editor/intellisense/{tableName}

Default URL: /api/now/syntax_editor/intellisense/{tableName}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>tableName</td>
<td>Name of the table that contains the script includes, such as sys_script_include. Data type: String</td>
</tr>
</tbody>
</table>
### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scriptIncludes</td>
<td>Required. List of sys_ids for the script includes. For example, a list of sys_ids from the Script Includes [sys_script_include] table. Data type: Array</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#)

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

#### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#)
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>result</td>
<td>Key/value pairs for each script include. The key is the sys_id of the script include. The value is the type definition. Both the key and value are strings.</td>
</tr>
<tr>
<td>result.&lt;sys_id&gt;</td>
<td>The script include, identified by its sys_id, followed by its type definition.</td>
</tr>
</tbody>
</table>

Data type: String

Example: cURL request

This example retrieves the type definition for two script includes.

```bash
curl "https://instance.servicenow.com/api/now/syntax_editor/intellisense/sys_script_include" \
   --request POST \
   --header "Accept:application/json" \
   --header "Content-Type:application/json" \
   --data "{
   "scriptIncludes": [
    "8e7bfa6c0a8002700f0e559b09de474",
    "b1ed0f5c3b23300daa79624a1d3ae0b"
   ]
}
   --user 'username':'password'
```

```json
| "result": { |
```
Table API

The Table API allows you to perform create, read, update, and delete (CRUD) operations on existing tables.

**Table - DELETE /now/table/{tableName}/{sys_id}**

Deletes the specified record from the specified table.

**URL format**

Versioned URL: /api/now/{api_version}/table/{tableName}/{sys_id}

Default URL: /api/now/table/{tableName}/{sys_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the record to delete. Data type: String</td>
</tr>
<tr>
<td>tableName</td>
<td>Name of the table from which to delete the specified record, such as &quot;incident&quot; or &quot;asset&quot;. Data type: String</td>
</tr>
</tbody>
</table>
### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| sysparm_query_no_domain     | Flag that indicates whether to restrict the record search to only the domains for which the logged in user is configured. Valid values:  
  - false: Exclude the record if it is in a domain that the currently logged in user is not configured to access.  
  - true: Include the record even if it is in a domain that the currently logged in user is not configured to access.  
  Data type: Boolean  
  Default: false |

**Note:** The `sysparm_query_no_domain` parameter is available only to system administrators or users who have the `query_no_domain_table_api` role.

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
</table>
| Accept | Data format of the response body. Supported types: `application/json`  
  Of `application/xml`.  
  Default: `application/json` |
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Indicates that the request completed successfully.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Example: cURL request

Delete a record from the Incident table.

```bash
curl "https://instance.servicenow.com/api/now/table/incident/d977b66a4f411200adf9f8e18110c7b2" \
--request DELETE \
--header "Accept:application/json" \
--user 'username':'password'
```

There is no response body.

None

Example: Python request

Delete a record from the Incident table.

```python
# Need to install requests package for python
import requests

# Set the request parameters
```
url =
'https://instance.servicenow.com/api/now/table/incident/d977b66a4f411200adf9f8e18110c7b2'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept':"application/json"}

# Do the HTTP request
response = requests.delete(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 204
if response.status_code != 204:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',
    response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

There is no response body.

None

Table - GET /now/table/{tableName}

Retrieves multiple records for the specified table.

For basic instructions, see Retrieve existing incidents.

URL format

Versioned URL: /api/now/{api_version}/table/{tableName}
Default URL: /api/now/table/{tableName}
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| api_version    | Optional. Version of the endpoint to access. For example, `v1` or `v2`. Only specify this value to use an endpoint version other than the latest. Depending on the version, this endpoint returns different results on a valid query.  
  • Version 1 returns error code 404 with no results.  
  • Version 2 returns success code 200 and an empty array as the response body.                                                                 |
|                | Data type: String                                                                                                                                                                                            |
| tableName      | Name of the table from which to retrieve the records.                                                                                                                                                        |
|                | Data type: String                                                                                                                                                                                            |

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| name-value pairs | Name-value pairs to use to filter the result set. This parameter is mutually exclusive with `sysparm_query`. For example, instead of using `&sysparm_query=active=true`, you can simplify the calling statement by using `&active=true`.  
  Based on this value, retrieves the display value and/or the actual value from the database.  
  • true: Returns the display values for all fields.  
  • false: Returns the actual values from the database.  
  • all: Returns both actual and display values.                                                                                             |
|                | Data type: String                                                                                                                                                                                            |
| sysparm_display_value | Data retrieval operation for reference and choice fields. Based on this value, retrieves the display value and/or the actual value from the database.                                                                 |
|                | Data type: String                                                                                                                                                                                            |
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| sysparm_exclude_reference_link | Flag that indicates whether to exclude Table API links for reference fields.  
Valid values:  
- true: Exclude Table API links for reference fields.  
- false: Include Table API links for reference fields.  
Data type: Boolean  
Default: false |
| sysparm_fields              | Comma-separated list of fields to return in the response.  
Data type: String |
| sysparm_limit               | Maximum number of records to return. For requests that exceed this number of records, use the **sysparm_offset** parameter to paginate record retrieval.  
This limit is applied before ACL evaluation. If no records you have access to return, rearrange the record order so records you have access to return first.  
⚠️ **Note:** Unusually large **sysparm_limit** values can impact system performance.  
Data type: Number  
Default: 1000 |
| sysparm_no_count            | Flag that indicates whether to execute a `select count(*)` query on the table to return the number of rows in the associated table.  
Valid values:  
- true: Do not execute a `select count(*)`.  
- false: Execute a `select count(*)`.  
Data type: Boolean  
Default: false |

Note: There is no preferred method for setting the `sysparm_no_count` parameter. Specifying the display value may cause performance issues since it is not reading directly from the database and may include referencing other fields and records. For more information on display values and actual values, see Table API FAQs (KB0534905).
## Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sysparm_offset</code></td>
<td>Starting record index for which to begin retrieving records. This functionality enables you to paginate record retrieval. This functionality enables the retrieval of all records, regardless of the number of records, in small manageable chunks. For example, the first time you call this endpoint, <code>sysparm_offset</code> is set to &quot;0&quot;. To simply page through all available records, use <code>sysparm_offset=sysparm_offset+sysparm_limit</code>, until you reach the end of all records. Do not pass a negative number in the <code>sysparm_offset</code>. Data type: Number Default: 0</td>
</tr>
</tbody>
</table>
| `sysparm_query`       | Encoded query used to filter the result set. Syntax: `sysparm_query=<col_name><operator><value>`.  
  • `<col_name>`: Name of the table column to filter against.  
  • `<operator>`: Supports the following values:  
    - `=`: Exactly matches `<value>`.  
    - `!`: Does not match `<value>`.  
    - `^`: Logically AND multiple query statements.  
    - `^OR`: Logically OR multiple query statements.  
    - `LIKE`: `<col_name>` contains the specified string. Only works for `<col_name>` fields whose data type is string.  
    - `STARTSWITH`: `<col_name>` starts with the specified string. Only works for `<col_name>` fields whose data type is string.  
    - `ENDSWITH`: `<col_name>` ends with the specified string. Only works for `<col_name>` fields whose data type is string.  
  `<value>`: Value to match against. All parameters are case-sensitive. Queries can contain more than one entry, such as `sysparm_query=<col_name><operator><value>[<operator><col_name><operator><value>]`. For example:  
  ```javascript
  (sysparm_query=caller_id=javascript:gs.getUserID()^active=true)
  ```  
  Encoded queries also supports order by functionality. To sort responses based on certain fields, use the `ORDERBY` and `ORDERBYDESC` clauses. |
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Syntax:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>For example: sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory</td>
</tr>
<tr>
<td></td>
<td>This query filters all active records and orders the results in ascending order by number, and then in descending order by category.</td>
</tr>
<tr>
<td></td>
<td>If part of the query is invalid, such as by specifying an invalid field name, the instance ignores the invalid part. It then returns rows using only the valid portion of the query. You can control this behavior using the glide.invalid_query.returns_no_rows property. Set this property to true to return no rows on an invalid query.</td>
</tr>
<tr>
<td></td>
<td>Note: The glide.invalid_query.returns_no_rows property controls the behavior of all queries across the instance, GlideRecord.query(), and web service APIs.</td>
</tr>
</tbody>
</table>

Data type: String

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_query_category</td>
<td>Name of the category to use for queries.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_query_no_domain</td>
<td>Flag that indicates whether to restrict the record search to only the domains for which the logged in user is configured. Valid values: false: Exclude the record if it is in a domain that the user is not configured to access. true: Include the record even if it is in a domain that the user is not configured to access.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td></td>
</tr>
<tr>
<td>Default: false</td>
<td></td>
</tr>
</tbody>
</table>

Note: The sysparm_query_no_domain parameter is available only to system administrators or users who have the query_no_domain_table_api role.
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| sysparm_suppress_pagination_header          | Flag that indicates whether to remove the Link header from the response. The Link header provides various URLs to relative pages in the record set which you can use to paginate the returned record set.  
Valid values:  
• true: Remove the Link header from the response.  
• false: Do not remove the Link header from the response.  
Data type: Boolean  
Default: false |
| sysparm_view                                 | UI view for which to render the data. Determines the fields returned in the response.  
Valid values:  
• desktop  
• mobile  
• both  
If you also specify the `sysparm_fields` parameter, it takes precedence.  
Data type: String |

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

### Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link</td>
<td>Relative URLs, based on the previous request, that you can use to page through the available record set. For example:</td>
</tr>
<tr>
<td></td>
<td>https://&lt;instance name&gt;.servicenow.com/api/now/table/cmdb_ci?sysparm_offset=40&amp;sysparm_limit=10000&gt;;rel=&quot;next&quot;,</td>
</tr>
<tr>
<td></td>
<td>https://&lt;instance name&gt;.servicenow.com/api/now/table/cmdb_ci?sysparm_offset=40&amp;sysparm_limit=10000&gt;;rel=&quot;prev&quot;,</td>
</tr>
<tr>
<td></td>
<td>https://&lt;instance name&gt;.servicenow.com/api/now/table/cmdb_ci?sysparm_offset=0&amp;sysparm_limit=10000&gt;;rel=&quot;first&quot;,</td>
</tr>
<tr>
<td></td>
<td>https://&lt;instance name&gt;.servicenow.com/api/now/table/cmdb_ci?sysparm_offset=2780&amp;sysparm_limit=10000&gt;;rel=&quot;last&quot;</td>
</tr>
<tr>
<td></td>
<td>For additional information on the rel parameter, refer to <a href="https://html.spec.whatwg.org/multipage/links.html#linkTypes">https://html.spec.whatwg.org/multipage/links.html#linkTypes</a>.</td>
</tr>
<tr>
<td>X-Total-Count</td>
<td>Total count of records returned by the query.</td>
</tr>
</tbody>
</table>

### Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](https://service-now.com).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Request completed successfully. If a valid query returned no results, the response body contains only an empty result array.</td>
</tr>
</tbody>
</table>
Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Field names and values of all parameters within the specified record or those specified in the query parameters.</td>
</tr>
</tbody>
</table>

Example: cURL request

Retrieve the first record from the Problem table.

```
curl "https://instance.servicenow.com/api/now/table/problem?sysparm_limit=1" \
--request GET \
--header "Accept:application/json" \
--user 'username':'password'
```

The response contains the name-value pairs for the requested record.

```
{
    "result": [ 
        {
            "parent": "",
            "made_sla": "true",
            "watch_list": "",
            "upon_reject": "cancel",
            "sys_updated_on": "2016-01-19 04:52:04",
            "approval_history": "",
            "number": "PRB0000050",
            "sys_updated_by": "glide.maint",
            "opened_by": { 
                "link": "https://instance.servicenow.com/api/now/table/sys_user/glide.maint",
                "value": "glide.maint"
            },
            "user_input": "",
            "sys_created_on": "2016-01-19 04:51:19",
            "sys_domain": { 
                "link": "https://instance.servicenow.com/api/now/table/sys_user_group/global",
                "value": "global"
            },
            "state": "4",
            "sys_created_by": "glide.maint",
            "knowledge": "false",
            "order": "",
            "closed_at": "2016-01-19 04:52:04",
        }
    ]
}
```
"cmdb_ci": {
    "link": "https://instance.servicenow.com/api/now/table/cmdb_ci/55b35562c0a8010e01cff22378e0aea9",
    "value": "55b35562c0a8010e01cff22378e0aea9"
},
"delivery_plan": "",
"impact": "3",
"active": "false",
"work_notes_list": "",
"business_service": "",
"priority": "4",
"sys_domain_path": "/",
"time_worked": "",
"expected_start": "",
"rejection_goto": "",
"opened_at": "2016-01-19 04:49:47",
"business_duration": "1970-01-01 00:00:00",
"group_list": "",
"work_end": "",
"approval_set": "",
"wf_activity": "",
"work_notes": "",
"short_description": "Switch occasionally drops connections",
"correlation_display": "",
"delivery_task": "",
"work_start": "",
"assignment_group": "",
"additional_assignee_list": "",
"description": "Switch occasionally drops connections",
"calendar_duration": "1970-01-01 00:02:17",
"close_notes": "updated firmware",
"sys_class_name": "problem",
"closed_by": "",
"follow_up": "",
"sys_id": "04ce72c9c0a8016600b5b7f75ac67b5b",
"contact_type": "phone",
"urgency": "3",
"company": "",
"reassignment_count": "",
"activity_due": "",
"assigned_to": "",
"comments": "",
"approval": "not requested",
"sla_due": ""}
Example: Python request
Retrieve the first record from the Problem table.

```python
# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/table/problem?sysparm_limit=1'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept':"application/xml"}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

The response contains the name-value pairs for the requested record.
<priority>1</priority>
<sys_domain_path/>
<rfc/>
<time_worked/>
<expected_start/>
<rejection_goto/>
<opened_at>2015-10-19 23:09:51</opened_at>
<group_list/>
<work_end/>
<caller_id/>
<resolved_at>2016-01-19 19:56:12</resolved_at>
<approval_set/>
<subcategory/>
<wf_activity/>
<work_notes/>
<short_description>Can't read email</short_description>
<close_code>Closed/Resolved by Caller</close_code>
<correlation_display/>
<delivery_task/>
<work_start/>
<assignment_group/>
<additional_assignee_list/>
<business_stc>1892781</business_stc>
<description>User can't access email on mail.company.com.</description>
<calendar_duration>1970-04-02 20:46:21</calendar_duration>
<close_notes>Closed before close notes were made mandatory</close_notes>
<notify>1</notify>
<sys_class_name>incident</sys_class_name>
<closed_by>
Table - GET /now/table/{tableName}/{sys_id}

Retrieves the record identified by the specified sys_id from the specified table.

**URL format**

Versioned URL: /api/now/{api_version}/table/{tableName}/{sys_id}

Default URL: /api/now/table/{tableName}/{sys_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the record to retrieve.</td>
</tr>
<tr>
<td>tableName</td>
<td>Name of the table from which to retrieve the record.</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_display_value</td>
<td>Data retrieval operation for reference and choice fields. Based on this value, retrieves the display value and/or the actual value from the database.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Valid values: ____________________________________________________________________________</td>
</tr>
</tbody>
</table>
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_exclude_reference_link</td>
<td>Flag that indicates whether to exclude Table API links for reference fields.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Exclude Table API links for reference fields.</td>
</tr>
<tr>
<td></td>
<td>• false: Include Table API links for reference fields.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>sysparm_fields</td>
<td>Comma-separated list of fields to return in the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sysparm_query_no_domain</td>
<td>Flag that indicates whether to restrict the record search to only the</td>
</tr>
<tr>
<td></td>
<td>domains for which the logged in user is configured.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
</tbody>
</table>

Note: There is no preferred method for setting this parameter. However, specifying the display value may cause performance issues since it is not reading directly from the database and may include referencing other fields and records. For more information on display values and actual values, see Table API FAQs (KB0534905).
**Query parameters (continued)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>sysparm_query_no_domain</strong></td>
<td>• false: Exclude the record if it is in a domain that the currently logged in user is not configured to access.</td>
</tr>
<tr>
<td></td>
<td>• true: Include the record even if it is in a domain that the currently logged in user is not configured to access.</td>
</tr>
<tr>
<td>Data type: Boolean</td>
<td>Default: false</td>
</tr>
<tr>
<td><strong>Note:</strong> The <em>sysparm_query_no_domain</em> parameter is available only to system administrators or users who have the <em>query_no_domain_table_api</em> role.</td>
<td></td>
</tr>
<tr>
<td>sysparm_view</td>
<td>UI view for which to render the data. Determines the fields returned in the response.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• desktop</td>
</tr>
<tr>
<td></td>
<td>• mobile</td>
</tr>
<tr>
<td></td>
<td>• both</td>
</tr>
<tr>
<td></td>
<td>If you also specify the <em>sysparm_fields</em> parameter, it takes precedence.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
</tbody>
</table>

**Request body parameters (XML or JSON)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Field names and values of all parameters within the specified record or those specified in the query parameters.</td>
</tr>
</tbody>
</table>

Example: cURL request

Retrieve a record from the Incident table.

curl
  "https://instance.servicenow.com/api/now/table/incident/a9e30c7dc61122760116894de7bcc7bd"
  --request GET
  --header "Accept:application/json"
  --user 'username':'password"
The response contains the name-value pairs for the requested record.

```json
{
    "result": {
        "upon_approval": "",
        "location": {
            "link": "https://instance.servicenow.com/api/now/table/cmn_location/105cf7f3c611227501e75e08b14a38ba",
            "value": "105cf7f3c611227501e75e08b14a38ba"
        },
        "expected_start": "",
        "reopen_count": "",
        "close_notes": "",
        "additional_assignee_list": "",
        "impact": "1",
        "urgency": "3",
        "correlation_id": "",
        "sys_tags": "",
        "sys_domain": {
            "link": "https://instance.servicenow.com/api/now/table/sys_user_group/global",
            "value": "global"
        },
        "description": "",
        "group_list": "",
        "priority": "3",
        "delivery_plan": "",
        "sys_mod_count": "4",
        "work_notes_list": "",
        "business_service": "",
        "follow_up": "",
        "closed_at": "",
        "sla_due": "2015-11-11 22:04:15",
        "delivery_task": "",
        "sys_updated_on": "2015-11-01 22:37:27",
        "parent": "",
        "work_end": "",
        "number": "INC0000046",
        "closed_by": "",
        "work_start": "",
        "calendar_stc": "",
        "category": "software",
        "business_duration": "",
        "incident_state": "1"
    }
}
```
"activity_due": "",
"correlation_display": "",
"company": "",
"active": "true",
"due_date": "",
"assignment_group": {
  "link": "https://instance.servicenow.com/api/now/table/sys_user_group/8a4dde73c6112278017a6a4baf547aa7",
  "value": "8a4dde73c6112278017a6a4baf547aa7"
},
"caller_id": {
  "link": "https://instance.servicenow.com/api/now/table/sys_user/46c6f9efa9fe198101ddf5eed9adf6e7",
  "value": "46c6f9efa9fe198101ddf5eed9adf6e7"
},
"knowledge": "false",
"made_sla": "false",
"comments_and_work_notes": "",
"parent_incident": "",
"state": "1",
"user_input": "",
"sys_created_on": "2015-11-01 22:05:30",
"approval_set": "",
"reassignment_count": "1",
"rfc": "",
"child_incidents": "",
"opened_at": "2015-11-02 22:04:15",
"short_description": "Can't access SFA software",
"order": "",
"sys_updated_by": "glide.maint",
"resolved_by": "",
"notify": "1",
"upon_reject": "",
"approval_history": "",
"problem_id": {
  "link": "https://instance.servicenow.com/api/now/table/problem/a9e4890bc6112276003d7a5a5c774a74",
  "value": "a9e4890bc6112276003d7a5a5c774a74"
},
"work_notes": "",
"calendar_duration": "",
"close_code": "",
"sys_id": "a9e30c7dc61122760116894de7bcc7bd"}
Example: Python request

Retrieve the first record from the Incident table.

```python
# Need to install requests package for python
import requests

# Set the request parameters
url =
'https://instance.servicenow.com/api/now/table/incident/a9e30c7dc61122760116894de7bdc7bd'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'
```
# Set proper headers
headers = {'Accept': 'application/xml'}
# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)
# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()
# Decode the XML response into a dictionary and use the data
print(response.content)
The response contains the name-value pairs for the requested record.

<?xml version="1.0" encoding="UTF-8"?>
<response>
    <result>
        <upon_approval />
        <location>
            <link>https://instance.servicenow.com/api/now/table/cmn_location/105cf7f3c611227501e75e08b14a38ba</link>
            <value>105cf7f3c611227501e75e08b14a38ba</value>
        </location>
        <expected_start />
        <reopen_count />
        <close_notes />
        <additional_assignee_list />
        <impact>1</impact>
        <urgency>3</urgency>
        <correlation_id />
        <priority>3</priority>
        <sys_tags />
        <sys_domain>
            <link>https://instance.servicenow.com/api/now/table/sys_user_group/global</link>
            <value>global</value>
        </sys_domain>
        <description />
        <group_list />
        <delivery_plan />
        <sys_mod_count>4</sys_mod_count>
    </result>
</response>
Table - PATCH /now/table/{tableName}/{sys_id}
Updates the specified record with the name-value pairs included in the request body.

**URL format**

Versioned URL: /api/now/{api_version}/table/{tableName}/{sys_id}
Default URL: /api/now/table/{tableName}/{sys_id}

**Supported request parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sys_id</td>
<td>Sys_id of the record to update.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>tableName</td>
<td>Name of the table in which the specified record is located.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_display_value</td>
<td>Data retrieval operation for reference and choice fields. Based on this value, retrieves the display value and/or the actual value from the database.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• true: Returns the display values for all fields.</td>
</tr>
<tr>
<td></td>
<td>• false: Returns the actual values from the database.</td>
</tr>
<tr>
<td></td>
<td>• all: Returns both actual and display values.</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> There is no preferred method for setting this parameter. However,</td>
</tr>
<tr>
<td></td>
<td>specifying the display value may cause performance issues since it is not</td>
</tr>
<tr>
<td></td>
<td>reading directly from the database and may include referencing other fields</td>
</tr>
<tr>
<td></td>
<td>and records. For more information on display values and actual values, see</td>
</tr>
<tr>
<td></td>
<td>Table API FAQs (KB0534905).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_fields</td>
<td>Comma-separated list of fields to return in the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_input_display_value</td>
<td>Flag that indicates whether to set field values using the display value or</td>
</tr>
<tr>
<td></td>
<td>the actual value.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Treats input values as display values and they are manipulated so</td>
</tr>
<tr>
<td></td>
<td>they can be stored properly in the database.</td>
</tr>
<tr>
<td></td>
<td>• false: Treats input values as actual values and stored them in the</td>
</tr>
<tr>
<td></td>
<td>database without manipulation.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_query_no_domain</td>
<td>Flag that indicates whether to restrict the record search to only the domains for which the logged in user is configured.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• false: Exclude the record if it is in a domain that the currently logged in user is not configured to access.</td>
</tr>
<tr>
<td></td>
<td>• true: Include the record even if it is in a domain that the currently logged in user is not configured to access.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>

Note:
- If this parameter is set to true, pay attention to input values, especially date values, as these are interpreted as being supplied via the user time zone preference and are transformed into UTC format.
- To set the value of an encrypted field, you must set this parameter to true. If this parameter is not set to true, values submitted to encrypted fields are not saved. Additionally, the requesting user must have the appropriate encryption context prior to submitting the request. Encrypted fields are hidden for users without the appropriate encryption context. For more information on display values and actual values, see Table API FAQs (KB0534905). For more information on field encryption see Encryption support.
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note: The sysparm_query_no_domain parameter is available only to system administrators or users who have the query_no_domain_table_api role.</td>
<td></td>
</tr>
</tbody>
</table>
| sysparm_view          | UI view for which to render the data. Determines the fields returned in the response. Valid values:  
  • desktop  
  • mobile  
  • both  
  If you also specify the sysparm_fields parameter, it takes precedent.  
  Data type: String |

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| name-value pairs| Field name and the new value for each parameter to update in the specified record.  
  Note: All fields within a record may not be available for update. For example, fields that have a prefix of "sys_" are typically system parameters that are automatically generated and cannot be updated. |

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.
Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>X-no-response-body</td>
<td>By default, responses include body content detailing the modified record. Set this request header to true to suppress the response body.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Indicates that the request completed successfully.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Field names and values of all parameters within the specified record or those specified in the query parameters.</td>
</tr>
</tbody>
</table>

Example: cURL request

Update a record in the Incident table.
The response contains the name-value pairs for the updated record.

```json
{
  "result": {
    "upon_approval": "proceed",
    "location": {
      "link": "https://instance.servicenow.com/api/now/table/cmn_location/108752c8c611227501d4ab0e392ba97f",
      "value": "108752c8c611227501d4ab0e392ba97f"
    },
    "expected_start": "",
    "reopen_count": "",
    "close_notes": "",
    "additional_assignee_list": "",
    "impact": "1",
    "urgency": "1",
    "correlation_id": "",
    "sys_tags": "",
    "sys_domain": {
      "link": "https://instance.servicenow.com/api/now/table/sys_user_group/global",
      "value": "global"
    },
    "description": "",
    "group_list": "",
    "priority": "1",
    "delivery_plan": "",
    "sys_mod_count": "7",
    "work_notes_list": "",
    "business_service": "",
    "follow_up": "",
    "closed_at": "",
    "sla_due": "2017-07-05 05:58:24",
  }
}
```
"short_description": "Can't access Exchange server - is it down?",
"order": "",
"sys_updated_by": "admin",
"resolved_by": "",
"notify": "1",
"upon_reject": "cancel",
"approval_history": "",
"problem_id": "",
"work_notes": "",
"calendar_duration": "",
"close_code": "",
"sys_id": "ef43c6d40a0b5700c77f9bf387afe3",
"approval": "not requested",
"caused_by": "",
"severity": "3",
"sys_created_by": "glide.maint",
"resolved_at": "",
"assigned_to": {
  "link": 
  "https://instance.servicenow.com/api/now/table/sys_user/681b365ec0a80164000fb0b05854a0cd",
  "value": "681b365ec0a80164000fb0b05854a0cd"
},
"business_stc": "",
"wf_activity": "",
"sys_domain_path": "/",
"cmdb_ci": {
  "link": 
  "https://instance.servicenow.com/api/now/table/cmdb_ci/281190e3c0a8000b003f593aa3f20ca6",
  "value": "281190e3c0a8000b003f593aa3f20ca6"
},
"opened_by": {
  "link": "https://instance.servicenow.com/api/now/table/sys_user/glide.maint",
  "value": "glide.maint"
},
"subcategory": "",
"rejection_goto": "",
"sys_class_name": "incident",
"watch_list": "",
"time_worked": "",
"contact_type": "phone",
"escalation": "3",
"comments": ""
Example: Python request

Update a record in the Incident table.

```python
# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/table/incident/ef43c6d40a0a0b5700c77f9bf387afe3'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Content-Type': 'application/xml', 'Accept': 'application/xml'}

# Do the HTTP request
response = requests.patch(url, auth=(user, pwd), headers=headers,
                          data='"<request><entry><assigned_to>681b365ec0a80164000fb0b05854a0cd</assigned_to><urgency>1</urgency><comments>Elevating urgency, this is a blocking issue</comments></entry></request>"

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)
```

The response contains the name-value pairs for the updated record.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <result>
    <upon_approval>proceed</upon_approval>
  </result>
</response>
```
<assigned_to/>

<link>https://instance.servicenow.com/api/now/table/sys_user/681b365ec0a80164000fb0b5854a0cd</link>

<value>681b365ec0a80164000fb0b5854a0cd</value>

</assigned_to>
<resolved_at />
<business_stc />
<cmdb_ci/>

<link>https://instance.servicenow.com/api/now/table/cmdb_ci/281190e3c0a8000b003f593aa3f20ca6</link>

<value>281190e3c0a8000b003f593aa3f20ca6</value>

</cmdb_ci>
<sys_domain_path>/</sys_domain_path>
<wf_activity />

<opened_by>
<link>https://instance.servicenow.com/api/now/table/sys_user/glide.maint</link>

<value>glide.maint</value>

</opened_by>
<rejection_goto />
<subcategory />
<sys_class_name>incident</sys_class_name>
<watch_list />
<escalation>3</escalation>
<contact_type>phone</contact_type>
<time_worked />
<comments />

</result>
</response>

Table - POST /now/table/{tableName}

Inserts one record in the specified table. Multiple record insertion is not supported by this method.

URL format

Versioned URL: /api/now/{api_version}/table/{tableName}
Default URL: /api/now/table/{tableName}
## Supported request parameters

### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>tableName</td>
<td>Name of the table in which to save the record.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_display_value</td>
<td>Data retrieval operation for reference and choice fields. No retrieval of display values required.</td>
</tr>
<tr>
<td></td>
<td>Based on this value, retrieves the display value and/or the actual value from the database.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Returns the display values for all fields.</td>
</tr>
<tr>
<td></td>
<td>• false: Returns the actual values from the database.</td>
</tr>
<tr>
<td></td>
<td>• all: Returns both actual and display values.</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>

**Note:** There is no preferred method for setting this parameter. However, specifying the display value may cause performance issues since it is not reading directly from the database and may include referencing other fields and records. For more information on display values and actual values, see Table API FAQs (KB0534905).
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_exclude_reference_link</td>
<td>Flag that indicates whether to exclude Table API links for reference fields.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Exclude Table API links for reference fields.</td>
</tr>
<tr>
<td></td>
<td>• false: Include Table API links for reference fields.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>sysparm_fields</td>
<td>Comma-separated list of fields to return in the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>sysparm_input_display_value</td>
<td>Flag that indicates whether to set field values using the display value or</td>
</tr>
<tr>
<td></td>
<td>the actual value.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Treats input values as display values and they are manipulated so</td>
</tr>
<tr>
<td></td>
<td>they can be stored properly in the database.</td>
</tr>
<tr>
<td></td>
<td>• false: Treats input values as actual values and stored them in the</td>
</tr>
<tr>
<td></td>
<td>database without manipulation.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
</tbody>
</table>
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **sysparm_view** | UI view for which to render the data. Determines the fields returned in the response. Valid values:  
  - desktop  
  - mobile  
  - both  
  If you also specify the `sysparm_fields` parameter, it takes precedent.  
  Data type: String |

**Note:**

- If this parameter is set to `true`, pay attention to input values, especially date values, as these are interpreted as being supplied via the user time zone preference and are transformed into UTC format.

- To set the value of an encrypted field, you must set this parameter to `true`. If this parameter is not set to `true`, values submitted to encrypted fields are not saved. Additionally, the requesting user must have the appropriate encryption context prior to submitting the request. Encrypted fields are hidden for users without the appropriate encryption context. For more information on display values and actual values, see Table API FAQs (KB0534905). For more information on field encryption see Encryption support.
Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Field name and the associated value for each parameter to define in the specified record.</td>
</tr>
</tbody>
</table>

**Note:** All fields within a record may not be available for update. For example, fields that have a prefix of "sys_" are typically system parameters that are automatically generated and cannot be updated.

Fields that are not specified and not auto generated by the system are set to the associated data type's null value.

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
<tr>
<td>X-no-response-body</td>
<td>By default, responses include body content detailing the new record. Set this header to true in the request to suppress the response body.</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>Location of the created resource.</td>
</tr>
</tbody>
</table>
Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Indicates that the request completed successfully.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Field names and values of all parameters within the newly created record or those specified in the query parameters.</td>
</tr>
</tbody>
</table>

Example: cURL request

Insert a new record into the Incident table.

curl "https://instance.servicenow.com/api/now/table/incident" \
    --request POST \
    --header "Accept:application/json" \
    --header "Content-Type:application/json" \
    --data "{"short_description":'Unable to connect to office wifi',
             "assignment_group":'287ebd7da9fe198100f92cc8d1d2154e',
             "urgency":'2',
             "impact":'2'}" \
    --user 'username':'password'

The response contains the name-value pairs for the new record.

```json
{
    "result": {
        "upon_approval": "proceed",
        "location": "",
        "expected_start": "",
        "reopen_count": "0",
        "close_notes": "",
        "additional_assignee_list": "",
        "impact": "2",
        "urgency": "2",
        "correlation_id": "",
        "sys_tags": "",
        "sys_domain": {
```
"reassignment_count": "0",
"rfc": "",
"child_incidents": "0",
"opened_at": "2016-01-22 14:28:24",
"short_description": "Unable to connect to office wifi",
"order": "",
"sys_updated_by": "admin",
"resolved_by": "",
"notify": "1",
"uponReject": "cancel",
"approval_history": "",
"problem_id": "",
"work_notes": "",
"calendar_duration": "",
"close_code": "",
"sys_id": "c537bae64f411200ad9f8e18110c76e",
"approval": "not requested",
"caused_by": "",
"severity": "3",
"sys_created_by": "admin",
"resolved_at": "",
"assigned_to": "",
"business_stc": "",
"wf_activity": "",
"sys_domain_path": "",
"cmdb_ci": "",
"opened_by": {
  "link": "https://instance.servicenow.com/api/now/table/sys_user/6816f79cc0a8016401c5a33be04be441",
  "value": "6816f79cc0a8016401c5a33be04be441"
},
"subcategory": "",
"rejection_goto": "",
"sys_class_name": "incident",
"watch_list": "",
"time_worked": "",
"contact_type": "phone",
"escalation": "0",
"comments": ""
}
Example: Python request

Insert a new record into the Incident table.

# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/table/incident'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {"Content-Type":"application/xml","Accept":"application/json"}

# Do the HTTP request
response = requests.post(url, auth=(user, pwd), headers=headers,
data="<request><entry><short_description>Unable to connect to office wifi</short_description><assignment_group>287ebd7da9fe198100f92cc8d1d2154e</assignment_group><urgency>2</urgency><impact>2</impact></entry></request>"
)

# Check for HTTP codes other than 201
if response.status_code != 201:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()

# Decode the JSON response into a dictionary and use the data
data = response.json()
print(data)

The response contains the name-value pairs for the new record.

{ "result": { "upon_approval": "proceed", "location": "", "expected_start": "", "reopen_count": "0", "close_notes": "", "additional_assignee_list": "", "impact": "2", "urgency": "2", "correlation_id": "" }}
Table - PUT /now/table/{tableName}/{sys_id}
Updates the specified record with the request body.

**URL format**

Versioned URL: /api/now/{api_version}/table/{tableName}/{sys_id}
Default URL: /api/now/table/{tableName}/{sys_id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest.</td>
</tr>
<tr>
<td>sys_id</td>
<td>Unique identifier of the record to update.</td>
</tr>
<tr>
<td>tableName</td>
<td>Name of the table in which the record is located.</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_display_value</td>
<td>Data retrieval operation for reference and choice fields. Based on this value, retrieves the display value and/or the actual value from the database.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Returns the display values for all fields.</td>
</tr>
<tr>
<td></td>
<td>• false: Returns the actual values from the database.</td>
</tr>
<tr>
<td></td>
<td>• all: Returns both actual and display values.</td>
</tr>
</tbody>
</table>
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_exclude_reference_link</td>
<td>Flag that indicates whether to exclude Table API links for reference fields.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Exclude Table API links for reference fields.</td>
</tr>
<tr>
<td></td>
<td>• false: Include Table API links for reference fields.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td></td>
<td>Default: false</td>
</tr>
<tr>
<td>sysparm_fields</td>
<td>Comma-separated list of fields to return in the response.</td>
</tr>
<tr>
<td>sysparm_input_display_value</td>
<td>Flag that indicates whether to set field values using the display value or the actual value.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Treats input values as display values and they are manipulated so they can be stored properly in the database.</td>
</tr>
<tr>
<td></td>
<td>• false: Treats input values as actual values and stored them in the database without manipulation.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
</tbody>
</table>

**Note:** There is no preferred method for setting this parameter. However, specifying the display value may cause performance issues since it is not reading directly from the database and may include referencing other fields and records. For more information on display values and actual values, see [Table API FAQs (KB0534905)](KB0534905).
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_query_no_domain</td>
<td>Flag that indicates whether to restrict the record search to only the domains for which the logged in user is configured.</td>
</tr>
</tbody>
</table>

**Default**: false

**Note:**
- If this parameter is set to `true`, pay attention to input values, especially date values, as these are interpreted as being supplied via the user time zone preference and are transformed into UTC format.
- To set the value of an encrypted field, you must set this parameter to `true`. If this parameter is not set to `true`, values submitted to encrypted fields are not saved. Additionally, the requesting user must have the appropriate encryption context prior to submitting the request. Encrypted fields are hidden for users without the appropriate encryption context. For more information on display values and actual values, see Table API FAQs (KB0534905). For more information on field encryption see Encryption support.

Data type: Boolean

**Default**: false
### Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysparm_view</td>
<td>UI view for which to render the data. Determines the fields returned in the response. Valid values: desktop, mobile, both. If you also specify the sysparm_fields parameter, it takes precedence. Data type: String</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The <code>sysparm_query_no_domain</code> parameter is available only to system administrators or users who have the <code>query_no_domain_table_api</code> role.</td>
</tr>
</tbody>
</table>

### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>Name-value pairs for the field(s) to update in the associated table. For example, to update the short description file, enter a name-value pair similar to the following: --data &quot;{&quot;&quot;short_description&quot;: &quot;my short desc&quot;}&quot; .</td>
</tr>
</tbody>
</table>

### Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

#### Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: <code>application/json</code> or <code>application/xml</code>.</td>
</tr>
</tbody>
</table>
Request headers (continued)

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default:</td>
<td>application/json</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Supported types: application/json or</td>
</tr>
<tr>
<td></td>
<td>application/xml.</td>
</tr>
<tr>
<td>X-no-response-</td>
<td>By default, responses include body content detailing the modified record.</td>
</tr>
<tr>
<td>body</td>
<td>Set this header to true in the request to suppress the response body.</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Indicates the request completed successfully.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name-value pairs</td>
<td>All fields (not just modified) with their associated values for the change request.</td>
</tr>
</tbody>
</table>

Example: cURL request

Update a record in the Incident table.

curl
   "https://instance.servicenow.com/api/now/table/incident/ef43c6d40a0b5700c77f9bf387afe3"
   --request PUT
   --header "Accept:application/json"
The response contains the name-value pairs for the updated record.

```json
{
    "result": {
        "upon_approval": "proceed",
        "location": {
            "link": "https://instance.servicenow.com/api/now/table/cmn_location/108752c8c611227501d4ab0e392ba97f",
            "value": "108752c8c611227501d4ab0e392ba97f"
        },
        "expected_start": "",
        "reopen_count": "",
        "close_notes": "",
        "additional_assignee_list": "",
        "impact": "1",
        "urgency": "1",
        "correlation_id": "",
        "sys_tags": "",
        "sys_domain": {
            "link": "https://instance.servicenow.com/api/now/table/sys_user_group/global",
            "value": "global"
        },
        "description": "",
        "group_list": "",
        "priority": "1",
        "delivery_plan": "",
        "sys_mod_count": "7",
        "work_notes_list": "",
        "business_service": "",
        "follow_up": "",
        "closed_at": "",
        "sla_due": "2017-07-05 05:58:24",
        "delivery_task": "",
        "sys_updated_on": "2016-01-22 14:12:37",
        "parent": "",
        "work_end": "",
        "number": "INC0000050",
    }
}
```
Example: Python request

Update a record in the Incident table.
# Need to install requests package for python
import requests

# Set the request parameters
url = 'https://instance.servicenow.com/api/now/table/incident/ef43c6d40a0b5700c77f9bf387afe3'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Content-Type': 'application/xml', 'Accept': 'application/xml'}

# Do the HTTP request
response = requests.put(url, auth=(user, pwd), headers=headers,
data='"<request><entry><assigned_to>681b365ec0a80164000fb0b05854a0cd</assigned_to><urgency>1</urgency><comments>Elevating urgency, this is a blocking issue</comments></entry></request>"

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.content)
    exit()

# Decode the XML response into a dictionary and use the data
print(response.content)

The response contains the name-value pairs for the updated record.

<?xml version="1.0" encoding="UTF-8"?>
<response>
    <result>
        <upon_approval>proceed</upon_approval>
        <location>
            <link>https://instance.servicenow.com/api/now/table/cmn_location/108752c8c611227501d4ab0e392ba97f</link>
            <value>108752c8c611227501d4ab0e392ba97f</value>
        </location>

        <expected_start />
        <reopen_count />
        <close_notes />
    </result>
</response>
<value>8a5055c9c61122780043563ef53438e3</value>
</assignment_group>
<caller_id>
<link>https://instance.servicenow.com/api/now/table/sys_user/5b7c200d0a640069006b3845b5d0fa7c</link>
<value>5b7c200d0a640069006b3845b5d0fa7c</value>
</caller_id>

<knowledge>false</knowledge>
<made_sla>true</made_sla>
<comments_and_work_notes />
<parent_incident />
<state>2</state>
<user_input />
<sys_created_on>2015-11-02 18:05:40</sys_created_on>
<approval_set />
<reassignment_count>0</reassignment_count>
<rfc />
<opened_at>2015-11-02 21:58:24</opened_at>
<child_incidents />
<order />
<short_description>Can't access Exchange server - is it down?</short_description>
<resolved_by />
<sys_updated_by>admin</sys_updated_by>
<notify>1</notify>
<upon_reject>cancel</upon_reject>
<approval_history />
<problem_id />
<calendar_duration />
<work_notes />
<close_code />
<approval>not requested</approval>
<sys_id>ef43c6d40a0b5700c77f9bf387afe3</sys_id>
<caused_by />
<severity>3</severity>
<sys_created_by>glide.maint</sys_created_by>
<assigned_to>
<link>https://instance.servicenow.com/api/now/table/sys_user/681b365ec0a80164000fb0b05854a0cd</link>
<value>681b365ec0a80164000fb0b05854a0cd</value>
</assigned_to>
<resolved_at />
<business_stc />
Trouble Ticket Open API

Creates, updates, and retrieves data from the Case [sn_customerservice_case] and Incident [incident] tables.

Use this API to manage ticket information between external ticketing systems and the Now Platform. The Trouble Ticket Open API is a ServiceNow® implementation of the TM Forum Trouble Ticket Management API REST specification. This implementation is based on the TMF621 Trouble Ticket Management API REST Specification Release 19, June 2019.

This API is included in the Telecommunications Assurance Workflows application, which is available on the ServiceNow Store. The Customer Service (com.sn_customerservice) and Customer Service Install Base Management (com.snc.install_base) applications are required to use this API with the Case [sn_customerservice_case] table.

This API is provided within the sn_ind_tsm_sdwan namespace.

The calling user must have the ticket_integrator role.
Extending the Trouble Ticket Open API

The Trouble Ticket Open API can be extended by editing the following script includes. These script includes should only be edited with an understanding of the consequences of the changes.

- **TMFTroubleTicketAPIConstants**: Contains constants and required parameter information.
- **TMFTroubleTicketAPIUtil**: Contains functions to handle POST, GET, and PATCH requests.
- **TroubleTicketProcessorOOB**: Contains helper functions that support functions in TMFTroubleTicketAPIUtil.
- **TroubleTicketProcessor**: An empty script include file. Use this file to define any functions that you want to override from TroubleTicketProcessorOOB.

Extend the Trouble Ticket Open API to make the following customizations.

**Required parameters**

To change which request body parameters are required or not required, update the TMFTroubleTicketAPIConstants script include. By default, the request body to create a ticket (POST) requires the parameters **description**, **severity**, and **ticketType**. By default, the request body to update a ticket (PATCH) doesn't have required parameters.

- **TMFTroubleTicketAPIConstants.TROUBLE_TICKET_CREATION_SCHEMA**: Sets the required request body parameters to create a ticket.
- **TMFTroubleTicketAPIConstants.TROUBLE_TICKET_UPDATE_SCHEMA**: Sets the required request body parameters to update a ticket.

In this example, the **description**, **severity**, **status**, and **ticketType** parameters are required to create a ticket.

```javascript
// TMFTroubleTicketAPIConstants
TMFTroubleTicketAPIConstants.TROUBLE_TICKET_CREATION_SCHEMA = {
  "title": "CreateTroubleTicket",
  "type": "object",
  "properties": {
    "description": {
      "type": "string"
    },
    "severity": {
      "type": "string"
    },
    "status": {
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Request body validation

To perform additional validation on the request body, override TroubleTicketProcessorOOB functions. TroubleTicketProcessorOOB contains the following six helper functions that return true by default. These functions are called in TMFTroubleTicketAPIUtil.

- verifyGetRequestForCase() - Called by processGetRequestForCase().
- verifyPatchRequestForCase() - Called by processPatchRequestForCase().
- verifyPostRequestForCase() - Called by processPostRequestForCase().
- verifyGetRequestForIncident() - Called by processGetRequestForIncident().
- verifyPatchRequestForIncident() - Called by processPatchRequestForIncident().
- verifyPostRequestForIncident() - Called by processPostRequestForIncident().

If a helper function returns false, it stops the API operation. To apply custom validation, override TroubleTicketProcessorOOB helper functions by creating functions with identical names and parameters in TroubleTicketProcessor. These new TroubleTicketProcessor functions will be called by TMFTroubleTicketAPIUtil to replace the default TroubleTicketProcessorOOB helper functions.

In this example, a function in TroubleTicketProcessor overrides a default function in TroubleTicketProcessorOOB to perform validation on the name attribute.
// TroubleTicketProcessor
var TroubleTicketProcessor = Class.create();
TroubleTicketProcessor.prototype =
Object.extendObject(TroubleTicketProcessorOOB, {
    // Define overriding functions here
    // Function name and parameters must be identical to the function it
    // overrides

    verifyPostRequestForCase: function(caseObject, apiResponseProcessor){
        // Returning false terminates the POST request
        // Make sure to assign error message and reason
        if (caseObject.name != "Hello world") {
            apiResponseProcessor.setMessage(TMFTroubleTicketAPIConstants.MESSAGES.CASE_CREATION_FAILURE);
            apiResponseProcessor.setReason("No reason needed");
            return false;
        }
    },

    type: 'TroubleTicketProcessor'
});

### Additional REST operations

To create additional operations beyond the existing GET, PATCH, and POST operations, create additional scripted REST resources for the Trouble Ticket Open API. The logic of the new scripted REST resources should be consistent with the existing operations. Define functions for the new operations in TMFTroubleTicketAPIUtil.

### Field mapping

When creating or updating records, the API maps request body parameters to case and incident record fields. When retrieving records, the API maps record fields to response object attributes. TroubleTicketProcessorOOB contains the following functions to map a PATCH or POST request body to a case or incident GlideRecord.

- `mapPatchRequestToCase()`
- `mapPostRequestToCase()`
- `mapPatchRequestToIncident()`
- `mapPostRequestToIncident()`
TroubleTicketProcessorOOB contains the following functions to map a case or incident GlideRecord to a JSON response object for GET, PATCH, or POST requests.

- `createGetResponseForCase()`
- `createPatchResponseForCase()`
- `createPostResponseForCase()`
- `createGetResponseForIncident()`
- `createPatchResponseForIncident()`
- `createPostResponseForIncident()`

Customize field mappings to add and retrieve data for additional case and incident fields, or to change the default mappings for fields. To customize the field mappings, override TroubleTicketProcessorOOB mapping functions by creating functions with identical names and parameters in TroubleTicketProcessor. These new TroubleTicketProcessor functions will be used by TMFTroubleTicketAPIUtil to replace the default TroubleTicketProcessorOOB mapping functions.

In this example, two functions in TroubleTicketProcessor override the default functions in TroubleTicketProcessorOOB to change the Description field mapping and add a mapping for the Contract field.

```javascript
// TroubleTicketProcessor
var TroubleTicketProcessor = Class.create();
TroubleTicketProcessor.prototype = Object.extendsObject(TroubleTicketProcessorOOB, {
  // Define overriding functions here
  // Function name and parameters must be identical to the function it overrides
  mapPostRequestToCase: function(caseGr, caseObject){
    // Override default mapping for the Description field
    caseGr.description = "All cases will be created with this description";
    // Add new mapping to the Contract field
    caseGr.contract = caseObject.contract;
  },
  createPostResponseForCase: function(caseGr, caseObject){
    // Override default mapping for the description attribute
```
Mapping logic for choice fields

The State, Priority, and contact_type fields are choice fields, where each choice is composed of a label and a corresponding value. For example, if the State field is set to New, the label is New and the value is 1. The mapping between choice labels and values can be changed by overriding the choice field mapping functions.

TroubleTicketProcessorOOB contains the following choice field mapping functions.

- transformCaseSeverity()
- transformCaseChannel()
- transformCaseStatus()
- transformIncidentSeverityToUrgency()
- transformIncidentSeverityToImpact()
- transformIncidentChannel()
- transformIncidentStatus()

Override TroubleTicketProcessorOOB choice field mapping functions by creating functions with identical names and parameters in TroubleTicketProcessor. These new TroubleTicketProcessor functions will be used by TMFTroubleTicketAPIUtil to replace the default TroubleTicketProcessorOOB choice field mapping functions. In this example, a function in TroubleTicketProcessor overrides a default function in TroubleTicketProcessorOOB to change the choice mapping for the Status field.

```javascript
// TroubleTicketProcessor
var TroubleTicketProcessor = Class.create();
TroubleTicketProcessor.prototype =
    Object.extendsObject(TroubleTicketProcessorOOB, {
        // Define overriding functions here
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
```javascript
// Function name and parameters must be identical to the function it overrides

transformCaseStatus: function(status) {
    if (status == "Draft")
        return 1;
    else return 10;
},

type: 'TroubleTicketProcessor'
});
```

**Trouble Ticket - GET /sn_ind_tsm_sdwan/troubleticket/{ticketType}/{id}**

Gets a record from the Case or Incident table.

**URL format**

Default URL: /api/sn_ind_tsm_sdwan/troubleticket/{ticketType}/{id}

**Supported request parameters**

**Path parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| ticketType | The type of ticket. This value determines whether a record is retrieved from the Case [sn_customerservice_case] or Incident [incident] table. Valid values:  
  • case  
  • incident |

Data type: String

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Sys_id of the case or incident record to retrieve. Data type: String</td>
</tr>
</tbody>
</table>

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fields</td>
<td>Comma-separated list of fields to return in the response. Invalid fields are ignored. If this value is not passed in, all fields are returned. Valid fields:</td>
</tr>
</tbody>
</table>
Query parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• channel</td>
<td></td>
</tr>
<tr>
<td>• creationDate</td>
<td></td>
</tr>
<tr>
<td>• description</td>
<td></td>
</tr>
<tr>
<td>• id</td>
<td></td>
</tr>
<tr>
<td>• lastUpdate</td>
<td></td>
</tr>
<tr>
<td>• name</td>
<td></td>
</tr>
<tr>
<td>• note</td>
<td></td>
</tr>
<tr>
<td>• relatedParty</td>
<td></td>
</tr>
<tr>
<td>• severity</td>
<td></td>
</tr>
<tr>
<td>• status</td>
<td></td>
</tr>
<tr>
<td>• ticketType</td>
<td></td>
</tr>
<tr>
<td>• @type</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
</tbody>
</table>
## Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

## Status codes

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
| 400         | Bad Request. A bad request type or malformed request was detected for one or more of the following reasons.  
  • Invalid path parameter.  
  • Invalid URI. |
| 404         | Record not found. A record associated with the ID is not found in the Case [sn_customerservice_case] or Incident [incident] table. |

## Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>@type</td>
<td>This value is always TroubleTicket. Data type: String</td>
</tr>
<tr>
<td>channel</td>
<td>The method of contact that the ticket was created through. Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;channel&quot;:{</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>channel.name</td>
<td>The name of the contact method. Maps to the Channel [contact_type] field for cases and the Contact type [contact_type] field for incidents.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>creationDate</td>
<td>The date that the case or incident record was created. Maps to the <strong>Created</strong> field. Data type: String</td>
</tr>
<tr>
<td>description</td>
<td>The description of the issue from the ticket. Maps to the <strong>Description</strong> field. Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>The sys_id of the case or incident record. Data type: String</td>
</tr>
<tr>
<td>lastUpdate</td>
<td>The date the record was last updated. Maps to the <strong>Updated</strong> field. Data type: String</td>
</tr>
<tr>
<td>name</td>
<td>The name of the trouble ticket, typically a short description of the issue. Maps to the <strong>Short description</strong> field. Data type: String</td>
</tr>
<tr>
<td>note</td>
<td>A list of all comments on the ticket. This list does not include work notes. Data type: Array</td>
</tr>
</tbody>
</table>

```
"note": [
  {
    "author": "String",
    "date": "String",
    "text": "String",
    "@type": "String"
  }
]
```

| note.@type   | The type of note. Only comments are returned (not work notes) so this value is always **comments**. Data type: String |
| note.author  | The name of the ServiceNow integration user who made the POST or PATCH request that created the comment.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>note.date</td>
<td>The date the comment was created in the ServiceNow instance. Data type: String</td>
</tr>
<tr>
<td>note.text</td>
<td>The comment text. Data type: String</td>
</tr>
<tr>
<td>relatedParty</td>
<td>Details about the companies and contacts associated with the ticket. Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;relatedParty&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;name&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;@referredType&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>relatedParty.@referredType</td>
<td>The type of related party. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• customer: Related party is a company or account.</td>
</tr>
<tr>
<td></td>
<td>• customer_contact: Related party is a caller or contact.</td>
</tr>
<tr>
<td>relatedParty.id</td>
<td>The sys_id for the related party. For cases, the sys_id is from the Contact [customer_contact] or Account [customer_account] tables. For incidents, the sys_id is from the User [sys_user] or Company [core_company] tables. Data type: String</td>
</tr>
<tr>
<td>relatedParty.name</td>
<td>The name of the related party. For cases, maps to the Contact or Account field. For incidents, maps to the Caller or Company field. Data type: String</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>severity</td>
<td>The severity of the issue described by the trouble ticket. Maps to the <strong>Priority</strong> field.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>The current status of the trouble ticket. Maps to the <strong>State</strong> field.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>ticketType</td>
<td>The type of ticket (case or incident).</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

This example gets the name, description, severity, and status of a case.

```bash
curl
   "https://instance.servicenow.com/api/sn_ind_tsm_sdwan/troubleticket/case/a78e2c8cdb68b41015364c9b0b96193e?fields=name%2Cdescription%2Cseverity%2Cstatus"
   --request GET \
   --header "Accept:application/json" \
   --user 'username':'password'

{
   "name": "Short description of issue",
   "description": "Example description of an issue",
   "severity": "2 - High",
   "status": "Open"
}
```

**Trouble Ticket - PATCH /sn_ind_tsm_sdwan/troubleticket/{ticketType}/{id}**

Updates a record in the Case or Incident table.

**URL format**

Default URL: /api/sn_ind_tsm_sdwan/troubleticket/{ticketType}/{id}
### Supported request parameters

#### Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| ticketType| The type of ticket. This value determines whether a record is updated in the Case [sn_customerservice_case] or Incident [incident] table. Valid values:  
  - `case`  
  - `incident`  
  Data type: String |
| id        | The sys_id of the case or incident record to update.                         |
|           | Data type: String                                                           |

#### Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

#### Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel</td>
<td>The method of contact that the ticket was created through.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>channel.name</td>
<td>The name of the contact method. Maps to the Channel [contact_type] field for cases and the Contact type [contact_type] field for incidents. Valid values are the choices for the contact_type field. You can provide the choice label or value. For example, Virtual Agent or virtual_agent.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the trouble ticket, typically a short description of the issue. Maps to the <strong>Short description</strong> field.</td>
</tr>
</tbody>
</table>
| name.@type      | The type of note. This value determines whether the note is recorded in the **Work notes** or **Additional comments** field. Valid values:  
• work_notes  
• comments                                                                                                                   |
| note            | A list of work notes and comments to add to the ticket.                                                                                                                                                        |
| note.@type      | The type of note. This value determines whether the note is recorded in the **Work notes** or **Additional comments** field. Valid values:  
• work_notes  
• comments                                                                                                                   |
| note.text       | The note text. Maps to the **Work notes** or **Additional comments** field.                                                                                                                                 |
| relatedParty    | Details about the companies and contacts associated with the ticket. For cases, this parameter is used to populate the **Contact** and **Account** fields. For incidents, this parameter is used to populate the **Caller** and **Company** fields. |
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If you provide a contact or a caller, their account or company is automatically added.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>&quot;relatedParty&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;@referredType&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td>relatedParty.@referredType</td>
<td>The type of related party.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• customer: Related party is a company or account.</td>
</tr>
<tr>
<td></td>
<td>• customer_contact: Related party is a caller or contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>relatedParty.id</td>
<td>The sys_id for the related party. For cases, the sys_id is from the Contact [customer_contact] or Account [customer_account] tables. For incidents, the sys_id is from the User [sys_user] or Company [core_company] tables.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>severity</td>
<td>The severity of the issue described by the trouble ticket. Maps to the Priority field.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• 1: Critical.</td>
</tr>
<tr>
<td></td>
<td>• 2: High.</td>
</tr>
<tr>
<td></td>
<td>• 3: Moderate.</td>
</tr>
<tr>
<td></td>
<td>• 4: Low.</td>
</tr>
<tr>
<td></td>
<td>• 5: Planning. Incident only.</td>
</tr>
</tbody>
</table>
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority</td>
<td>Valid values are the choices for the Priority field. You must provide the choice value only. For example, 1. If an invalid value is provided, the severity is set to Low. Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>The current status of the trouble ticket. Maps to the State field. Valid values are the choices for the State field. You can provide the choice label or value. For example, New or 1. Data type: String</td>
</tr>
</tbody>
</table>

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.
### Status codes

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>200</strong></td>
<td>Successful. The request was successfully processed.</td>
</tr>
</tbody>
</table>
| **400**     | Bad Request. A bad request type or malformed request was detected for one or more of the following reasons.  
• Invalid path parameter.  
• Invalid URI. |
| **404**     | Record not found. A record associated with the ID is not found in the Case [sn_customerservice_case] or Incident [incident] table. |

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| @type        | This value is always `TroubleTicket`.  
Data type: String |
| channel      | The method of contact that the ticket was created through.  
Data type: Object |
| channel.name | The name of the contact method.  
Data type: String |
| creationDate | The date the record was created.  
Data type: String |
| description  | The description of the issue from the ticket.  
Data type: String |
| id           | The sys_id of the case or incident record.  
Data type: String |
<p>| lastUpdate   | The date the record was last updated. |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>The name of the trouble ticket, typically a short description of the issue. Data type: String</td>
</tr>
<tr>
<td>note</td>
<td>A list of all comments on the ticket. This list does not include work notes. Data type: Array</td>
</tr>
</tbody>
</table>
|            | "note": [
|            |   {
|            |     "author": "String",
|            |     "date": "String",
|            |     "text": "String",
|            |     @type": "String"
|            |   }
| note.@type | The type of note. Only comments are returned (not work notes) so this value is always comments. Data type: String |
| note.author| The name of the ServiceNow integration user who made the POST or PATCH request that created the comment. Data type: String |
| note.date  | The date the comment was created in the ServiceNow instance. Data type: String |
| note.text  | The comment text. Data type: String |
| relatedParty| Details about the companies and contacts associated with the ticket. Data type: Array |

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relatedParty.@referredType</td>
<td>The type of related party. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• customer: Related party is a company or account.</td>
</tr>
<tr>
<td></td>
<td>• customer_contact: Related party is a caller or contact.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>relatedParty.id</td>
<td>The sys_id for the related party. For cases, the sys_id is from the Contact [customer_contact] or Account [customer_account] tables. For incidents, the sys_id is from the User [sys_user] or Company [core_company] tables. Data type: String</td>
</tr>
<tr>
<td>relatedParty.name</td>
<td>The name of the related party. Data type: String</td>
</tr>
<tr>
<td>severity</td>
<td>The severity of the issue described by the trouble ticket. Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>The current status of the trouble ticket. Data type: String</td>
</tr>
<tr>
<td>ticketType</td>
<td>The type of ticket (case or incident). Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

```bash
curl -H "Content-Type: application/json" \
"https://instance.servicenow.com/api/sn_ind_tsm_sdwan/troubleticket/case/a78e2c8c8db68b41015364c9b0b96193e"
    -X PATCH
    -H "Accept: application/json"
```
--header "Content-Type:application/json" \
--data "{
   "description":"Updated description",
   "name":"Updated name",
   "note": [
      {
         "text":"New comment",
         "@type": "comments"
      }
   ],
   "relatedParty": [
      {
         "id":"dc0185d94f341200025ba3618110c77c",
         "@referredType": "customer_contact"
      }
   ]
}" \
--user 'username':'password'

{
   "id": "a78e2c8c8c68b41015364c9b0b96193e",
   "creationDate": "2021-05-30 23:26:42",
   "lastUpdate": "2021-06-01 00:49:50",
   "description": "Updated description",
   "severity": "2 - High",
   "ticketType": "Case",
   "name": "Updated name",
   "status": "Open",
   "channel": {
      "name": "email"
   },
   "relatedParty": [  
      {
         "id": "86837a386f0331003b3c498f5d3ee4ca",
         "name": "Cindy's account",
         "@referredType": "customer"
      },
      {
         "id": "dc0185d94f341200025ba3618110c77c",
         "name": "Cindy Contact",
         "@referredType": "customer_contact"
      }
   ]
}
Trouble Ticket - POST /sn_ind_tsm_sdwan/troubleticket/{ticketType}

Creates a record for the trouble ticket in the Case or Incident table.

URL format

Default URL: /api/sn_ind_tsm_sdwan/troubleticket/{ticketType}

Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ticketType</td>
<td>The type of ticket. This value determines whether the record is created in the Case [sn_customerservice_case] or Incident [incident] table. Valid values: • case • incident</td>
</tr>
</tbody>
</table>

Data type: String
## Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

## Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| channel    | The method of contact that the ticket was created through.  
Data type: Object  
"channel":{  
  "name": "String"
} |
| channel.name | The name of the contact method. Maps to the Channel [contact_type] field for cases and the Contact type [contact_type] field for incidents.  
Valid values are the choices for the contact_type field. You can provide the choice label or value. For example, Virtual Agent or virtual_agent.  
Data type: String |
| description | Required. A description of the issue. Maps to the Description field.  
Data type: String |
| name       | The name of the trouble ticket, typically a short description of the issue. Maps to the Short description field.  
Data type: String |
| note       | A list of work notes and comments to add to the ticket.  
Data type: Array  
"note":[
  
  
  "text": "String",
] |
### Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>note.@type</code></td>
<td>The type of note. This value determines whether the note is recorded in the <strong>Work notes</strong> or <strong>Additional comments</strong> field. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• work_notes</td>
</tr>
<tr>
<td></td>
<td>• comments</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td><code>note.text</code></td>
<td>The note text. Maps to the <strong>Work notes</strong> or <strong>Additional comments</strong> field.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td><code>relatedParty</code></td>
<td>Details about the companies and contacts associated with the ticket. For cases, this parameter is used to populate the <strong>Contact</strong> and <strong>Account</strong> fields. For incidents, this parameter is used to populate the <strong>Caller</strong> and <strong>Company</strong> fields. If you provide a contact or a caller, their account or company is automatically added.</td>
</tr>
<tr>
<td>Data type: Array</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;relatedParty&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;id&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;@referredType&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td><code>relatedParty.@referredType</code></td>
<td>The type of related party. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• customer: Related party is a company or account.</td>
</tr>
<tr>
<td></td>
<td>• customer_contact: Related party is a caller or contact.</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>relatedParty.id</td>
<td>The sys_id for the related party. For cases, the sys_id is from the Contact [customer_contact] or Account [customer_account] tables. For incidents, the sys_id is from the User [sys_user] or Company [core_company] tables.</td>
</tr>
</tbody>
</table>
| severity            | Required. The severity of the issue described by the trouble ticket. Maps to the **Priority** field. Valid values:  
|                     | • 1: Critical.                                                                             |
|                     | • 2: High.                                                                                 |
|                     | • 3: Moderate.                                                                             |
|                     | • 4: Low.                                                                                  |
|                     | • 5: Planning. Incident only.                                                              |
| status              | The current status of the trouble ticket. Maps to the **State** field.                      |
|                     | Valid values are the choices for the **State** field.                                       |
|                     | You can provide the choice label or value. For example, **New** or 1.                      |
| ticketType          | Required. The type of ticket. This value determines whether the record is created in the Case [sn_customerservice_case] or Incident [incident] table. |
|                     | Valid values:                                                                             |
Request body parameters (XML or JSON) (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>case</td>
<td></td>
</tr>
<tr>
<td>incident</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

**Headers**

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see [Supported REST API headers](#).

**Request headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports <code>application/json</code>.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports <code>application/json</code>.</td>
</tr>
</tbody>
</table>

**Response headers**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see [REST API HTTP response codes](#).

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected for one or more of the following reasons.</td>
</tr>
</tbody>
</table>
### Status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Empty payload.</td>
</tr>
<tr>
<td></td>
<td>• Invalid payload. Mandatory field missing.</td>
</tr>
<tr>
<td></td>
<td>• Invalid ticketType or ticketType does not match path parameter ticketType. For example, ticketType is case but path parameter is incident.</td>
</tr>
</tbody>
</table>

### Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>@type</td>
<td>This value is always TroubleTicket. Data type: String</td>
</tr>
<tr>
<td>channel</td>
<td>The method of contact that the ticket was created through. Data type: Object</td>
</tr>
<tr>
<td>channel.name</td>
<td>The name of the contact method. Data type: String</td>
</tr>
<tr>
<td>creationDate</td>
<td>The date that the case or incident record was created. Data type: String</td>
</tr>
<tr>
<td>description</td>
<td>The description of the issue from the ticket. Data type: String</td>
</tr>
<tr>
<td>id</td>
<td>The sys_id of the case or incident record that was created. Data type: String</td>
</tr>
<tr>
<td>lastUpdate</td>
<td>The date that the case or incident record was last updated.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>name</td>
<td>The name of the trouble ticket, typically a short description of the issue. Data type: String</td>
</tr>
<tr>
<td>note</td>
<td>A list of the comments and work notes that were added to the ticket by the POST request. Data type: Array</td>
</tr>
<tr>
<td>note.@type</td>
<td>The type of note. Valid values: work_notes, comments                        Data type: String</td>
</tr>
<tr>
<td>note.author</td>
<td>The name of the ServiceNow integration user who made the POST request that created the note. Data type: String</td>
</tr>
<tr>
<td>note.date</td>
<td>The date the note was created in the ServiceNow instance. Data type: String</td>
</tr>
<tr>
<td>note.text</td>
<td>The note text. Data type: String</td>
</tr>
<tr>
<td>relatedParty</td>
<td>Details about the companies and contacts associated with the ticket. Data type: Array</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>relatedParty.id</td>
<td>The sys_id for the related party. For cases, the sys_id is from the Contact [customer_contact] or Account [customer_account] tables. For incidents, the sys_id is from the User [sys_user] or Company [core_company] tables. Data type: String</td>
</tr>
<tr>
<td>relatedParty.referredType</td>
<td>The type of related party. Valid values: • customer: Related party is a company or account. • customer_contact: Related party is a caller or contact. Data type: String</td>
</tr>
<tr>
<td>severity</td>
<td>The severity of the issue described by the trouble ticket. Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>The current status of the trouble ticket. Data type: String</td>
</tr>
<tr>
<td>ticketType</td>
<td>The type of ticket (case or incident). Data type: String</td>
</tr>
</tbody>
</table>

**Example: cURL request**

This example creates a case with high priority and open state.

```bash
curl "https://instance.servicenow.com/api/sn_ind_tsm_sdwan/troubleticket/case" \
   --request POST \   
   --header "Accept:application/json" \   
   --header "Content-Type:application/json" \   
   --data "{
     "description":"Example description of an issue",
   }
```

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
"name": "Short description of issue",
"severity": "2",
"status": "10",
"ticketType": "Case",
"channel": {
    "name": "Email"
},
"note": [
    {
        "text": "This is a testing work note",
        "@type": "work_notes"
    },
    {
        "text": "This is a testing comment",
        "@type": "comments"
    }
],
"relatedParty": [
    {
        "id": "f3af54c413651200042ab3173244b053",
        "@referredType": "customer_contact"
    }
]
User Role Inheritance API

The User Role Inheritance API allows you to see the roles that a specific user inherited.

This API requires the Contextual Security: Role Management V2 REST API (com.glide.role_management.inh_count.rest_api) plugin, automatically activated on new instances starting with the Jakarta release, or the Contextual Security: Role Management Enhancements REST API plugin, automatically activated starting with the Geneva release. The Contextual Security: Role Management V2 REST API is dependent on the Contextual Security: Role Management V2 (com.glide.role_management.inh_count) plugin.

A user’s role can be directly granted, inherited from other roles, or inherited from groups. You must have the user_admin role to access this API.

User Role Inheritance - GET /global/user_role_inheritance

Returns a specified user's granted and inherited roles.

URL format

Versioned URL: /api/global/{api_version}/user_role_inheritance
Default URL: /api/global/user_role_inheritance
Supported request parameters

Path parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api_version</td>
<td>Optional. Version of the endpoint to access. For example, v1 or v2. Only specify this value to use an endpoint version other than the latest. Data type: String</td>
</tr>
</tbody>
</table>

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user_sysid</td>
<td>Required. Sys_id of the user for which to return role information.</td>
</tr>
</tbody>
</table>

Request body parameters (XML or JSON)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Supported types: application/json or application/xml. Default: application/json</td>
</tr>
</tbody>
</table>

Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
Status codes
The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successful. The request was successfully processed.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. A bad request type or malformed request was detected.</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. The user credentials are incorrect or have not been passed.</td>
</tr>
<tr>
<td>404</td>
<td>Not found. The requested item was not found.</td>
</tr>
<tr>
<td>500</td>
<td>Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error.</td>
</tr>
</tbody>
</table>

Response body parameters (JSON or XML)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>from_group</td>
<td>List of roles that are inherited from groups that have roles. Data type: Array</td>
</tr>
<tr>
<td>from_role</td>
<td>List of roles that are either granted directly or inherited from other roles. Data type: Array</td>
</tr>
<tr>
<td>user_name</td>
<td>Full name of user. Data type: String</td>
</tr>
</tbody>
</table>

Example: Sample cURL request

```
curl -H "Accept:application/json" -u "username:'password" "https://instance.service-now.com/api/global/user_role_inheritance?user_sysid=62826bf03710200044e0bfc8bcbbe5df1" 
```
Example: Sample Python request

```python
#easy_install requests
import requests

# Set the request parameters
url = 'https://instance.service-now.com/api/global/user_role_inheritance?user_sysid=62826bf037120044e0bfc8bcbe5df1'

# Eg. User name="username", Password="password" for this code sample.
user = 'username'
pwd = 'password'

# Set proper headers
headers = {'Accept': 'application/json'}

# Do the HTTP request
response = requests.get(url, auth=(user, pwd), headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
    print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:', response.json())
    exit()
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Virtual Agent Bot Integration API

The Virtual Agent Bot Integration API allows users to integrate any chat interface or a bot ecosystem with the ServiceNow Virtual Agent and/or Live Agent. This API is built on the conversational custom chat integration framework provided with Virtual Agent and operates in the `sn_va_as_service` namespace.

For additional information on installing and configuring the Virtual Agent Bot Integration API, see Virtual Agent API.

VA Bot Integration - POST /sn_va_as_service/bot/integration

Sends a specified message, attachment, or URL to a specified secondary ServiceNow Virtual Agent (VA) bot.

A customer's site can have one or more secondary ServiceNow bots. In addition to passing messages to a ServiceNow VA secondary bot, the primary bot or chat client can use this endpoint to pass messages to an agent through Live Agent chat. Use the `action` parameter passed in the request body to specify how the specified content should be handled.
Note: This endpoint can be either synchronous or asynchronous depending on how it is configured. If asynchronous, when the request is received by the endpoint, it validates the request and responds with a call status of success or failure and the appropriate HTTP status code. The VA then processes the request and sends a second response to the configured response endpoint. For example, for a bot-to-bot integration, this would be to the URL of your primary bot. Both of these responses are defined below. For additional information on this configured response endpoint, see Configure the response endpoint for Virtual Agent API.

Note: The responses in the code examples for this endpoint show what is sent from the VA to the configured response endpoint. They do not show the actual response to this endpoint which is a simple success or failure response.

URL format
Default URL: /api/sn_va_as_service/bot/integration

Supported request parameters

<table>
<thead>
<tr>
<th>Path parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Request body parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
</tbody>
</table>
| action | Action that the VA should take. Valid values:  
  • AGENT: Switches the conversation from VA to Live Agent.  
  • END_CONVERSATION: Ends the chat conversation. The |
Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message.text</td>
<td>parameter should be empty when using this action.</td>
</tr>
<tr>
<td>START_CONVERSATION</td>
<td>Starts a chat conversation.</td>
</tr>
<tr>
<td>SWITCH</td>
<td>Switch the conversation to a topic that matches what is specified in the intent.id or topic.name parameters.</td>
</tr>
<tr>
<td>Note:</td>
<td>Only use the SWITCH action for topic switching when the intent discovery happens in the primary bot. If the intent discovery happens in the ServiceNow® VA, use the message.text parameter.</td>
</tr>
<tr>
<td>SET_USER_TIMEZONE</td>
<td>Sets the user’s time zone to the time zone specified in the timezone parameter. This time zone remains in effect until you reset it using this same parameter.</td>
</tr>
<tr>
<td>TYPING/VIEWING</td>
<td>Displays the typing indicator in Live Agent. First send TYPING and then once the user finishes typing, send VIEWING.</td>
</tr>
<tr>
<td>Null or parameter not passed: Sends the message to the ServiceNow VA.</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String
Default: Null
### Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| botToBot         | Flag that indicates whether the conversation is between bots, such as between a primary bot and secondary bot. Possible values:  
  - true: Bot-to-bot conversation.  
  - false: Standalone ServiceNow bot conversation.  
  
  Data type: Boolean  
  Default: true       |
| clientSessionId  | Unique client session identifier of the application/service that the end-user is using to communicate with the ServiceNow bot. This ID helps identify the set of request/responses for a specific user conversation. The format of this parameter is determined by the implementer.  
  Data type: String  
  Default: Null       |
| clientVariables  | Pass-through JSON-formatted name-value pairs that are sent back in the response. These values are not processed by the endpoint, but can be used to keep specific customer information attached to the conversation.  
  Data type: Object  
  Default: Null       |
| contextVariables | JSON-formatted name-value pairs that provide addition                                                                                      |
# Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>information about the user. Provide this information to customize the conversation experience. For additional information on context variables, see Define and publish chat context variables. For example:</td>
</tr>
<tr>
<td></td>
<td>&quot;contextVariables&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;requester_session_language&quot;: &quot;es&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;deviceTimeZone&quot;: &quot;America/Los_Angeles&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;live_agent_only&quot;: &quot;true&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;liveagent_deviceType&quot;: &quot;test_device_type&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;topic&quot;: &quot;7c3c819073a3101066c7e9344ef6a7e4&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>Default: Null</td>
</tr>
<tr>
<td>emailId</td>
<td>Required when enabling account linking, otherwise all users are considered guest users. Email address of the chat client end-user; the person the bot is interacting with. For additional information on account linking, see Link your ServiceNow user account to a messaging application for Virtual Agent conversations.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>history</td>
<td>History of the primary bot’s conversation with the user, starting with the initial request by the user.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Data type</strong>: Object</td>
<td></td>
</tr>
<tr>
<td>&quot;history&quot;: {</td>
<td></td>
</tr>
<tr>
<td>&quot;displayName&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;isBotMessage&quot;: Boolean,</td>
<td></td>
</tr>
<tr>
<td>&quot;type&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;value&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>For example:</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;isBotMessage&quot;: true,</td>
<td></td>
</tr>
<tr>
<td>&quot;value&quot;: &quot;message 1&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;displayName&quot;: &quot;Bot&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;type&quot;: &quot;text&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;isBotMessage&quot;: false,</td>
<td></td>
</tr>
<tr>
<td>&quot;value&quot;: &quot;message 2&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;displayName&quot;: &quot;User&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;type&quot;: &quot;text&quot;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
<tr>
<td><strong>history.displayName</strong></td>
<td>Name of the user to whom the message was sent. If the primary bot, pass Bot.</td>
</tr>
<tr>
<td><strong>Data type</strong>: String</td>
<td></td>
</tr>
<tr>
<td><strong>history.isBotMessage</strong></td>
<td>Flag that indicates whether the message was sent by the primary bot. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Message was sent by primary bot.</td>
</tr>
<tr>
<td></td>
<td>• false: Message was sent by user.</td>
</tr>
<tr>
<td><strong>Data type</strong>: Boolean</td>
<td></td>
</tr>
</tbody>
</table>
## Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>history.type</td>
<td>Type of message sent. Valid values: text</td>
<td>String</td>
</tr>
<tr>
<td>history.value</td>
<td>Content of the information. For example, the message text.</td>
<td>String</td>
</tr>
<tr>
<td>intent</td>
<td>Describes the intent of the conversation. Provide this information when the <code>action</code> parameter is set to <code>SWITCH</code>.</td>
<td>Object</td>
</tr>
<tr>
<td>intent.id</td>
<td>NLU Intent ID of the system topic to switch the conversation to. Located in the CS Topic Language [sys_cs_topic_language] table. The <code>intent.id</code> is resolved based on the language passed in the chat session. If no language is passed when starting a conversation, it obtains the language from the user's session/profile.</td>
<td>String</td>
</tr>
<tr>
<td>message</td>
<td>Required. Details of the message to send to the ServiceNow bot.</td>
<td>Object</td>
</tr>
</tbody>
</table>

```json

{
    "intent": {
        "id": "String"
    }

"message": { 
    "attachment": {Object},
```
### Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;clientMessageId&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;text&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;typed&quot;: Boolean</td>
<td></td>
</tr>
<tr>
<td>message.attachment</td>
<td>Required if sending an attachment. Details of the attachment.</td>
</tr>
<tr>
<td></td>
<td>You can only pass attachments as a URL and only one attachment can be passed per call. A virus scan is triggered on all attachments. There is no restriction on the size of an attachment however, large files can take significant time to transfer.</td>
</tr>
<tr>
<td>Data type: Object</td>
<td></td>
</tr>
<tr>
<td>&quot;attachment&quot;: {</td>
<td></td>
</tr>
<tr>
<td>&quot;clientAttachmentId&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;contentType&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;fileName&quot;: &quot;String&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;headers&quot;:{</td>
<td></td>
</tr>
<tr>
<td>&quot;Authorization&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>},</td>
<td></td>
</tr>
<tr>
<td>&quot;url&quot;: &quot;String&quot;</td>
<td></td>
</tr>
<tr>
<td>message.attachment.clientAttachmentId</td>
<td>Unique identifier of the attachment to send to the ServiceNow bot. The format of this identifier is determined by the implementer.</td>
</tr>
<tr>
<td>Data type: String</td>
<td></td>
</tr>
<tr>
<td>Default: Null</td>
<td></td>
</tr>
<tr>
<td>message.attachment.contentType</td>
<td>Required if attachment specified. Standard mime type of the attachment.</td>
</tr>
</tbody>
</table>
### Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message.attachment.fileName</td>
<td>Required if attachment specified. File name of the attachment. Data type: String</td>
</tr>
<tr>
<td>message.attachment.headers</td>
<td>Additional information needed if the attachment file is private and protected. Data type: Object</td>
</tr>
<tr>
<td>message.attachment.headers.Authorization</td>
<td>Information needed to obtain authorization to access the attachment, such as a user name and password. Data type: String</td>
</tr>
<tr>
<td>message.attachment.url</td>
<td>Required if attachment specified. URL where the attachment is located. The VA obtains the attachment from this URL, runs a virus scan, and then processes the file. If the attachment is not a public file, you must pass an <code>message.attachment.header</code></td>
</tr>
</tbody>
</table>

Valid values (not case-sensitive):
- application/pdf
- application/msword
- image/png
- image/jpeg
- text/plain
- video/mpeg

Data type: String
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>object that contains the authorization information so that the attachment can be accessed. You must also ensure that URL domain is present in the trusted domains filed in <code>sys_cs_provider</code>. Data type: String</td>
</tr>
<tr>
<td>message.clientMessageId</td>
<td>Unique alphanumeric identifier for the message. The format of this identifier is determined by the implementer. Data type: String Default: Null</td>
</tr>
<tr>
<td>message.text</td>
<td>Required if attachment parameter is not specified. Message text. The language of the message should be passed in the contextVariables parameter. Data type: String Maximum length: Unlimited</td>
</tr>
</tbody>
</table>
| message.typed               | Required. Flag that indicates whether the message was typed in or was an option selection (clicked). Valid values:  

- true: Message was typed in by the user.  
- false: Message was an option selection.  

Data type: Boolean                                                                                                                                         |
| requestId                   | Required. Unique alphanumeric identifier for this request.                                                                                                                                                                                                                                                                                   |
Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| silentMessage  | Flag that indicates whether this is a silent message. A silent message is a message that does not require a response. Valid values:  
  • true: Silent message, no response message expected.  
  • false: Regular message, response message expected.  

  Note: If a request with `silentMessage = true` is sent, all subsequent bot messages remain suppressed until a request to turn-off silent mode is sent by passing `silentMessage = false`.  

  Data type: Boolean  
  Default: false  

| timestamp      | Unix epoch time when the message was sent from the end-user of the chat client.  

  Data type: Number  
  Unit: Milliseconds  
  Default: Current time  

| timezone       | GMT time zone of the end-user of the chat client.  

  For example:  

  "timezone":"Asia/Kolkata"
### Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>For a list of the available time zone values, see <a href="https://en.wikipedia.org/wiki/List_of_tz_database_time_zones">https://en.wikipedia.org/wiki/List_of_tz_database_time_zones</a>. Data type: String Default: GMT time zone of the ServiceNow instance.</td>
</tr>
</tbody>
</table>

**topic**

Required if the user wants to switch to a particular topic by specifying a topic name. Name-value pair of the topic to switch to. You can either specify the topic name or the topic sys_id.

#### Note:

You can only specify a single topic name or id in the object.

Data type: Object

```json
"topic": {
  "name": "String",
  "id": "String"
}
```

For example:

```json
{
  "requestId": "xxxx-xxxx-xxxx-xxxx",
  "clientSessionId": "xxx-xxx-xxx-xxx",
  "action": "SWITCH",
  "topic": {
    "name": "Topic Name"
  },
  "userId": "beth"
}
```
Request body parameters (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>topic.name</td>
<td>Name of the topic to switch to. Located in the name field of the Virtual Agent topics [sys_cs_topic] table. Data type: String</td>
</tr>
<tr>
<td>topic.id</td>
<td>Sys_id of the topic to switch to. Located in the sys_id field of the Virtual Agent topics [sys_cs_topic] table. Data type: String</td>
</tr>
<tr>
<td>userId</td>
<td>Required. Unique user identifier. This is the identifier of the end-user who is interacting with the bot. Data type: String</td>
</tr>
</tbody>
</table>

Headers

The following request and response headers apply to this HTTP action only, or apply to this action in a distinct way. For a list of general headers used in the REST API, see Supported REST API headers.

Request headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Data format of the response body. Only supports application/json.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Data format of the request body. Only supports application/json.</td>
</tr>
<tr>
<td>token</td>
<td>Required if using token based authentication; optional for Basic or OAuth authentication. Authentication token to use if token authentication has been set up for this endpoint. Data type: String</td>
</tr>
</tbody>
</table>
Response headers

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**Status codes**

The following status codes apply to this HTTP action. For a list of possible status codes used in the REST API, see REST API HTTP response codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
</table>
| 200         | Successful. The request was successfully processed. If you receive a 200 status code, but do not receive a response back, check the following things:  
• Whether the request is failing in Static, JWT, or Hash Based Token validation which generally happens in the worker thread of Virtual Agent.  
• Whether the request body format is correct. If incorrect, the conversion does not happen correctly.  
• Whether the request body is missing some mandatory parameters. Exceptions and errors can be traced in logs.  
• Whether the site defined in the REST Message to receive the response is operational; able to detect in logs.  
• Ensure the REST Message that defines where to post the response is set correctly. |
| 400         | Bad Request. A bad request type or malformed request was detected. |
| 401         | Unauthorized. The user credentials are incorrect or have not been passed. |
| 405         | Invalid method. The functionality is disabled. |
| 500         | Internal server error. An unexpected error occurred while processing the request. The response contains additional information about the error. |

**Response body parameters**

Status of the endpoint call. This is sent in response when the initial call request is received. The request has not yet been processed.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>error</td>
<td>Description of the detected error. Only returned if an error occurs.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;error&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;detail&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>error.detail</td>
<td>Details about the encountered error.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>error.message</td>
<td>Error message thrown.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>status</td>
<td>Status of the endpoint call.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• failure</td>
</tr>
<tr>
<td></td>
<td>• success</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**VA response body parameters**

Response body sent from the VA to the configured response endpoint.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentChat</td>
<td>Flag that indicates whether the conversation is connected to the ServiceNow Live Agent.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Chat messages are coming from a live agent.</td>
</tr>
<tr>
<td></td>
<td>• false: Chat messages are coming from a bot and this parameter is not returned in the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>body</td>
<td>List of objects that describe the elements of the response message body.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;body&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;actionType&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;agentInfo&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;data&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;group&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;header&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;height&quot;: Number,</td>
</tr>
<tr>
<td></td>
<td>&quot;itemType&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;label&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;maskType&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;multiSelect&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;nluTextEnabled&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;options&quot;: {Object},</td>
</tr>
<tr>
<td></td>
<td>&quot;promptMsg&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;required&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;style&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;templateName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;type&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;uiType&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String/{Object}&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;width&quot;: Number</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>body.actionType</td>
<td>Type of action message dispatched by the secondary bot to the primary bot. For example, typing indicator or agent wait time. Valid values:</td>
</tr>
<tr>
<td></td>
<td>• EndTypingIndicator</td>
</tr>
<tr>
<td></td>
<td>• StartSpinner</td>
</tr>
<tr>
<td></td>
<td>• StartTypingIndicator</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
</tbody>
</table>

For example:
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Sending</strong> StartTypeIndicator <strong>to the primary bot</strong> when the typing indicator is enabled.</td>
</tr>
<tr>
<td></td>
<td>```json</td>
</tr>
<tr>
<td></td>
<td>&quot;body&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;uiType&quot;: &quot;ActionMsg&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;actionType&quot;: &quot;StartTypingIndicator&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td><strong>Sending</strong> StartSpinner <strong>when transferring to Live Agent</strong> to display the wait time to the end user.</td>
</tr>
<tr>
<td></td>
<td>```json</td>
</tr>
<tr>
<td></td>
<td>&quot;body&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;uiType&quot;: &quot;ActionMsg&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;actionType&quot;: &quot;StartSpinner&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;spinnerType&quot;: &quot;wait_time&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: &quot;Routing you to a live agent...&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;waitTime&quot;: &quot;8 Seconds&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>body.agentInfo</td>
<td><strong>If enabled in the agent chat set up, the name and avatar of the associated agent.</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Data type: Object</strong></td>
</tr>
<tr>
<td></td>
<td>```json</td>
</tr>
<tr>
<td></td>
<td>&quot;agentInfo&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;agentAvatar&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;agentName&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;sentFromAgent&quot;: Boolean</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td><strong>For example:</strong></td>
</tr>
<tr>
<td></td>
<td>```json</td>
</tr>
<tr>
<td></td>
<td>&quot;agentInfo&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;sentFromAgent&quot;: true,</td>
</tr>
<tr>
<td></td>
<td>&quot;agentName&quot;: &quot;Beth Anglin&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;agentAvatar&quot;: &quot;<a href="https://instance.servicenow.com/ee4eebf30a0004d963b5c5ac0d734dc4.iix?t=small">https://instance.servicenow.com/ee4eebf30a0004d963b5c5ac0d734dc4.iix?t=small</a>&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>body.agentInfo.agentAvatar</td>
<td><strong>URL of the agent’s associated avatar file.</strong></td>
</tr>
<tr>
<td></td>
<td><strong>For example:</strong> &quot;<a href="https://instance.servicenow.com/">https://instance.servicenow.com/</a>&quot;</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>ee4eef30a0004d963b5c5ac0d734dc4.iix?t=small&quot;</code></td>
<td>Data type: String</td>
</tr>
<tr>
<td><code>body.agentInfo.agentName</code></td>
<td>Display name of the agent. Data type: String</td>
</tr>
<tr>
<td><code>body.agentInfo.sentFromAgent</code></td>
<td>Flag that indicates whether the message was sent by a live agent. Possible values: • true: Sent by a live agent. • false: Sent by a virtual agent. Data type: Boolean</td>
</tr>
<tr>
<td><code>body.data</code></td>
<td>JSON string that defines the data within the card. Data type: String</td>
</tr>
<tr>
<td><code>body.group</code></td>
<td>Rich control group to which the body element belongs. You can further transform these default controls by writing outbound transformation scripts. For additional information on these scripts, see Virtual Agent action scripts. Possible values: • DefaultOutputCard • DefaultDate • DefaultGroupedPartsOutputControl • DefaultHtml • DefaultOutputImage • DefaultOutputLink • DefaultText • DefaultPicker Data type: String</td>
</tr>
<tr>
<td><code>body.header</code></td>
<td>Link header.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>body.height</td>
<td>Height of the HTML element. Data type: Number Unit: Pixels</td>
</tr>
</tbody>
</table>
| body.itemType     | Type of body element. Possible values: (case-sensitive)  
|                   | • DefaultText  
|                   |   ◦ File  
|                   |   ◦ Image  
|                   |   ◦ Inputtext  
|                   |   ◦ outputtext  
|                   | • DefaultPicker  
|                   |   ◦ Picture: Items appear in a carousel.  
|                   |   ◦ List: Items appear in a dropdown-style list. Data type: String |
| body.label        | Label to display for the associated element. Data type: String |
| body.maskType     | Indicates the sensitivity of the associated data. Possible values:  
|                   | • NONE: Associated data is of a general nature and does not need to be secured.  
<p>|                   | • SECURE: Associated data is of a sensitive nature and should be handled securely. Data type: String |
| body.message      | Only returned when <strong>actionType</strong> is <strong>StartSpinner</strong>. Message to display to user. Data type: String |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body.multiSelect</td>
<td>Flag that indicates whether the user can select multiple options.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Multiple selections.</td>
</tr>
<tr>
<td></td>
<td>• false: Single selection.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>body.nluTextEnabled</td>
<td>Flag that indicates whether the user chat client can receive NLU text messages.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Can receive NLU messages.</td>
</tr>
<tr>
<td></td>
<td>• false: Cannot receive NLU messages.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>body.options</td>
<td>List of objects that describe the options available for a picker control.</td>
</tr>
<tr>
<td></td>
<td>Data type: Array</td>
</tr>
<tr>
<td></td>
<td>&quot;options&quot;: [</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td>&quot;attachment&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;description&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;enabled&quot;: Boolean,</td>
</tr>
<tr>
<td></td>
<td>&quot;label&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;renderStyle&quot;: &quot;String&quot;,</td>
</tr>
<tr>
<td></td>
<td>&quot;value&quot;: &quot;String&quot;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>]</td>
</tr>
<tr>
<td>body.options.attachment</td>
<td>URL of the image in the carousel to retrieve.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>body.options.description</td>
<td>Description of the carousel item.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>
| body.options.enabled          | Flag that indicates whether the picker control is enabled on the end-user's chat client. In bot-to-bot implementations, the
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>primary bot is responsible for ensuring the picker control is disabled. Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Enabled</td>
</tr>
<tr>
<td></td>
<td>• false: Disabled</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>body.options.label</td>
<td>Label to display for the associated body.options.value parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>body.options.renderStyle</td>
<td>Style to use when rendering the associated element.</td>
</tr>
<tr>
<td></td>
<td>Possible value: data</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>body.options.value</td>
<td>Value to display for the associated body.options.label parameter.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>body.promptMsg</td>
<td>Prompt message to display with the picker control.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>body.required</td>
<td>Flag that indicates whether the end-user must respond to the query in the body element.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Required</td>
</tr>
<tr>
<td></td>
<td>• false: Optional</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>body.spinnerType</td>
<td>Only returned when <strong>actionType</strong> is StartSpinner. Type of spinner shown to chat user.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• wait_time: Message indicating the amount of delay before being transferred to a live agent.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>body.style</td>
<td>Style of body element. Possible values: button, carousel, null.</td>
</tr>
<tr>
<td>body.templateName</td>
<td>Name of the template to use to display the associated card.</td>
</tr>
<tr>
<td>body.type</td>
<td>Only returned for DefaultOutputLink. Type of return data. Possible values: link.</td>
</tr>
<tr>
<td>body.uiType</td>
<td>Type of body element. The possible values depend on the value in the body.group parameter. These values map back to the controls defined in the Virtual Agent Designer. For more information, see Virtual Agent Designer. Possible values: ActionMsg, DefaultOutputCard, DefaultDate, DefaultGroupedPartsOutputControl, DefaultHtml.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>◦ OutputHtml</td>
<td></td>
</tr>
<tr>
<td>◦ DefaultOutputImage</td>
<td></td>
</tr>
<tr>
<td>◦ OutputImage</td>
<td></td>
</tr>
<tr>
<td>◦ DefaultOutputLink</td>
<td></td>
</tr>
<tr>
<td>◦ OutputImage</td>
<td></td>
</tr>
<tr>
<td>◦ DefaultText</td>
<td></td>
</tr>
<tr>
<td>◦ InputText</td>
<td></td>
</tr>
<tr>
<td>◦ OutputText</td>
<td></td>
</tr>
<tr>
<td>◦ FileUpload</td>
<td></td>
</tr>
<tr>
<td>◦ DefaultPicker</td>
<td></td>
</tr>
<tr>
<td>◦ Picker</td>
<td></td>
</tr>
<tr>
<td>◦ TopicPickerControl</td>
<td></td>
</tr>
<tr>
<td>◦ Boolean</td>
<td></td>
</tr>
</tbody>
</table>

Data type: String

**body.value**

Data to use to process the body element. The type of information that is passed back depends on the type of body element.

OutputLink: Object.

```
"value": {
  "action": "String"
}
```

OutputImage: String. Image URL.

OutputHtml: String. HTML that the client needs to render.

Data type: String or Object

**body.value.action**

Only returned for DefaultOutputLink. URL of returned link.

Data type: String

**body.waitTime**

Amount of time that the user has to wait before being connected to a live agent.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>body.width</td>
<td>Width of the associated HTML element.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td></td>
<td>Unit: Pixels</td>
</tr>
<tr>
<td>clientSessionId</td>
<td>Unique client session identifier of the chat application/service that the end-user is using to communicate with the ServiceNow bot.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>clientVariables</td>
<td>JSON-formatted name-value pairs that are simply copied from what is passed in the clientVariables parameter in the request body. These values are not processed by the endpoint, but can be used to keep specific customer information attached to the conversation.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td>completed</td>
<td>Flag that indicates whether the ServiceNow Virtual Agent has successfully completed the conversation.</td>
</tr>
<tr>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td>• true: Conversation has been successfully completed.</td>
</tr>
<tr>
<td></td>
<td>• false: Conversation is not complete and this parameter is not returned in the response.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>enterpriseId</td>
<td>Currently unused.</td>
</tr>
<tr>
<td>message</td>
<td>Details of the message sent to the ServiceNow VA. This is exactly the same as the passed in message object.</td>
</tr>
<tr>
<td></td>
<td>Data type: Object</td>
</tr>
<tr>
<td></td>
<td>&quot;message&quot;: {</td>
</tr>
<tr>
<td></td>
<td>&quot;attachment&quot;: {Object},</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>message.attachment</td>
<td>Details of the attachment. Attachments can be either a file or a URL. Data type: Object</td>
</tr>
<tr>
<td>message.attachment.clientAttachmentId</td>
<td>Unique identifier of the attachment sent to the ServiceNow VA. The format of this identifier is determined by the implementer. Data type: String</td>
</tr>
<tr>
<td>message.attachment.contentType</td>
<td>Standard mime type of the attachment. Possible values (not case-sensitive):</td>
</tr>
<tr>
<td></td>
<td>• text/plain</td>
</tr>
<tr>
<td></td>
<td>• application/pdf</td>
</tr>
<tr>
<td></td>
<td>• application/msword</td>
</tr>
<tr>
<td></td>
<td>• image/png</td>
</tr>
<tr>
<td></td>
<td>• image/jpeg</td>
</tr>
<tr>
<td></td>
<td>• video/mpeg</td>
</tr>
<tr>
<td>message.attachment.fileName</td>
<td>File name of the attachment. Data type: String</td>
</tr>
<tr>
<td>message.attachment.url</td>
<td>URL where the attachment is located. Data type: String</td>
</tr>
<tr>
<td>message.clientMessageId</td>
<td>Unique alphanumeric identifier for the message.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>message.text</td>
<td>Message text.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td></td>
<td>Maximum length: Unlimited</td>
</tr>
<tr>
<td>message.typed</td>
<td>Flag that indicates whether the message was typed in or was an option selection (clicked). Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Message was typed in by the user.</td>
</tr>
<tr>
<td></td>
<td>• false: Message was an option selection.</td>
</tr>
<tr>
<td></td>
<td>Data type: Boolean</td>
</tr>
<tr>
<td>nowBotId</td>
<td>Currently unused.</td>
</tr>
<tr>
<td>nowSessionId</td>
<td>Currently unused.</td>
</tr>
<tr>
<td>requestId</td>
<td>Unique identifier for this request.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
<tr>
<td>score</td>
<td>Intent confidence score from the ServiceNow NLU model, specified as a percent, such as 87.3%. This score enables the primary bot to make the decision as to whether to surface the associated intent based on the primary bot’s NLU model’s confidence threshold. The ServiceNow NLU model’s confidence threshold is ignored on the ServiceNow instance so that the primary bot’s NLU model’s confidence threshold can be consistently applied across all secondary bots.</td>
</tr>
<tr>
<td></td>
<td>Data type: Number</td>
</tr>
<tr>
<td>takeControl</td>
<td>Flag that indicates that the ServiceNow Virtual Agent wants to hand control back over to the primary bot.</td>
</tr>
<tr>
<td></td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>• true: Return control to the primary bot.</td>
</tr>
<tr>
<td></td>
<td>• false: Keep control.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>userId</td>
<td>Identifier of the end-user who is interacting with the bot.</td>
</tr>
<tr>
<td></td>
<td>Data type: String</td>
</tr>
</tbody>
</table>

**Example:** Start a conversation using the **START_CONVERSATION** action to start a conversation.

The following example shows how to use the **START_CONVERSATION** action to start a conversation with Virtual Agent. This call directs the user to the greeting topic.

```bash
curl "https://instance.servicenow.com/api/sn_va_as_service/bot/integration" \
--request POST \
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--data "{
  "requestId": "asd2423-sda23-qwe23-we23",
  "action": "START_CONVERSATION",
  "enterpriseId": "ServiceNow",
  "nowBotId": "A85PWLERF",
  "clientSessionId": ",",
  "nowSessionId": ",",
  "message":{
    "text": ",",
    "typed": true,
    "clientMessageId": "ABC-123"
  },
  "userId": "beth.anglin",
  "emailId": "beth.anglin@example.com",
  "timestamp": 1588824102,
  "timezone": "America/New_York"
}"
--user "username":"password"
```

The following is the response body sent from the VA to the configured response endpoint.

```json
{
  "requestId": "asd2423-sda23-qwe23-we23",
  "clientSessionId": ",",
  "nowSessionId": ",",
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: Transfer a conversation to Live Agent using utterance

The following example illustrates how to transfer a conversation to Live Agent, bypassing the greeting topic, by passing agent as an utterance in the message.text parameter. Possible utterance values are defined in the sys_cs_contextual_action table. For additional information on contextual actions, see Contextual actions for custom chat integrations.
The following is the response body sent from the VA to the configured response endpoint.

```json
{
  "requestId": "322bas2be70-sadsa-we32-3eq2-1231ra9",
  "clientSessionId": "",
  "message": {
    "text": "agent",
    "typed": true
  },
  "userId": "abel.tuter",
  "emailId": "abel.tuter@servicenow.com",
  "timestamp": 1588824102,
  "timezone": "America/New_York"
}
```
Example: Transfer a conversation to Live Agent Support based on the user's selection

The following example shows transferring the conversation to Live Agent Support based on the user's selection from the prior interaction.

curl "https://instance.servicenow.com/api/sn_va_as_service/bot/integration" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --header "token: ORANGE" \
  --data "{
    "requestId": "322bas2be70-sadsa-we32-3eq2-1231ra9",
    "botToBot": true,
    "clientSessionId": "",
    "silentMessage": false,
    "message": {
      "text": "Live Agent Support.",
      "typed": true
    },
    "userId": "abel.tuter",
    "score": 1
  }"
The following is the response body sent from the VA to the configured response endpoint.

```
{
  "requestId": "322bas2be70-sadsa-we32-3eq2-1231ra9",
  "clientSessionId": "",
  "message": {
    "text": "Live Agent Support.",
    "typed": true
  },
  "userId": "abel.tuter",
  "body": [
    {
      "uiType": "OutputText",
      "group": "DefaultText",
      "value": "Please stand by while I connect you to a live agent."
    }
  ],
  "agentChat": true,
  "score": 1
}
```

**Example: End a Virtual Agent conversation by setting the action parameter.**

The following example shows how to end a Virtual Agent conversation by setting the `action` parameter to `END_CONVERSATION`.

```
curl "https://instance.servicenow.com/api/sn_va_as_service/bot/integration" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --data "{
  "requestId": "s345ew-sjg554-qwe23-we23",
  "action": "END_CONVERSATION",
  "enterpriseId": "ServiceNow",
  "nowBotId": "A85PWLERF",
  "clientSessionId": "",
  "nowSessionId": "",
  "message": {
  ```
The following is the response body sent from the VA to the configured response endpoint showing the conversation ended.

```json
{
    "requestId": "s345ew-sjg554-qwe23-we23",
    "clientSessionId": "",
    "nowSessionId": "",
    "message": {
        "text": "",
        "typed": true,
        "clientMessageId": "ABC-123"
    },
    "userId": "beth.anglin",
    "body": [
        {
            "uiType": "OutputText",
            "group": "DefaultText",
            "value": "The conversation has ended. If you need help again, type hi."
        }
    ],
    "score": 1
}
```

Example: End a Live Agent conversation by setting the action parameter

The following example shows how to end a Live Agent conversation by setting the `action` parameter to `END_CONVERSATION`.

```
curl "https://instance.servicenow.com/api/sn_va_as_service/bot/integration" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The following is the response body sent from the VA to the configured response endpoint showing the conversation ended.

```json
{
  "requestId": "sad-sjg554-qwe23-we23",
  "clientSessionId": "",
  "nowSessionId": "",
  "message": {
    "text": "",
    "typed": true,
    "clientMessageId": "ABC-123"
  },
  "userId": "beth.anglin",
  "body": {
    {
      "uiType": "OutputText",
      "group": "DefaultText",
      "value": "The conversation has ended. If you need help again, type hi."
    }
  },
  "completed": true,
  "score": 1
}
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: Use commands defined in the sys_cs_contextual_action table as an utterance to start a conversation with Virtual Agent

The following example shows how to use the **hi** command defined in the sys_cs_contextual_action table as an utterance to start a conversation with Virtual Agent instead of setting the action parameter to START_CONVERSATION as shown in prior examples. This example shows using one of the language variants of the **hi** command, "hola".

```bash
curl "https://instance.servicenow.com/api/sn_va_as_service/bot/integration" \
  --request POST \ 
  --header "Accept:application/json" \ 
  --header "Content-Type:application/json" \ 
  --header "token: ORANGE" \ 
  --data "{
    "requestId": "302bas2be70-sadsa-we32-3eq2-1231ra9",
    "botToBot": true,
    "clientSessionId": "",
    "silentMessage": false,
    "message": {
      "text": "hola",
      "typed": true
    },
    "userId": "fred.ludy",
    "emailId": "abel.tuter@servicenow.com",
    "timestamp": 1588824102,
    "timezone": "America/New_York"
  }"
```

The following is the response body sent from the VA to the configured response endpoint.

```json
{
    "requestId": "302bas2be70-sadsa-we32-3eq2-1231ra9",
    "clientSessionId": "",
    "message": {
      "text": "hola",
      "typed": true
    },
    "userId": "fred.ludy",
    "body": {
      "uiType": "OutputText",
      "group": "DefaultText",
      "value": "Hi, I'm your Virtual Agent. Let me know how I can help you today.",
      "maskType": "NONE"
    }
}
```
uiType":"TopicPickerControl",
"group":"DefaultPicker",
"nluTextEnabled":false,
"promptMsg":"Hi guest, please enter your request or make a selection of what I can help with. You can type helllllp! any time when you need help.",
"label":"Show me everything",
"options":[
{
"label":"boolean check",
"value":"1b5257a1db712010d3d6d9595e9619f3",
"enabled":true
},
{
"label":"Test_table_datepicker",
"value":"6515f0cadb4220108249f7541d961980",
"enabled":true
},
{
"label":"Test File upload",
"value":"76c1f4b2db7520108249f7541d9619a7",
"enabled":true
},
{
"label":"Test B2B 2",
"value":"8373e55cdb8120108249f7541d96196d",
"enabled":true
},
{
"label":"FAQ Conversation Builder_Global",
"value":"9f3114a3dbc520108249f7541d961981",
"enabled":true
},
{
"label":"Get to know you",
"value":"a443dd6adb812010d3d6d9595e961954",
"enabled":true
},
{
"label":"Get to know you_Global",
"value":"b1a56b87db8120108249f7541d9619b2",
"enabled":true
}];
Example: Use commands defined in the sys_cs_contextual_action table as an utterance to end a conversation

The following example shows how to use the `Bye` command defined in the sys_cs_contextual_action table as an utterance to end a conversation with Virtual Agent instead of setting `action` to `END_CONVERSATION` as shown in prior examples. This example shows using one of the language variants of the `Bye` command, "choa".

```
curl "https://instance.servicenow.com/api/sn_va_as_service/bot/integration" \
  --request POST \
  --header "Accept:application/json" \
  --header "Content-Type:application/json" \
  --header "token: ORANGE" \
  --data "{
    "requestId": "302bas2be70-sadsa-we32-3eq2-1231ra9",
    "botToBot": true,
    "clientSessionId": "",
    "silentMessage": false,
    "message": {
      "text": "choa",
      "typed": true
    },
    "userId": "fred.ludy",
    "emailId": "abel.tuter@servicenow.com",
    "timestamp": 1588824102,
    "timezone": "America/New_York"
  }"
```
The following is the response body sent from the VA to the configured response endpoint.

```json
{
    "requestId":"302bas2be70-sadsa-we32-3eq2-1231ra9",
    "clientSessionId":"",
    "message":{
        "text":"chao",
        "typed":true
    },
    "userId":"fred.ludy",
    "body":{
        "uiType":"Picker",
        "group":"DefaultPicker",
        "required":true,
        "nluTextEnabled":false,
        "label":"Please make a selection.",
        "itemType":"ID",
        "style":"list",
        "multiSelect":false,
        "options":{
            "label":"End conversation",
            "value":"1",
            "renderStyle":"data",
            "enabled":false
        },
        "scriptedData":null
    },
    "score":1
}
```
Example: Change an ongoing conversation using the SWITCH action parameter.

The following example shows how to use the **SWITCH** action parameter to change an ongoing conversation to the topic that matches what is specified in the **intent.id** parameter.

```bash
curl "https://instance.servicenow.com/api/sn_va_as_service/bot/integration" \
--request POST \
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--data "{
  "requestId": "adfsfs-hgfhg-34fssd-zxcz-sd43",
  "enterpriseId": "ServiceNow",
  "nowBotId": "A85PWLERF",
  "clientSessionId": "",
  "nowSessionId": "",
  "action": "SWITCH",
  "intent": {
    "id": "Saloon Vehicle"
  },
  "message":{
    "typed": true,
    "clientMessageId": "ABC-123"
  },
  "userId": "Guest",
  "timestamp": 1588824102,
  "timezone": "America/New_York"
}"
--user "username":"password"
```

The following is the response body sent from the VA to the configured response endpoint showing the VA switched to the new topic.

```json
{
  "requestId": "adfsfs-hgfhg-34fssd-zxcz-sd43",
  "clientSessionId": "",
  "nowSessionId": "",
  "message":{
    "typed": true,
    "clientMessageId": "ABC-123"
  },
  "userId": "Guest",
  "body":{
    "uiType": "InputText",
    "body": "Please enter your vehicle details.
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: Obtain a Natural Language Understanding (NLU) confidence score for an utterance

The following example shows how to obtain a NLU confidence score for an utterance if NLU is enable in VA.

```
curl "https://instance.servicenow.com/api/sn_va_as_service/bot/integration" \
   --request POST \
   --header "Accept:application/json" \
   --header "Content-Type:application/json" \
   --data "{
   "requestId": "sdfsfs-hgfhg-34fssd-zxcz-sd43",
   "enterpriseId": "ServiceNow",
   "nowBotId": "A85PWLERF",
   "clientSessionId": "",
   "nowSessionId": "",
   "message":{
      "text": "most popular choice",
      "typed": true,
      "clientMessageId": "ABC-123"
   },
   "userId": "Guest",
   "timestamp": 1588824102,
   "timezone": "America/New_York"
}"
   --user "username":"password"
```

The following is the response body sent from the VA to the configured response endpoint showing the NLU confidence score.

```
{
   "requestId":"sdfsfs-hgfhg-34fssd-zxcz-sd43",
   "clientSessionId":"",
   "nowSessionId":"",
   "message":{
```
Example: Transfer a conversation to Live Agent in a specific queue.

The following example shows how to transfer a conversation to Live Agent in a specific queue. You must first set up your queues with specific conditions. You then pass those conditions as contextVariables in the request body. For additional information on creating a work item queue, see Create a work item queue.

curl "https://instance.servicenow.com/api/sn_va_as_service/bot/integration" \ 
--request POST \ 
--header "Accept:application/json" \ 
--header "Content-Type:application/json" \ 
--data "{
  "requestId": "34re-dasd-qwe23-dsds", // Unique value for a message (hash)
  "enterpriseId": "ServiceNow", // ServiceNow (Optional)
  "nowBotId": "A85PWLERF", // In case of multiple ServiceNow secondary bots (one for HR, one for ITSM)
  "action": "AGENT", // Actions to close conversation. Other possible value is END_CONVERSATION
  "clientSessionId": ", // Client conversation id
  "nowSessionId": ", // ServiceNow conversation id
  "message": {
    "text": "How are you?", // User passed message
    "typed": true, // typed or clicked
    "clientMessageId": "ABC-123"
  },
  "userId": "abraham.lincoln", // User’s id in their system
}
The following is the response body sent from the VA to the configured response endpoint.

```json
{
  "requestId": "34re-dasd-qwe23-dsds",
  "clientSessionId": "",
  "nowSessionId": "",
  "message": {
    "text": "How are you?",
    "typed": true,
    "clientMessageId": "ABC-123"
  },
  "userId": "abraham.lincoln",
  "body": [
    {
      "uiType": "OutputText",
      "group": "DefaultText",
      "value": "I am sorry but I didn't understand your request.",
      "maskType": "NONE"
    },
    {
      "uiType": "OutputText",
      "group": "DefaultText",
      "value": "Please try giving me your request in a different way. I'm currently better at understanding short sentences.",
      "maskType": "NONE"
    },
    {
      "uiType": "TopicPickerControl",
      "group": "DefaultPicker",
      "nluTextEnabled": false,
      "promptMsg": "Hi guest, please enter your request or make a selection of what I can help with. You can type help any time when you need help.",
      "label": "Show me everything",
      "options": [
```
Example: Transfer a public application/pdf attachment file

This example shows how to transfer a public application/pdf attachment.

curl "https://instance.servicenow.com/api/sn_va_as_service/bot/integration" \
   --request POST \
   --header "Accept:application/json" \
   --header "Content-Type:application/json" \
   --header "token:BOT_Token" \
   --data "{
   "requestId": "f42f3550-5b44-4cde-aa52-9b6756b3131c",
   "clientSessionId": "U94CSJLEN",
   "message": {
     "attachment": {
       "clientAttachmentId": "my-unique-identifier",
       "contentType": "application/pdf",
       "fileName": "MY PDF TEST",
       "url": "http://www.africau.edu/images/default/sample.pdf"
     }
   },
   "userId": "U94CSJLEN",
   "emailId": "admin@example.com"
}"
   --user "username":"password"

The following is the response body sent from the VA to the configured response endpoint.

{  
   "requestId": "f42f3550-5b44-4cde-aa52-9b6756b3131c",
   "clientSessionId": "U94CSJLEN",
   "message": {
     "attachment": {
       "clientAttachmentId": "my-unique-identifier",
   
© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Example: Transfer a video/mpeg attachment

This example shows the request/response for uploading a video/mpeg attachment.

curl "https://instance.servicenow.com/api/sn_va_as_service/bot/integration" \
--request POST \ 
--header "Accept:application/json" \
--header "Content-Type:application/json" \
--header "token: BOT_TOKEN" \
--data "{
  "requestId": "f42f3550-5b44-4cde-aa52-9b6756b3131c",
  "clientSessionId": "U94CSJLEN",
  "message": {
    "attachment": {
      "clientAttachmentId": "my-unique-identifier10",
    
```
The following is the response body sent from the VA to the configured response endpoint.

```json
{
    "requestId":"f42f3550-5b44-4cde-aa52-9b6756b3131c",
    "clientSessionId":"U94CSJLEN",
    "message":{
        "attachment":{
            "headers":{
                "Authorization":"Basic amFzbWluZS53YWRod2FuaWE6VGVzdGluz0AxMjM="
            },
            "clientAttachmentId":"my-unique-identifier10",
            "contentType":"video/mpeg",
            "fileName":"MY TEST10.mp4",
            "url":"https://sample-videos.com/video123/mp4/720/big_buck_bunny_720p_1mb.mp4"
        }
    },
    "userId":"U94CSJLEN",
    "body":[
        {
            "uiType":"OutputText",
            "group":"DefaultText",
            "required":false,
            "nluTextEnabled":false,
            "label":"File got",
            "maskType":"NONE",
            "itemType":"outputtext"
        },
        {
            "uiType":"OutputText",
            "group":"DefaultText",
            "required":false,
            "nluTextEnabled":false,
            "label":"The conversation has ended. If you need help again, type hi."
        }
    ]
}
```
Example: Transfer a image/jpeg attachment

This example shows the request/response for uploading a image/jpeg attachment.

curl "https://instance.servicenow.com/api/sn_va_as_service/bot/integration" \
   --request POST \
   --header "Accept:application/json" \
   --header "Content-Type:application/json" \
   --header "token: BOT_TOKEN" \
   --data "{
   "requestId": "f42f3550-5b44-4cde-aa52-9b6756b3131c",
   "clientSessionId": "U94CSJLEN",
   "message": {
   "attachment": {
   "clientAttachmentId": "my-unique-identifier10",
   "contentType": "image/jpeg",
   "fileName": "MY TEST10.jpg",
   "url": "https://image.imagstock.com/image-photo/pink-butterfly-isolated-on-white-260nw-121810407.jpg"
   }
   },
   "userId": "U94CSJLEN",
   "emailId": "admin@example.com"
   }" \
   --user "username":"password"

The following is the response body sent from the VA to the configured response endpoint.

{ 
   "requestId": "f42f3550-5b44-4cde-aa52-9b6756b3131c",
   "clientSessionId": "U94CSJLEN",
   "message": {
   "attachment": {},
   "headers": {
   "Authorization": "Basic amFzbWluZS55YS9WbGRod2FuaW9VZzdGluZ0AxMkJM="
   } 
   }
Scripted REST APIs

The scripted REST API feature allows application developers to build custom web service APIs.

You can define service endpoints, query parameters, and headers for a scripted REST API, as well as scripts to manage the request and response.

Scripted REST APIs generally follow the REST architecture, but you can customize them to use different conventions. You define scripted REST APIs using the
Scripted REST Service form found under **Scripted Web Services → Scripted REST APIs**.

The following podcast offers additional information on the use of scripted REST APIs.

**Scripted REST APIs Podcast**

**Scripted REST API URIs**

Scripted REST API URIs have the following format:

- ![https://<instance.service-now.com>/api/<name_space>/<version>/<api_id>/<relative_path>](https://<instance.service-now.com>/api/<name_space>/<version>/<api_id>/<relative_path>)

In this URI:

- `<instance.service-now.com>`: Path to the ServiceNow instance where users access the scripted REST API.

- `<name_space>`: For web services in the global scope, the name space is the value of the property `glide.appcreator.company.code`. For web services in a scoped application, the name space is the scope name, such as

---

© 2021 ServiceNow, Inc. All rights reserved.

ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
x_company_appname. For additional information on name spaces, see
Application scope.

• <version>: Optional. Version of the endpoint to access if the API uses
  versioning, such as v1. You can access the default version of a versioned API
  by specifying the URI without a version number.

• <api_id>: Value of the API ID field on the Scripted REST Service form. By default
  this value is based on the service name.

• <relative_path>: Relative path defined for the resource in the Scripted REST
  Service form. Specifying a relative resource path allows you to have multiple
  resources using the same HTTP method, such as GET, in one web service. For
  example, a resource may specify the path /{id} when the web service has
  only one GET resource, or /user/{id} and /message/{id} when the web service
  has different resources for requesting user and message records.

Scripted REST API versioning

Scripted REST API URIs may include a version number, such as /api/management/
  v1/table/{tableName}. Version numbers identify the endpoint version that a URI
  accesses. By specifying a version number in your URIs, you can test and deploy
  changes without impacting existing integrations.

Default API version

A version may be marked as default. Specifying a default version allows users to
  access that version using a scripted REST endpoint without a version number. If
  no version is marked as default, the latest version is used as the default.

Scripted REST API resources

A scripted REST API resource is equivalent to a REST endpoint. It defines the HTTP
  method to execute, the processing script, and any override settings from the
  parent API. You can define one or more resources per API.

Scripted REST API query parameters

Query parameters define values that requesting users can pass in a request.
  When creating a scripted REST API, you can specify which parameters are
  available and which are mandatory for each request. You can also associate a
  query parameter with multiple resources.

Access request parameters in scripts using the request object params field.

Scripted REST API roles

To work with scripted REST APIs, you must have the web service administrator
  [web_service_admin] role. Users with this role can read, create, modify, and
  delete scripted REST APIs and web service resources.
Note: These roles are not required to access a scripted REST API endpoint.

Request and response formats
By default, all resources in an API support the following request and response formats: application/json, application/xml, and text/xml. You can override the default formats at the API level. The new formats apply to all resources belonging to the API, unless an individual resource overrides the defaults.

Scripted REST API security
You can configure your scripted REST APIs with the necessary level of security. From public APIs/endpoints that don’t require any security to highly secure APIs/endpoints that require user authentication with tight access control to all resources.

Scripted REST API access controls
Access control lists (ACLs) define criteria, such as the roles needed and conditions that a user must meet to access a scripted REST API or endpoint. A requesting user must satisfy at least one of the ACLs. It is not necessary to satisfy all selected ACLs. You can define a single ACL for an entire REST API or for an individual endpoint.

Note: By default, scripted REST APIs contain an ACL that prohibits users with the snc_external role from making requests to the API.

When defining a scripted REST API ACL, it must have the Type value REST_Endpoint.

For additional information on ACLs, see Access control list rules and Configure a scripted REST API resource to require an ACL.

Scripted REST API security matrix
There are multiple possible security configurations for scripted REST APIs. Use this table to identify the scripted REST API security configuration that best suits your needs, and the field values to implement that configuration.
### Scripted REST API security

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Scripted REST API</th>
<th>Scritped REST Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Default ACLs</td>
<td>Requires authentication</td>
</tr>
<tr>
<td>The resource is public. No authentication or ACL is required.</td>
<td>Any value</td>
<td>False</td>
</tr>
<tr>
<td>The resource requires basic authentication only. No ACL is required.</td>
<td>Any value</td>
<td>True</td>
</tr>
<tr>
<td>The resource requires basic authentication only. ACL is required.</td>
<td>No ACL selected</td>
<td>True</td>
</tr>
<tr>
<td>An ACL selected in the resource record is required.</td>
<td>Any value</td>
<td>True</td>
</tr>
<tr>
<td>An ACL selected in the scripted REST API record is required.</td>
<td>One or more ACLs selected</td>
<td>True</td>
</tr>
</tbody>
</table>

### Scripted REST API error objects

Scripted REST APIs include error objects that allow you to respond to a request with a standard HTTP error message when an error occurs during request processing. You can use error objects in scripted REST API resources to alert requesting clients of errors. Use error objects to respond to incoming requests, not to catch errors within your server-side code.

### Error response format

The content type of the response depends on the request Accept header. If the Accept header specifies an unsupported format, such as image/jpeg, the error response uses JSON.

Error responses follow this format:

```json
{
    "error": {
        "message": "My error message"
    }
}
```
The numeric status code, such as 404, is included in the response Status code header, not in the response body.

**Automated Test Framework support**

The Automated Test Framework (ATF) supports Inbound REST test steps. You can create automated tests for custom Inbound REST APIs that you create. Creating tests for your custom REST APIs simplifies upgrade testing, and makes it possible to verify that modifications to a REST API are backward compatible. See Administering REST test step configurations and ATF REST test step configurations.

**Developer training**

In the ServiceNow® Developer Site, you can find training for **Scripted REST APIs**.

**Create a scripted REST API**

Create a scripted REST API to define web service endpoints.

**Before you begin**
Role required: web_service_admin

**About this task**
By default, scripted REST APIs contain an ACL that prohibits users with the snc_external role from making requests to the API.

**Procedure**

1. Navigate to **System Web Services > Scripted REST APIs**.
2. Click **New**.
3. Enter a **Name** for the service.
   - The **API ID** is set automatically based on the **Name**. You can modify the **API ID** if needed.
4. Click **Submit**.

**What to do next**
After you create the API, configure the service as needed such as by creating resources, assigning ACLs, or specifying supported request and response formats.
Create a scripted REST API resource

Create a scripted REST API resource to define the HTTP method, the processing script, and to override settings from the parent service.

**Before you begin**
There must be a scripted REST API defined before you can create resources.
Role required: web_service_admin

**About this task**
By default, any new Scripted REST API resource you create contains an ACL that prohibits users with the snc_external role from making requests to the API.

**Procedure**
1. Navigate to **System Web Services > Scripted REST APIs**.
2. Select a scripted REST API record.
3. In the **Resources** related list, click **New**.
4. Enter a **Name**.
   - The resource name affects the URI for sending requests to the API.
5. Select the **HTTP method** this resource implements, such as **GET**.
6. In the **Script** field, define how the operation parses and responds to requests.
7. **Optional:** Override settings from the parent REST API as needed, such as by specifying different security settings or valid content types.
8. **Optional:** On the **Documentation** tab, provide a **Short description** explaining how to access the resource.
   - This information appears when exploring this resource using the REST API Explorer.
9. Click **Submit**.

**What to do next**
After creating the resource, you can associate headers and query parameters. For details, see **Define scripted REST API headers** and **Define available query parameters**.

**Define scripted REST API headers**
Define scripted REST API headers to control which headers the API accepts and can respond with.

**Before you begin**
There must be a scripted REST API defined before you can create headers.
Role required: web_service_admin

Procedure
1. Navigate to System Web Services > Scripted REST APIs.
2. Select a scripted REST API record.
3. In the Request Headers related list, click New.
4. Enter a Header name.
5. Enter a Short description and Example value to explain how to use the header.
6. Click Submit.
7. Optional: From the Request Headers tab, locate the header name and set Is required to "true" to make this header mandatory for all requests to associated scripted REST resources.

What to do next
After defining available headers, associate the headers with a scripted REST resource.

Define available query parameters
Define available query parameters to control what values a requesting user can pass in the request URI.

Before you begin
Role required: web_service_admin

Procedure
1. Navigate to System Web Services > Scripted REST APIs.
2. Select a scripted REST API record.
3. In the Query Parameters related list, click New.
4. Specify the Query parameter name.
5. Enter a Short description and Example value to explain how to use the parameter.
6. Click Submit.
7. Optional: From the Query Parameters tab, locate the query parameter and set Is required to "true" to make this query parameter mandatory for all requests to associated scripted REST resources.
What to do next
After defining available query parameters, associate the parameters to a scripted REST resource. For details, see Associate query parameters to a resource.

Associate query parameters to a resource
Associate scripted REST API query parameters to a resource.

Before you begin
Role required: web_service_admin

Procedure
1. Navigate to System Web Services > Scripted REST APIs.
2. Select a scripted REST API record.
3. In the Resources related list, click New
4. In the API query parameter, select or enter the query parameter to associate with the resource.
5. In the API resource, select or enter the scripted REST API resource to associate with the query parameter.
6. Click Submit.

What to do next
After associating the parameters with a scripted REST resource, configure any required ACLs for the API or endpoint. For details, see Configure a scripted REST API resource to require an ACL.

Configure a scripted REST API to require an ACL
Requests to scripted REST APIs respect platform ACLs, and the requesting user must meet any table ACL requirements to access instance data. Additionally, you can configure the scripted REST API to require a specific ACL.

Before you begin
Role required: web_service_admin

About this task
The ACLs selected in this task apply to all API endpoints.

Procedure
1. Navigate to System Web Services > Scripted REST APIs.
2. Select a scripted REST API.
3. In the **Default ACLs** field, select one or more ACLs that meet the security needs for the API. Select only those ACLs that have a **Type** of **REST_Endpoint**. A requesting user must satisfy at least one of the selected ACLs. It is not necessary to satisfy all selected ACLs.

4. Click **Update**.

**What to do next**
You can override the API security settings for each individual API resource/endpoint. For details, see **Configure a scripted REST API resource to require an ACL**.

**Configure a scripted REST API resource to require an ACL**
By default, API resources/endpoints inherit security settings from the parent API. Define custom ACLs for a specific resource/endpoint to override the inherited settings.

**Before you begin**
Role required: web_service_admin or admin

**About this task**
ACLs are checked for an authenticated user only.

**Procedure**
1. Navigate to **System Web Services > Scripted REST APIs**.
2. Select a scripted REST API.
3. In the **Resources** related list, select a resource.
4. In the **Security** tab, select the **Requires authentication** check box. You must select this check box to require an ACL for the resource. If you clear this check box, the resource becomes public and requires no credentials. Clear this check box only if you want to allow unauthenticated requests to access the resource, even if the parent REST service requires an ACL.
5. Select the **Requires ACL authorization** check box.
6. In the **ACL** field, select one or more ACLs that meet the security needs for the endpoint. Select only those ACLs that have a **Type** of **REST_Endpoint**. Only users who have roles defined in the selected REST_Endpoint type ACL are granted access to this resource. Selecting an ACL for a resource overrides any ACLs selected for the parent web service. Leave this field blank to use the ACLs selected for the parent web service.
Related information

Configure a scripted REST API to require an ACL

Enable versioning for a scripted REST API

Enable versioning for a scripted REST API to provide multiple versions of the API while maintaining compatibility with existing integrations.

Before you begin
There must be a scripted REST API defined before you can enable versioning.
Role required: web_service_admin

Procedure

1. Navigate to System Web Services > Scripted REST APIs.
2. Select a scripted REST API.
3. Under Related Links, click Enable versioning.
   The Enable versioning pop up appears. The Make version v1 default check box is selected by default.
4. Optional: Clear the Make version v1 default check box to enable versioning without a default version.
   Versioned APIs without a default version are accessible only by using the version-specific URI. Make version v1 default, or select a different version as default after you enable versioning.
5. Click OK.
   The Versioning embedded list is added to the Scripted REST Service form. You can add new versions or control which version is default from this list.

Add a version to a scripted REST API

Add a new version to a versioned scripted REST API to define new API behavior without impacting older versions.

Before you begin
There must be a scripted REST Service that has versioning enabled before you can add a new version.
Role required: web_service_admin

Procedure

1. Navigate to System Web Services > Scripted REST APIs.
2. Select a scripted REST API.
   The Add new version pop up appears.

4. Optional: Select Make this version the default to configure the REST service to 
   use the new version as the default version.

5. Optional: In the Copy existing resources from version choice list, select an 
   existing version to copy all resources from that version to the new version.

6. Click OK.

Control request and response content type
Controls which content types are allowed in scripted REST API requests and 
responses.

By default, scripted REST APIs support application/json, application/xml, and 
text/xml. User-defined custom content types (with json or xml subtypes) are 
also supported. For example, application/vnd.collection+json and application/
vnd.adobe.xdp+xml are treated as JSON and XML, respectively.

Important: If the request body format is not of a json or xml subtype, use 
only the request body dataStream field to access the request body. Using 
request body data, dataString, nextEntry(), or hasNext() with a non-json or 
non-xml format results in a 500 error response.

Setting defaults
You can set default values for the API using the Default supported request 
formats and Default supported response formats fields. These fields define 
acceptable values users can pass in the Content-Type and Accept request 
headers, respectively. If a requesting user specifies an Accept or Content-Type 
header not supported by the API or resource, the instance responds with an 
HTTP error code of 406 or 415.

You can override these values for each resource using the Supported request 
formats and Supported response formats on the Scripted REST Service form.

Note: The Supported request formats field appears only for PUT, POST, and 
PATCH resources.

Using wildcard values
You can use wildcard values when specifying valid content types.

• To perform a single-character wildcard search, use the percent sign (%) 
  character. This wildcard finds words that contain any one character in place 
  the percent-sign-character. For example, to find words such as text or test, 
  search for: te%t.
• To perform a multiple-character wildcard search, use the asterisk (*) character. This wildcard finds words that contain zero or more characters in place of the asterisk-character. For example, to find words such as planned or placed, search for: `pl*d`.

**Using the x-www-form-urlencoded content type**

If a REST API or resource accepts the application/x-www-form-urlencoded Content-type, you can retrieve the urlencoded values provided in the request as a JSON map. You can then supply these urlencoded key-value pairs as query parameters, in the request body, or both. They are combined and stored in the request parameters. Access these parameters through the `request.queryParams` object.

For example, if your API is defined to accept the application/x-www-form-urlencoded content-type and your API is implemented as follows,

```javascript
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
    response.setBody(request.queryParams);
})(request, response);
```

... then the following request yields the respective response:

POST to localhost:8080/api/now/some_api/some_resource?name3=value3&name4=value4

**Body:**

name1=value1&name2=value2

**Response:**

```
{
    "result":
    {
        "name4": [ "value4" ],
        "name3": [ "value3" ],
        "name2": [ "value2" ],
        "name1": [ "value1" ]
    }
}
```

**Sending binary type in a response**

When sending a binary type in a response, you must set the response content type and write the binary stream directly using a `RESTAPIResponseStream` object. You can access this object by calling `getStreamWriter()` on the response object. For more information, see `RESTAPIResponse - Scoped, Global`. 
Controlling maximum request size

You can specify the maximum file size allowed in a scripted REST API request payload.

The file size limit applies when accessing any of the following variables or functions from a RESTAPIRequestBody object.

- `data`
- `dataString`
- `nextEntry()`
- `hasNext()`

Accessing these variables or functions with a request payload that exceeds the maximum size causes the service to respond with error code 400.

The file size limit does not apply when accessing the `dataStream` variable.

Maximum request size properties

Several properties control the maximum allowed request size. Add these properties to the System Properties [sys_properties] table to specify a maximum request size.

<table>
<thead>
<tr>
<th>Property</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>glide.rest.scripted.max_inbound_content_length_mb</td>
<td>10</td>
<td>The maximum size, in megabytes, for a scripted REST request body that is not gzip compressed.</td>
</tr>
<tr>
<td>glide.rest.scripted.max_inbound_gzip_content_length_mb</td>
<td>1</td>
<td>The maximum size, in megabytes, for a scripted REST request body that is gzip compressed.</td>
</tr>
<tr>
<td>glide.rest.max_content_length</td>
<td>10</td>
<td>The maximum size, in megabytes, for a scripted REST request body, whether or not it is gzip compressed.</td>
</tr>
</tbody>
</table>

- Maximum: 25

As a result, even if `glide.rest.scripted.max_inbound_content_length_mb` or `glide.rest.scripted.max_inbound_gzip_content_length_mb` are set, the request body is limited to the value of `glide.rest.max_content_length`.

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Scripted REST APIs good practices

Follow these guidelines when designing and implementing scripted REST APIs.

Follow REST API conventions

Use REST API standards to provide a consistent and easy to use interface for clients. REST API conventions define specific behavior for each type of method. Use the following guidelines as a starting point for designing your API.

- **GET** operations only query data. A GET request should never modify data.
- **POST** operations create new records but do not modify existing records.
- **PUT** and **PATCH** operations modify existing records.
- **DELETE** operations destroy records.

Use versioning to control changes to your API

Use versioning to implement new functionality and avoid breaking existing integrations. When you introduce significant functionality changes to your API, create a new version of the API first. Do not introduce behavior that will break existing integrations in a published version.

Using versioning allows you to implement significant changes to your API without breaking existing clients. You can then release the new version of the API for new clients while allowing existing clients to upgrade at their own pace.

Encourage clients to use a version-specific API, or configure the API without a default version to force clients to specify a version. You can also make new, optional behavior available by adding an optional parameter to an existing version.

Return an informative HTTP status code

Return a status code that informs the requestor of the success or failure of the request. Return an HTTP status code that helps the client understand the result of the request. Use the following guidelines for common status codes.

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Indicates that the request was completed successfully.</td>
</tr>
<tr>
<td>201</td>
<td>Indicates that a record was created successfully.</td>
</tr>
<tr>
<td>204</td>
<td>Indicates that a record was deleted successfully.</td>
</tr>
</tbody>
</table>
Common status codes (continued)

<table>
<thead>
<tr>
<th>Status code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>40X</td>
<td>Status codes in the 400 range indicate a client error, such as 400 for invalid request syntax.</td>
</tr>
<tr>
<td>(401, 404)</td>
<td></td>
</tr>
<tr>
<td>50X</td>
<td>Status codes in the 500 range indicate that a server error occurred. The client request may have been valid or invalid, but a problem occurred on the server that prevented it from processing the request.</td>
</tr>
<tr>
<td>(500, 503)</td>
<td></td>
</tr>
</tbody>
</table>

Return useful error information

Provide the client with enough information in error messages to allow them to understand the problem without having to refer to your API documentation. An error response should include a helpful error message, as well as an error status code.

For example, when a client queries a record that does not exist, you can return the error message "The specified record does not exist. Ensure that a record with the ID of <id value> exists in the application." along with a 404 status code.

The scripted REST API feature includes several preconfigured error objects you can use for commonly-encountered errors, and a customizable ServiceRequest error object you can use when the preconfigured error objects do not meet your needs.

Enforce and test access controls

Enforce existing access controls and require additional access to modify data. In addition to requiring authentication to access the API, require authorization to access data. Use the GlideRecordSecure API in your scripted REST API scripts. This API ensures that access controls defined on the underlying data are applied for the requesting user.

Require additional access controls for operations that modify data. Requests such as PUT, POST, and DELETE should require a higher level of access than GET. Configure these API resources to require a more strict ACL.

Test your access controls, both authentication and authorization, before releasing the API.

Build tests to verify functionality

Build tests that verify your scripted REST web services functionality as part of your development process. Use repeatable tests to ensure that your API functions
the way you expect it to. Testing also helps ensure that changes you make do not affect the expected API behavior after you release a version. You can use a REST client application that supports automated testing, such as Postman, to facilitate testing.

Tests should validate the response code, headers, and body content as appropriate for each resource you implement. You can also use tests to validate authentication requirements, and to confirm that errors return useful responses.

**Scripted REST API examples**

Multiple examples are available demonstrating how to create and use scripted REST APIs.

**Scripted REST API example - script samples**

These examples demonstrate how to create various resource scripts for a scripted REST API.

**Query parameters GET example**

This example demonstrates how to get query parameter values from a request.

```javascript
/**
 * GET - Sample Request API - Query Params
 */
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
    var uri = request.uri;
    var url = request.url;
    var queryParams = request.queryParams;
    var customHeader = request.getHeader('X-Custom');

    return {
        "uri": uri,
        "url": url,
        "queryParams": queryParams,
        "customHeader": customHeader
    };
})(request, response);
```

**Path parameters GET example**

This example demonstrates how to get path parameter values from a request.
/**
* GET - Sample Request API - Path Params
*/
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
    var uri = request.uri;
    var url = request.url;
    var path = request.pathParams;

    return {
        "uri": uri,
        "url": url,
        "path_params": path,
        "path.id": path.id
    };
})(request, response);

**Script include GET example**

This example demonstrates how to use a script include to provide a response. By using a script include you can reuse common code and maintain readability in the REST service scripts.

/**
* GET - Sample Request API - Script Include
*/
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
    var responseObj = global.SampleDataUtil.getSampleJSON();
    return responseObj;
})(request, response);

**String POST example**

This example demonstrates how to parse a POST message with a string body and send a response based on the request.

/**
* POST - Sample Request API - dataString
* sample usage:
* var requestBody = request.body;
* var requestString = requestBody.dataString;
*/
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
    var requestBody = request.body;
    var requestString = requestBody.dataString;

Binary POST example
This example demonstrates how to parse a POST message with a binary body and send a response based on the request.

```javascript
/**
 * POST - Sample Request API - Body
 */
(function process /*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
    var body = request.body.data;
    //do any additional processing on the request body, such as inserting a new record.
    return {
        "body.id": body.id
    }
})(request, response);
```

Not acceptable error example
This example demonstrates how to respond with a not acceptable error. Use this error type when the request Accept header value is not supported by the web service.

```javascript
/**
 * Sample Not Acceptable Error Sample
 */
(function process /*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
    response.setError(new sn_ws_err.NotAcceptableError('sample error message'));
})(request, response);
```

Bad request error example
This example demonstrates how to respond with a bad request error. Use this error type to indicate a mistake in the request syntax.

```javascript
/**
 * Bad Request Error Sample
 */
(function process /*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
    response.setError(new sn_ws_err.BadRequestError('sample error message'));
})(request, response);
```
Conflict error example
This example demonstrates how to respond with a conflict error. Use this error type in the event of multiple conflicting requests, such as multiple updates to the same record.

```javascript
/**
 * Error Response: Conflict Error Sample
 */
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
  response.setError(new sn_ws_err.ConflictError('sample error message'));
})(request, response);
```

Not found error example
This example demonstrates how to respond with a not found error. Use this error type if the requested resource does not exist or is unavailable.

```javascript
/**
 * Error Response: Not Found Error Sample
 */
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
  response.setError(new sn_ws_err.NotFoundError('sample error message'));
})(request, response);
```

Unsupported media type error example
This example demonstrates how to respond with an unsupported media type error. Use this error type to indicate that the Content-Type of the request is unsupported.

```javascript
/**
 * Error Response: Unsupported Media Type Error Sample
 */
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
  response.setError(new sn_ws_err.UnsupportedMediaTypeError('sample error message'));
})(request, response);
```

Service error example
This example demonstrates how to respond with a generic service error. The ServiceError object allows you to define the status code, message, and error detail. Use a ServiceError if the predefined error types do not meet your needs.

```javascript
/**
 * Error Response: Custom Error Sample
 */
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
var myError = new sn_ws_err.ServiceError();
myError.setStatus(418);
myError.setMessage("I am a Teapot");
myError.setDetail("Here are the details about this error");
response.setError(myError);
})((request, response);

**Scripted REST resource script example**
This sample REST API resource script parses the name and id values from the request body and returns those values in the response.

```javascript
/**
 * POST - Sample Request API - Body
 */
(function process(/*RESTAPIRequest*/ request, /*RESTAPIResponse*/ response) {
var body = request.body.data,
id0,name0,id1,name1;
name0 = body[0].name; // 'user0'
id0 = body[0].id; // '1234'
namel = body[1].name; // 'user1'
id1 = body[1].id; // '5678'

return {
"id": id0,
"name": name0,
"id1": id1,
"name1": namel
};
})((request, response);

**Requests**
The API can accept both XML and JSON requests.

<table>
<thead>
<tr>
<th>JSON Request</th>
<th>XML Request</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST /api/sn_demo_api/v1/example/body HTTP/1.1 Content-Type: application/json Accept: application/json Host: &lt;instance&gt;.service-now.com Connection: close Content-Length: 91</td>
<td>POST /api/sn_demo_api/v1/example/body HTTP/1.1 Content-Type: application/xml Accept: application/json Host: &lt;instance&gt;.service-now.com Connection: close Content-Length: 152</td>
</tr>
</tbody>
</table>
Requests (continued)

<table>
<thead>
<tr>
<th>JSON Request</th>
<th>XML Request</th>
</tr>
</thead>
</table>
| \[
|   { "name": "user0",  
|      "id": 1234    
| },
|   {  
|      "name": "user1",  
|      "id": 5678    
| }\] | <request><entry>
| |   <name>user0</name>
| |     <id>1234</id>
| | </entry>
| | <entry>
| |   <name>user1</name>
| |     <id>5678</id>
| | </entry>
| | </request> |

Responses

Both requests specify application/json as the Accept header value. This causes either response to use JSON formatting, even if the request content type is XML.

HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Transfer-Encoding: chunked
Date: Tue, 04 Aug 2015 15:20:44 GMT
Server: ServiceNow
Connection: close
Set-Cookie: BIGipServerpool_<Instance>=880838154.47166.0000; path=/

{"result":{"id":1234,"id1":5678,"name":"user0","name1":"user1"}}

Scripted REST API example - streaming vs object serialization

These examples demonstrate how to send a JSON response using streaming and using default object serialization.

Streaming vs object serialization

When sending a response, you can send a response as a stream or serialize an object. There are advantages and disadvantages to either approach. Pick a technique based on the needs of your integration.

Generally, if the response object is simple, can be represented as XML or JSON, and is a consistent size, use object serialization. If using a format other than XML or JSON, or if the size of the response varies, use streaming.
Streaming the response

Using a streaming responses provides advantages in response time, instance performance, and content flexibility, but adds additional complexity to the script. When using streaming, you are responsible for formatting the response, setting the response status, and setting the Content-Type header. When streaming a response, the requesting user receives a response quickly because the entire response does not need to be created before starting streaming.

This example demonstrates a Scripted REST Resource script that returns an array of incident records using streaming.

```javascript
/**
 * Sample Scripted REST Resource that returns custom JSON objects with properties from Incident GlideRecords
 * This sample uses ServiceNow JavaScript API to query incident records and then iterates over those records to build and stream a custom JSON object that includes some values from the incidents
 */
(function runOperation(/*RESTServiceRequest*/ request, /*RESTServiceResult*/ response) {
    var writer = response.getOutputStream(),
        hdrs = {},
        table = 'incident',
        record_limit = 100,
        gr = new GlideRecord(table);

    hdrs['Content-Type'] = 'application/json';
    response.setStatus(200);
    response.setHeaders(hdrs);

    gr.setLimit(record_limit);
    gr.query();

    // start building response object
    writer.writeString('{"results":[');

    // iterate over incident records and build JSON representations to be streamed out.
    while (gr.next()) {
        var incidentObj = {};

        incidentObj.number = gr.number + '';
        incidentObj.short_description = gr.short_description + '';
        writer.writeString(global.JSON.stringify(incidentObj));
    }

    writer.writeString(']}
```
if (gr.hasNext()) {
    writer.writeString("",");
}

// close the response object
writer.writeString("})");
})/(request, response);

A request to this resource returns the following response.

// sample response
/*
HTTP/1.1 200 OK
Content-Type: application/json
Server: ServiceNow

// sample response number of records returned has been truncated for this example

{
  "results": [
  {
    "number": "INC0011301",
    "short_description": "lorem ipsum short description 0 my new incident"
  },
  {
    "number": "INC0011302",
    "short_description": "lorem ipsum short description 1 my new incident"
  },
  {
    "number": "INC0011303",
    "short_description": "lorem ipsum short description 2 my new incident"
  },
  {
    "number": "INC0011304",
    "short_description": "lorem ipsum short description 3 my new incident"
  },
  {
    "number": "INC0011309",
    "short_description": "lorem ipsum short description 8 my new incident"
  }
]
Building an object

Using object serialization allows you to take advantage of ServiceNow provided serialization and content negotiation. When serializing an object instead of streaming, the entire object must be created and serialized before the client receives a response. This may delay the response, or require a large amount of system resources if the response object is very large. Object serialization is available only for XML or JSON responses. Responses using a different format must use streaming.

This example returns the same Incident data as the streaming example, but collects all of the response data in an array before sending the response.
/**
 * Sample Scripted REST Resource returns an array of custom JSON objects that include 2 incident properties.
 * This sample uses ServiceNow JavaScript API to query incident records and then iterates over those records building a custom JSON object that includes 2 values from the incident records.
 * Note that because we are returning a simple JSON object we can rely on built in serialization to set the content-type header as well as response status. The 'result_arr' object will not be returned until it has been completely built and stored
 */
(function runOperation(/*RESTServiceRequest*/ request, /*RESTServiceResult*/ response) {
    var table = 'incident',
        record_limit = 100,
        result_arr = [],
        gr = new GlideRecord(table);

    gr.setLimit(record_limit);
    gr.query();

    // iterate over incident records and build JSON representations to be streamed out.
    while (gr.next()) {
        var incidentObj = {};

        incidentObj.number = gr.number + '';
        incidentObj.short_description = gr.short_description + '';

        result_arr.push(incidentObj);
    }

    return result_arr;
})(request, response);

A request to this resource returns the following response.

/*
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Server: ServiceNow
*/
Scripted REST API example - streaming file attachments

This example demonstrates how to send an image attachment to a requesting user as a binary stream.
A request to this resource returns the following response.

// sample response
/*
HTTP/1.1 200 OK
Set-Cookie: glide_session_store=SYSTEM; Expires=Fri, 30-Oct-2015 21:57:00 GMT; Path=/;
HttpOnly
Content-Type: image/jpeg
Transfer-Encoding: chunked
Date: Fri, 30 Oct 2015 21:26:59 GMT
Connection: close
Server: ServiceNow

<binary response body excluded from this sample>
*/

GraphQL API framework

Create a custom GraphQL API to query record data from a component or a third-party system.

For example, you can create a component that displays the cases associated with an SLA. You can use the Now® Experience UI Framework to develop the component you need, and access case data from the platform by creating a GraphQL schema that defines data in the Case table.
To learn more about developing components, see Developing components for Workspace.

**Benefits of GraphQL**

GraphQL is a web query language optimized for client-side development. Using scripted GraphQL, you can:

- Discover fields and objects available to query through introspection.
- Query the exact data you need from a component.
- Manage multiple possible queries from a single API, as opposed to multiple endpoints for a REST request.
- Integrate with third-party systems by making the schema public.
- Generate the GraphQL query from your component and handle the response.

**What to know before you begin**

Before you start creating custom GraphQL APIs, make sure you have:

- GraphQL knowledge to create a schema.
- JavaScript knowledge to define the API behavior.
- General knowledge of web component concepts.
- A custom Workspace component to consume record data.
- Understanding of the ServiceNow data model that you want to expose in the schema.
- GlideRecord knowledge to map fields to record data in your resolver scripts.

**GraphQL overview**

Creating a scripted GraphQL API includes these parts:

**GraphQL Schema Definition Language (SDL)**

Define the structure and data type of fields available in a GraphQL query. You can define the SDL using the Schema script field in the GraphQL Scripted Schemas [sys_graphql_schema] table. The SDL only supports Query and Mutation operations.

**Resolvers**

Define the data returned by each field. You can define the resolvers for each field in the GraphQL Scripted Resolvers related list on the GraphQL Scripted Schemas form.
Typeresolvers
Resolve interfaces and unions into concrete GraphQL types. For example, you might define a union between an incident type and a problem type. Use the typeresolver script to define when to return which. You can define the typeresolvers in the GraphQL Scripted Typeresolvers related list on the GraphQL Scripted Schemas form.

Resolver mappings
Map resolvers to fields in the schema. You can define resolver mappings in the GraphQL Scripted Resolver Mappings related list on the GraphQL Scripted Schemas form.

To learn more about the GraphQL query language, see the GraphQL website.

Limitations
The following GraphQL features are not supported:

- Subscription operations
- Custom scalar types

Introspection
By default, introspective queries into your custom schemas are not allowed. To allow introspection, set a system property. See GraphQL system properties.

Namespaces
GraphQL APIs have two different namespaces:

- **Application namespace**
  The namespace for the custom application. To learn more about application namespaces, see Application scope.

- **Schema namespace**
  The namespace for the schema to ensure that all queries are unique. You can have multiple schema namespaces in a single application.

When querying data, you must include both namespaces in your query. For example, the following query is searching for data with the following namespaces:

- Application namespace: x_graph_scope
- Schema namespace: planet
query {
  x_graph_scope {
    planet {
      findAll {
        name
        mass
        distance
      }
    }
  }
}

Directives and global functions

@source schema directive
Maps a GraphQL field to the value of a property of the parent object. If the field has a separate resolver script, the system uses the record that it resolves to instead of the parent object.
Use in your schema script.

Resolver functions

These functions are available on the global env object.

• getArguments(): Returns the arguments of the previous field.
• getSource(): Returns the parent object.

Use in your resolver script.

Typersolver functions

These functions are available on the global env object.

• getArguments(): Returns the arguments of the previous field.
• getobject(): Returns the parent object.
• getTypeName(): Returns the name of the interface or union type.

Use in your typersolver script.

Demo application

To see a demo GraphQL PTO calendar schema with mutations and queries, enable the GraphQL Framework Demo Application plugin (com.glide.graphql.framework.demo).
Create a GraphQL schema

Create a GraphQL schema to make data available to GraphQL queries.

Before you begin
Role required: admin

Procedure
1. Navigate to System Web Services > GraphQL > Schemas.
2. Click New.
3. Complete these fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the schema.</td>
</tr>
<tr>
<td>Schema namespace</td>
<td>Must be unique within the name of the application.</td>
</tr>
<tr>
<td>Application</td>
<td>Read-only application that the schema is a part of.</td>
</tr>
<tr>
<td>Application namespace</td>
<td>Read-only. Works with the Schema namespace to prevent conflict with schemas with the same name.</td>
</tr>
<tr>
<td>Active</td>
<td>Checked if active, unchecked if not active.</td>
</tr>
<tr>
<td>Schema</td>
<td>Schema definition that adheres to the GraphQL SDL format. Must be GraphQL valid syntax. Otherwise, error messages appear on save to indicate the syntax problem.</td>
</tr>
</tbody>
</table>

Note: The following GraphQL features are not supported:
- Subscription operations
- Custom scalar types

You can use these directives in your schema:
- `@source`: Maps a GraphQL field to the value of a property of the parent object. If the field has a separate resolver script, the system uses the record that it resolves to instead of the parent object.

Example
This example defines an Incident object type in the schema and uses a resolver script to map the type to a GlideRecord from the Incident table. Using
the @source directive maps the fields within the Incident type to the value or display_value in the Incident GlideRecord.

```graphql
type Incident {
  id: String @source(value: "sys_id.value")
  active: Boolean @source(value: "active.display_value")
  state: String @source(value: "state.display_value")
  priority: String @source(value: "priority.display_value")
  severity: String
  description: DisplayableString
  resolvedBy: User @source(value: "resolved_by.value")
  openedBy: User @source(value: "opened_by.value")
  child_incidents: String
  opened_at: String
  resolved_at: String
  closed_at: String
  work_notes: String
  comments: String @source(value: "comments.display_value")
  parent_incident: String
}
```

4. Select the **Security** tab and complete the form.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requires authentication</td>
<td>Checked if the schema requires authentication, unchecked if the schema doesn’t require authentication and is available publicly.</td>
</tr>
<tr>
<td>Requires SNC internal</td>
<td>Appears only if the Explicit Roles plugin is enabled. Checked if the schema requires the SNC internal role, unchecked if the schema doesn’t require the SNC internal role.</td>
</tr>
<tr>
<td>Requires ACL authorization</td>
<td>Checked if the schema requires ACL authorization, unchecked if the schema doesn’t require ACL authorization.</td>
</tr>
<tr>
<td>ACLs</td>
<td>Appears only if Requires ACL authorization is checked. Checks incoming queries against ACLs of type <strong>GraphQL</strong>.</td>
</tr>
</tbody>
</table>

**Note:** The API returns clear error messages to users who do not have access to a schema, or who are not authenticated when the API requires authentication to access.
5. Save the form.

6. **Optional:** Create a resolver script to define what value the schema returns when a component queries a specific field. If you don’t define a resolver for a field, the query returns any matching field value from the parent object type. For example, suppose you have an Incident object type in your schema with a worknotes field. The Incident object type has a resolver that maps to a GlideRecord from the Incident table. If you do not create a resolver mapping for the worknotes field, the system searches the parent object’s data source, which is the GlideRecord from the Incident table, for a worknotes field and assigns the associated value.

   a. Select the GraphQL Scripted Resolvers tab and click **New**.

   b. Complete the form.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the resolver.</td>
</tr>
<tr>
<td>Schema</td>
<td>Read-only schema namespace.</td>
</tr>
<tr>
<td>Application</td>
<td>Read-only application that the schema is a part of.</td>
</tr>
<tr>
<td>Script</td>
<td>Define the value returned when the field is queried. Functions available on the global <code>env</code> object:</td>
</tr>
<tr>
<td></td>
<td>• <code>getArguments()</code>: Returns the arguments of the previous field.</td>
</tr>
<tr>
<td></td>
<td>• <code>getSource()</code>: Returns the parent object.</td>
</tr>
<tr>
<td></td>
<td>This script has access to GlideRecord.</td>
</tr>
</tbody>
</table>

**Example**

(Optional) This example returns a record from the User table when the associated field is queried:

```javascript
(function process(/*ResolverEnvironment*/ env) {
  var userid = env.getArguments().id != null ? env.getArguments().id : env.getSource();
  var now_GR = new GlideRecord('sys_user');
  gr.addQuery('sys_id', userid);
  gr.query();
```

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
c. Click Submit.

7. Optional: Define the typeresolvers for your schema to resolve interfaces and unions into concrete types.

a. Select the GraphQL Scripted Typeresolvers tab and click **New**.

b. Complete the form.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schema</td>
<td>Read-only schema namespace.</td>
</tr>
<tr>
<td>Type</td>
<td>The interface or union type defined in the schema.</td>
</tr>
<tr>
<td>Application</td>
<td>Read only application that the schema is a part of.</td>
</tr>
<tr>
<td>Script</td>
<td>Define the value returned for union and interface types. Functions available on the global <code>env</code> object:</td>
</tr>
<tr>
<td></td>
<td>• <code>getArguments()</code>: Returns the arguments of the previous field.</td>
</tr>
<tr>
<td></td>
<td>• <code>getObject()</code>: Returns the parent object.</td>
</tr>
<tr>
<td></td>
<td>• <code>getTypeName()</code>: Returns the name of the interface or union type.</td>
</tr>
<tr>
<td></td>
<td>This script has access to GlideRecord.</td>
</tr>
</tbody>
</table>

c. Click Submit.

8. Optional: Map the resolver and typeresolver records to fields in the schema. This mapping lets the system know what value to return when a component queries a specific field.

a. Select the GraphQL Scripted Resolver Mappings tab and click **New**.

b. Complete the form.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Path to the field in the schema you want to map.</td>
</tr>
<tr>
<td>Resolver</td>
<td>Resolver to use to define the data returned by the field.</td>
</tr>
<tr>
<td>Application</td>
<td>Read-only application that the schema is a part of.</td>
</tr>
<tr>
<td>Schema</td>
<td>Read-only schema namespace.</td>
</tr>
</tbody>
</table>

**c. Click Submit.**

**Test a GraphQL schema**

Test your GraphQL schema using a third-party tool.

**Before you begin**

Role required: admin

- Create a GraphQL schema.
- Download a third-party GraphQL tool, such as GraphiQL or Insomnia.

**Procedure**

1. Open the third-party tool on your machine.
2. Set the endpoint to query your custom schema: `https://<your-instance>.servicenow.com/api/now/graphql`.
3. Add any authentication headers required.
4. **Optional:** If introspection is enabled, you can send introspective queries to learn more about what the schema includes. To turn on introspection, see GraphQL system properties.

ℹ️ **Note:** Do not use introspective queries in a production environment.

5. Send test queries to the schema.

   Include both the application and schema namespaces. This example includes both namespaces:

   - Application namespace: `x_graph_scope`
   - Schema namespace: `planet`

   ```
   query {
     x_graph_scope {
       planet {
   ```
What to do next
Query a GraphQL schema from a component.

Query a GraphQL schema from a component
Access record data in a component by querying your scripted GraphQL schema.

Before you begin
Role required: admin
1. Create a GraphQL schema.
2. Test a GraphQL schema.
3. Create a custom component.

Procedure
1. Navigate to your component directory.
2. Define the query in your component's index.js file.

⚠️ Note: Do not use introspective queries in a production environment.

Example
```javascript
const incidentQuery = `query ($sys_id: String!) {
  query (sys_id: $sys_id) {
    incident {
      _results {
        sys_id {
          displayValue
        }
      }
    }
  }
}
`;
```
3. Add the following line to the top of the script to import the `createGraphQLEffect` API into your component.

```javascript
import {createGraphQLEffect} from '@servicenow/ui-effect-graphql';
```

4. Call the query using the `createGraphQLEffect` API. To learn more about this API, see the Developer Site.

5. Create action handlers to manage the result of the query.

**Example**

```javascript
actionHandlers: {
  [DATA_FETCH_REQUESTED]: dataFetchHandler,
  [DATA_FETCH_STARTED]: ({updateState}) => updateState({loading: true}),
  [DATA_FETCH_SUCCEEDED]: ({action, updateState}) => updateState({data: action.payload}),
  [DATA_FETCH_FAILED]: () => { /* handle network error */ },
}
```

**Example:**
This example shows a complete component including the query, the GraphQL API call, and action handlers to manage the response.

```javascript
import {createCustomElement} from '@servicenow/ui-core';
import snabbdom from '@servicenow/ui-renderer-snabbdom';
import {createGraphQLEffect} from '@servicenow/ui-effect-graphql';

const GQL_QUERY = `query query ($active: Boolean) {
  now {
    mySchema (active: $active) {
      myField {
        value
      }
    }
  }
}`;

const DATA_FETCH_REQUESTED = 'DATA_FETCH_REQUESTED';
const DATA_FETCH_STARTED = 'DATA_FETCH_STARTED';
const DATA_FETCH_SUCCEEDED = 'DATA_FETCH_SUCCEEDED';
const DATA_FETCH_FAILED = 'DATA_FETCH_FAILED';

const dataFetchHandler = createGraphQLEffect(GQL_QUERY, {
  variableList: ['active'],
});
```
GraphQL system properties

Configure GraphQL API framework behavior. For example, you can configure whether to allow introspective queries into your schema.

Navigate to **System Web Services > GraphQL > Properties** to change your GraphQL settings.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>glide.graphql.introspection_enabled</td>
<td>Enables introspective queries to discover the supported GraphQL queries and types.</td>
</tr>
<tr>
<td></td>
<td>• Type: Boolean</td>
</tr>
<tr>
<td></td>
<td>• Default: False</td>
</tr>
</tbody>
</table>
SOAP web service

Simple Object Access Protocol (SOAP) is an XML-based protocol for accessing web services over HTTP.

You can use SOAP to access data on your instance. Available SOAP web services are WS-I compliant, as outlined in the WS-I Basic Profile 1.0.

Web service provider

ServiceNow publishes its underlying table structures and associated data using the following web service methods:

- **Direct web services**: Use a URL query to request a table’s WSDL.
- **SOAP web service import sets**: Use import tables and transform maps to automate web service requests for tables.
- **Scripted SOAP web services**: Use custom JavaScript to execute SOAP web services requests.

⚠️ **Note**: SOAP messages are sent with the assumption that the recipient is XML compliant. No encoding is applied to a SOAP message. SOAP always decodes responses as UTF-8, the XML encoding header is not used.

WSDL

All tables and import sets dynamically generate Web Service Definition Language (WSDL) XML documents that describe its table schema and available operations.

You can obtain a table’s WSDL by issuing a URL call to your instance that contains the name of the table and the **WSDL** parameter. For example:

```plaintext
https://myinstance.service-now.com/incident.do?WSDL
```

All dynamically generated and served ServiceNow WSDLs accessible via HTTP are available for use under the terms defined in the Open Source Initiative OSI - Apache License, Version 2.0 license agreement.

Long-running SOAP request support

The Now Platform supports long-running SOAP requests by preventing socket timeouts due to inactivity of the network connection while the requests are in process.

This functionality improves the efficiency of the ODBC driver when requesting large numbers of records, doing aggregate queries, or using order by expressions that require sorting.
By default, the system provides timeout protection for web services clients provided by ServiceNow such as the ODBC driver and the MID Server. You can add timeout protection to your custom web services with system properties.

**Timeout protection**

Web services clients receive a 307-Temporary Redirect to keep long sessions alive and prevent a timeout due to socket inactivity. A 307-Temporary Redirect causes web services clients which support the status code to repeat their last request to the location specified in the HTTP location header. The value of the location header is the same URL that the web services client originally specified. The use of 307-Temporary Redirects is WS-I compliant.

A web service request that exceeds the timeout limit specified in `glide.soap.request_processing_timeout` can only receive a 307-Temporary Redirect when all of these conditions are met:

- The value of `glide.soapprocessor.allow_long_running_threads` is true.
- The request includes a `redirectSupported=true` URL parameter.
- The request is session-aware (supports HTTP cookies).
- The number of redirects has not exceeded the value set by `glide.soap.max_redirects`.

If any of these conditions is not met, the web service client receives a 408 Request Timeout error.

**Note:** To ensure that applications experience a socket timeout rather than a 408 Request Timeout, set the `glide.soap.request_processing_timeout` property to a value larger than the shortest socket timeout setting in effect for the connection between the application and the instance (300 seconds for hosted instances).

**SOAP web services security**

An instance enforces web service security using a combination of basic authentication challenge/response over the HTTPS protocol and system-level access control lists (ACLs) using contextual security. Administrators can control what system resources web services users can access by granting them one of the SOAP roles.

**SOAP roles**

To use SOAP web services, you must have the appropriate role for the operation you want to perform. Also, you must have any other roles required to access the target tables.
### SOAP Roles

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>soap</td>
<td>Can perform all SOAP operations.</td>
</tr>
<tr>
<td>soap_create</td>
<td>Can insert new records.</td>
</tr>
<tr>
<td>soap_delete</td>
<td>Can delete existing records.</td>
</tr>
<tr>
<td>soap_ecc</td>
<td>Can query, insert, and delete records on the Queues [ecc_queue] table.</td>
</tr>
<tr>
<td>soap_query</td>
<td>Can query record information.</td>
</tr>
<tr>
<td>soap_query_update</td>
<td>Can query record information and update records.</td>
</tr>
<tr>
<td>soap_script</td>
<td>Can run scripts that specify a .do endpoint. This role is required for running scripted web services.</td>
</tr>
<tr>
<td>soap_update</td>
<td>Can update records.</td>
</tr>
<tr>
<td>import_admin</td>
<td>Can manage all aspects of import sets and imports. Required for access to the Import Set Row [sys_import_set_row] table.</td>
</tr>
</tbody>
</table>

### Default web services role requirements

By default, a set of processor ACL rules require users to have the soap_query role to make WSDL, XSD, and XML schema requests.

If you want to change these role requirements, you can deactivate the ACL rules.
Basic authentication

To enforce basic authentication for the user associated with the instance for each WSDL or SOAP message request, administrators can set the property glide.basicauth.required to true.

When enabled, each WSDL and SOAP request must contain an "Authorization" header as specified in the Basic Authentication protocol.

Because web services requests are non-interactive, the Authorization header is always required during a request.

⚠️ Note: If configured, basic authentication refers to local credentials or LDAP authentication.

Supplying basic authentication information with every request (whether or not it is required) has the added advantage that the user supplied in the basic authentication credentials can be associated with web service invocation. For example, when creating an Incident record, the journal field lists the user ID contained in the basic authentication header instead of the default guest user.

SOAP security policies

The Enhanced Web Service Provider - Common plugin adds the SOAP Security Policies module to the System Web Services application. This module allows administrators to set the following security policies:

- Enable or disable signing SOAP requests when consuming an external web service
- Specify the authentication requirements SOAP requests must meet when communicating over WS-Security.

© 2021 ServiceNow, Inc. All rights reserved.
**Certificates required for signed SOAP requests**

To sign SOAP requests for WS-Security communications, the following certificates are required:

- X.509 certificate from the requester
- X.509 CA certificate of the certificate authority who signed the requester’s certificate

**SOAP default security policy**

Administrators can specify the SOAP security policy an instance uses with the system property `glide.soap.default_security_policy`. The `glide.soap.default_security_policy` system property specifies the name of the SOAP security policy the instance uses when enforcing Web Services-Security (WSS) for inbound requests.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>String</td>
</tr>
<tr>
<td>Default value</td>
<td>Default Security Policy</td>
</tr>
<tr>
<td>Location</td>
<td>Add to the System Properties [sys_properties] table</td>
</tr>
</tbody>
</table>

**WS-Security**

You can validate signed web services requests using WS-security. Enable WS-Security to:

- Verify that SOAP messages originate from a known sender
- Verify that SOAP messages have not been altered in transit

ServiceNow supports **WS-Security 1.1** to validate signed web services requests.

⚠️ **Note:** WS-Security is not used as an encryption mechanism. HTTPS protocol is used to encrypt all communications.

WS-Security is intended to work with basic authentication. When an instance receives a SOAP message, it reviews the basic authentication header to determine if the SOAP user has rights to the instance. It reviews the WS-Security header to determine the validity of the incoming message. Requests affected by attacks, such as a man-in-the-middle attack, have an invalid WS-Security header and are blocked.
WS-Security profiles

A WS-security profile determines how a web services message is authenticated when WS-security is enabled. The following mechanisms can be used to authenticate web services requests:

**Web service authentication mechanisms**

<table>
<thead>
<tr>
<th>Authentication mechanism</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Certificate verification</td>
<td>Verifies the certificate associated with the request. Verifying the request's certificate requires uploading the requester's certificate and certificate authority.</td>
</tr>
<tr>
<td>User credentials</td>
<td>Authenticates the web services request by verifying the user credentials associated with the request. This type of authentication can either verify that the request's credentials match an existing user's credentials or that the request's credentials match a user name and password provided in the profile record.</td>
</tr>
</tbody>
</table>

Specify the authentication mechanism you want to use when you create a new WS-security profile.

The WS-Security Profiles module lists the WS-Security profiles that are currently in effect.

**WS-Security Profiles module**

<table>
<thead>
<tr>
<th>Type</th>
<th>Active</th>
<th>Order</th>
<th>Bind session</th>
</tr>
</thead>
<tbody>
<tr>
<td>X509</td>
<td>true</td>
<td>100</td>
<td>false</td>
</tr>
<tr>
<td>Username</td>
<td>true</td>
<td>200</td>
<td>true</td>
</tr>
</tbody>
</table>

**WS-Security error logging**

The glide.processor.debug.SOAPProcessor system property allows error messages about WS-security to be displayed in the transaction log.

The system property glide.processor.debug.SOAPProcessor enables (true) or disables (false) debugging messages for SOAP processing such as certificate and keystore checks.
### glide.processor.debug.SOAPProcessor fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>true</td>
</tr>
<tr>
<td>Default value</td>
<td>false</td>
</tr>
<tr>
<td>Location</td>
<td>Add to the System Properties [sys_properties] table</td>
</tr>
</tbody>
</table>

### WSS X.509 Token Profile

Use the X.509 framework for a WSS X.509 security profile. An X.509 certificate is used to validate a public key that is then used to sign the incoming SOAP message. It specifies a binding between a public key and a set of attributes that includes at least the following:

- subject name
- issuer name
- serial number
- validity interval

Use the X.509 authentication framework as defined by the Web Services Security: SOAP Message Security specification.

Upload the certificate and reference it in the **X509 Certificate** field. If a bound session, select the user to impersonate when the WS-Security authentication succeeds.

### WSS X.509 Security Profile

![WSS X.509 Security Profile](image)

### WSS UsernameToken Profile

When specifying the X.509 Token Profile, you can also supply a UsernameToken in the SOAP request.

A UsernameToken is used as a means of identifying the requester by "user name", and optionally using a password, shared secret, or password equivalent, to authenticate that identity.

There are two ways to authenticate a UsernameToken.
1. Authenticate with existing user credentials.

Authenticate with existing user credentials

Use the user name of the incoming SOAP request to look up a user by the specified User field to match the UserName value. The system uses the password value in the incoming UsernameToken to authenticate the request. When the Bind session option is selected, the user that authenticates successfully is used for the session.

2. Authenticate with specified user credentials.

Authenticate with specified user credentials

Authenticate using login credentials unrelated to users in the User table. When the Bind session option is selected, the user that is specified in the Run as user field is used for the session.

Note: The UsernameToken Profile cannot be used independent of the X.509 Token Profile.

Strict security for web services

By default, basic authentication for web services only determines whether a user is authorized to access the instance with a SOAP connection. Once authorized, any user can access any table published as a web service.

The system property Enforce strict security on incoming SOAP requests changes this behavior and requires that users meet Contextual Security requirements to access instance resources from web services.

With this property enabled, only users that have the proper SOAP role and also meet the ACL conditions the table and operation can perform that operation from a SOAP connection.

Mutual authentication for web services

Mutual authentication is supported for outbound web services.
SOAP session management and reporting

A SOAP session is a Glide session established with an instance by any external SOAP client, such as a web services client application, a ServiceNow MID Server, or the ServiceNow ODBC driver. SOAP sessions are included in the list of user sessions at User Administration > Logged in users. The ?SOAP URLs identify SOAP sessions.

SOAP session properties

Certain properties control how SOAP sessions are maintained.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>glide.soap.invalidate_session_timeout</td>
<td>Duration, in seconds, that an active session remains open. After this duration is reached, the instance deactivates the session and reclaims any system resources. If the client sends another request after the timeout duration is reached, the instance establishes a new session. This property accepts values from 5 to 1200 seconds (20 minutes).</td>
</tr>
</tbody>
</table>

- Type: integer
- Default value: 60
- Location: Add to the System Properties [sys_properties] table

Note: To learn more about properties that affect SOAP web services processing, see the following topics in Instance Security Hardening Settings:

- Access control (instance security hardening)
- Basic auth: SOAP requests

Direct web services

A direct web service is available for any table in the system if the correct access control list is configured.
The supported format of the incoming message is document style literal XML SOAP documents (Document/Literal). To retrieve a direct web service WSDL description and XML schema, point to the relative URL `<tablename>.do?WSDL`. For example, to retrieve the WSDL for the Incident table on the online demo system, use the following URL: https://<instance name>.service-now.com/incident.do?WSDL.

**Extended query parameters**

Extended query parameters enable you to filter and modify the return results of a SOAP query when using the `get`, `getKeys`, and `getRecords` functions.

**Note:** Extended query element names are preceded by two underscore characters.

### Available extended query parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>__encoded_query</code></td>
<td>Specify an encoded query string to be used in filtering the returned results. The encoded query string format is similar to the value that may be specified in a <code>sysparm_query</code> URL parameter. See the encoded query building example in the RSS feed generator examples.</td>
<td><code>&lt;__encoded_query&gt;active=true^category='hardware'&lt;/__encoded_query&gt;</code></td>
</tr>
<tr>
<td><code>__order_by</code></td>
<td>Instruct the returned results to be ordered by the specified field.</td>
<td><code>&lt;__order_by&gt;priority&lt;/__order_by&gt;</code></td>
</tr>
</tbody>
</table>
### Available extended query parameters (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>__order_by_desc</td>
<td>Instruct the returned results to be ordered by the specified field, in descending order.</td>
<td><code>&lt;__order_by_desc&gt;opened_date&lt;/__order_by_desc&gt;</code></td>
</tr>
<tr>
<td>__exclude_columns</td>
<td>Specify a list of comma delimited field names to exclude from the result set.</td>
<td><code>&lt;__exclude_columns&gt;sys_created_on,sys_created_by,caller_id&lt;/__exclude_columns&gt;</code></td>
</tr>
<tr>
<td>__limit</td>
<td>Limit the number of records that are returned.</td>
<td><code>&lt;__limit&gt;100&lt;/__limit&gt;</code></td>
</tr>
<tr>
<td>__first_row</td>
<td>Instruct the results to be offset by this number of records from the beginning of the set. When used with __last_row has the effect of querying for a window of results. The results are inclusive of</td>
<td><code>&lt;__first_row&gt;250&lt;/__first_row&gt;</code></td>
</tr>
</tbody>
</table>
# Available extended query parameters (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>__last_row</td>
<td>Instruct the results to be limited by this number of records from the beginning of the set, or the __start_row value when specified. When used with __first_row has the effect of querying for a window of results. The results are less than the last row number, and does not include the last row.</td>
<td>__last_row&gt;500&lt;/__last_row&gt;</td>
</tr>
<tr>
<td>__use_view</td>
<td>Specify a Form view by name, to be used for limiting and expanding the results returned. When the form view contains deep referenced fields such as __use_view&gt;soap_view&lt;/__use_view&gt;</td>
<td></td>
</tr>
</tbody>
</table>
Available extended query parameters (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>caller_id.email</td>
<td>this field will be returned in the result as well.</td>
<td></td>
</tr>
</tbody>
</table>

**Direct web services namespace**

**Specifying a unique namespace for each table**

The `glide.wsdl.definition.use_unique_namespace` property ensures that each table's direct web service WSDL has a unique `targetNamespace` attribute. This property is `true` by default, which requires a table's direct web service WSDL to use a `targetNamespace` value of `http://www.service-now.com/<table name>`. When `false` (or when the property is not present), all tables use the same `targetNamespace` value of `http://www.service-now.com`. Since all tables also share the same operation names, a web service client attempting to consume more than one ServiceNow web service would be unable to differentiate between requests between multiple tables. Using a unique `targetNamespace` value allows web service clients to distinguish requests between multiple tables.

For example, the direct web service WSDL for the incident table uses this `targetNamespace` value:

```xml
<xsd:schema elementFormDefault = "unqualified" targetNamespace = "http://www.service-now.com/incident" >
```

**Setting namespace requirements**

ServiceNow's WSDL schema by default declares an attribute of `elementFormDefault="unqualified"`. This attribute indicates whether or not locally declared elements must be qualified by the `targetNamespace` in an instance document. If the value of this attribute is `unqualified`, then locally declared elements should not be qualified by the `targetNamespace`. If the value of this
attribute is **qualified**, then locally declared elements must be qualified by the `targetNamespace`.

However, this is incompatible with the way clients generated from WSDL (for example, .NET, Axis2, webMethods) process the embedded schema. It removes the schema namespace as a result, making the web service response unparsable.

To overcome this compatibility issue, a boolean property called `glide.wsdl.schema.UnqualifiedElementFormDefault` is introduced. This property has the value of **true** by default. Setting it to **false** enables clients generated from WSDL to parse the return value of the web service invocation. You can modify this property using the Web Services properties page at **System Properties > Web Services**.

**Allowing duplicate service names**

By default, service names from dynamically generated WSDL are unique and have the following format:

```
ServiceNow_<table name>
```

To allow duplicate service names, administrators can set the `glide.wsdl.unique_service_name` property to **false**. Create the property if it does not exist.

**Related information**

**SOAP web service**

**Use forms to limit or extend the query response**

On occasion, there is a need to limit the number of field values that a SOAP query returns.

You can use a form view to limit the number of field values returned by a SOAP query. Specifying a form view has the effects of:

1. Limiting the response elements to contain only the fields on the view.
2. Specifying reference record field values from referenced fields such as `caller_id.email`. This causes the value of the caller's email to be returned in the SOAP response.

To enable form views for SOAP queries, configure the `com.glidesoapview` property to contain the name of the view that you want to use for all SOAP query responses. The default is `soap_response`. You can also specify the view name as a URL parameter, `sysparm_view=<view name>`, when making the SOAP call. For example:
By default, if a specified view name does not exist, the response contains all fields.

**Return the display value for reference variables**

When you query a record using a `get` or `getRecords` function, the instance returns all fields associated with that record. The fields are often reference fields that contain a sys_id for a record on another table.

Use one of these options if you want the display value for the field to be returned instead of the sys_id:

1. Add the `glide.soap.return_displayValue` property to your system properties, and every SOAP request will return a display value for a reference field.

2. Add the `displayvalue=true` parameter to your SOAP request URL, and SOAP requests with that parameter will return a display value for a reference field as a string, instead of the sys_id. The SOAP URL would look as follows: `https://<instance name>.service-now.com/incident.do?displayvalue=true&SOAP`.

3. Add the `displayvalue=all` parameter to your SOAP request URL, and SOAP requests with that parameter will return a display value for a reference field, in addition to the sys_id. The response element name for the display value field will be prefixed with `dv`, such as `dv_caller_id`.

**Clear values from a target instance**

You can pass an empty value through a SOAP parameter to clear the respective value in the target instance.

You can also pass an empty (null) value through the `&allow_empty_value=true` SOAP query parameter to clear the respective value in the target instance.

For example, `https://<instance name>.service-now.com/incident.do?SOAP&allow_empty_value=true` lets you pass an empty value to the incident record in an instance.

You can then enter lines like the following in the SOAP request:

```xml
<assigned_to>value</assigned_to>
<assignment_group>value</assignment_group>
<category></category>
```

In the above example,
• `<assigned_to>value</assigned_to>` changes the value in the **Assigned to** field to the value specified in the SOAP request.

• `<assignment_group>value</assignment_group>` changes the value in the **Assignment group** field to the value specified in the SOAP request.

• `<category></category>` clears the value in the **Category** field.

**Retrieve journal entries using direct web services**

To get the contents of a journal field, make a second soap request against the `sys_journal_field` table to pull the appropriate journal records back for the record in question.

The URL for the WSDL would be in the following format.

```plaintext
https://instance-name.service-now.com/sys_journal_field.do?WSDL
```

To retrieve the journal entries, you will first need to query the incident for its `sys_id` value, and then supply it as the `element_id` value in a `getRecords` call. To specify records only for the `comments` field, specify the value `comments` for the `element` field. For example, a SOAP request would look like the following:

```xml
<soapenv:Envelope xmlns:soapenv= "http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sys= "http://www.service-now.com/sys_journal_field"
><soapenv:Header />
<soapenv:Body><sys:getRecords><element>comments</element><element_id>9d385017c611228701d22
104cc95c371</element_id></sys:getRecords></soapenv:Body></soapenv:Envelope>
```

**Retrieve choice fields using direct web services**

To retrieve or set choice fields, use the choice **Value**, not the **Label**.

For example, if you want to retrieve a list of all closed incidents, use the numerical value for **Closed**, which is 7 by default.

```xml
<state>7</state>
```

To see a list of choice values:

1. Navigate to the form containing the choice field. For example, navigate to **Incident > Open** and select an incident.

2. Right-click the choice value field and select **Configure Dictionary**. For example, configure the dictionary for the **State** field.

3. From the Choices related list, note the value for the label you want to query. For example, note that the **Closed** choice has a value of 7.
Persist an HTTP session across all SOAP calls

In circumstances when a SOAP client makes many calls in a short amount of time, you may want to re-use a single HTTP session for all SOAP calls.

Each SOAP call creates a new user session that persists until it expires. To create a single user session and re-use it for all inbound SOAP calls, develop your SOAP client following these guidelines:

- Use a module like HTTP::Cookies to create a "cookie jar."
- Save the cookies returned by ServiceNow after each request (handled automatically by HTTP::Cookies).
- Re-send the cookies in the cookie jar with each subsequent request.

⚠️ Note: If you have enabled the session rotation high security setting, it will immediately invalidate the JSESSIONID returned from the server with the first response header. The second response includes a new JSESSIONID.

In perl, you can automatically save and send cookies with the following code:

```perl
use HTTP::Cookies;

# we want to store and re-send cookies
my $cookies = HTTP::Cookies->new(ignore_discard => 1);
```
my $soap = SOAP::Lite->proxy('http://localhost:8080/glide/ecc_queue.do?SOAP');

# Set the cookie jar
$soap->transport->cookie_jar($cookies);

**SOAP direct web service API functions**

The standard SOAP API is a set of small, globally defined functions that can be performed on a targeted resource.

The targeted resource (or table) is defined in the URL by the format https://<instance name>.service-now.com/<table name>.do.

### Data Retrieval API

<table>
<thead>
<tr>
<th>Method Summary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getKeyList</td>
<td>Query the targeted table by example values and return a comma delimited sys_id list.</td>
</tr>
<tr>
<td>getRecords</td>
<td>Query the targeted table by example values and return all matching records and their fields.</td>
</tr>
<tr>
<td>getSysId</td>
<td>Query a single record from the targeted table by sys_id and return the record and its fields.</td>
</tr>
<tr>
<td>aggregate</td>
<td>Query using and aggregate functions SUM, COUNT MIN, MAX, LAST, and AVG. To enable the aggregate functions, activate the Aggregate Web Service Plugin.</td>
</tr>
</tbody>
</table>

### Data Modification API

<table>
<thead>
<tr>
<th>Method Summary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>insert</td>
<td>Creates a new record for the table targeted in the URL.</td>
</tr>
<tr>
<td>insertMultiple</td>
<td>Creates multiple new records for the table targeted in the URL. To enable multiple inserts, activate the Web Service Insert Multiple Plugin.</td>
</tr>
<tr>
<td>update</td>
<td>Updates an existing record in the targeted table in the URL, identified by the mandatory sys_id field.</td>
</tr>
<tr>
<td>deleteRecord</td>
<td>Deletes a record from the targeted table by supplying its sys_id.</td>
</tr>
</tbody>
</table>
Data Modification API (continued)

<table>
<thead>
<tr>
<th>Method Summary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>deleteMultiple</td>
<td>Delete multiple records from the targeted table by example values.</td>
</tr>
</tbody>
</table>

Data Retrieval API

Data Retrieval API method summaries and descriptions.

Data Retrieval API

<table>
<thead>
<tr>
<th>Method Summary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getKeys</td>
<td>Query the targeted table by example values and return a comma delimited sys_id list.</td>
</tr>
<tr>
<td>getRecords</td>
<td>Query the targeted table by example values and return all matching records and their fields.</td>
</tr>
<tr>
<td>get</td>
<td>Query a single record from the targeted table by sys_id and return the record and its fields.</td>
</tr>
<tr>
<td>aggregate</td>
<td>Query using and aggregate functions SUM, COUNT MIN, MAX, LAST, and AVG. To enable the aggregate functions, activate the Aggregate Web Service Plugin.</td>
</tr>
</tbody>
</table>

Related reference

Data Modification API

getKeys
Query the targeted table by example values and return a comma delimited sys_id list.

Input fields
Any field value in the targeted table.

Output fields
A SOAP response element sys_id that contains a comma delimited list of sys_id values.
Sample SOAP messages

Sample SOAP request

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:getKeys>
      <category>hardware</category>
    </inc:getKeys>
  </soapenv:Body>
</soapenv:Envelope>
```

Sample SOAP response

```xml
<soapenv:Envelope xmlns:inc="http://www.service-now.com/incident"
    xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <getKeysResponse>
      <sys_id>46e18c0fa9fe19810066a0083f76bd56,46e57642a9fe198100b96a5dca501ff,46f1784ba9fe19810018aa27fbb23482</sys_id>
      <count>7</count>
    </getKeysResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Language-specific sample messages

For language-specific `getKeys` samples, refer to the following topics:

- Perl SOAP::Lite
- Java Apache Axis2
- Python

`getRecords`

Query the targeted table by example values and return all matching records and their fields.

**Input fields**

Any field value in the targeted table.
Output fields

The getRecordResponse element may contain one or more getRecordsResult elements that encapsulate elements representing the field values of records matching the query.

Sample SOAP messages

Sample SOAP request

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
     xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:getRecords>
      <number>INC0000002</number>
    </inc:getRecords>
  </soapenv:Body>
</soapenv:Envelope>
```

Sample SOAP request using an encoded query to filter where incident number is INC0000001 or INC0000002

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
     xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:getRecords>
      <__encoded_query>number=INC0000001^ORnumber=INC0000002</__encoded_query>
    </inc:getRecords>
  </soapenv:Body>
</soapenv:Envelope>
```

Sample SOAP response that contains 1 record

```xml
<soapenv:Envelope xmlns:inc="http://www.service-now.com/incident"
     xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <getRecordsResponse>
      <getRecordsResult>
        <caller_id>5137153cc611227c000b0b1bd8cd2007</caller_id>
        <caller_id.email>david.loo@service-now.com</caller_id.email>
        <closed_at/>
        <number>INC0000002</number>
        <opened_at>2009-12-14 23:07:12</opened_at>
      </getRecordsResult>
    </getRecordsResponse>
  </soapenv:Body>
</soapenv:Envelope>
```
Sample SOAP response that contains more than 1 record

```xml
<soapenv:Envelope xmlns:inc="http://www.service-now.com/incident"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <getRecordsResponse>
      <getRecordsResult>
        <caller_id>5137153cc611227c000bbd1bd8cd2006</caller_id>
        <caller_id.email>rick.berzle@yourcompany.com</caller_id.email>
        <closed_at>2009-12-17 22:55:16</closed_at>
        <number>INC0000009</number>
        <opened_at>2009-12-16 22:50:23</opened_at>
        <short_description>Reset my password</short_description>
      </getRecordsResult>
      <getRecordsResult>
        <caller_id>5137153cc611227c000bbd1bd8cd2005</caller_id>
        <caller_id.email>fred.luddy@yourcompany.com</caller_id.email>
        <closed_at>2009-12-15 22:54:55</closed_at>
        <number>INC0000010</number>
        <opened_at>2009-12-10 22:53:02</opened_at>
        <short_description>Need Oracle 10GR2 installed</short_description>
      </getRecordsResult>
    </getRecordsResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Language-specific sample messages

For language-specific `getRecords` samples, refer to the following topics:

- Perl SOAP::Lite
- Java Apache Axis2
- Microsoft .NET web services client examples
- Python

© 2021 ServiceNow, Inc. All rights reserved.
**get**

Query a single record from the targeted table by *sys_id* and return the record and its fields.

**Input fields**

An element `<sys_id>` identifying the sys_id of the record to be retrieved.

**Output fields**

A `getResponse` element encapsulating all field values for the record retrieved.

**Sample SOAP messages**

**Sample SOAP request**

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:get>
      <sys_id>46e18c0fa9fe19810066a0083f76bd56</sys_id>
    </inc:get>
  </soapenv:Body>
</soapenv:Envelope>
```

The resulting response of a `get` function call looks like this:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header/>
  <soap:Body>
    <getResponse xmlns="http://www.service-now.com/incident">
      <active>1</active>
      <approval>not requested</approval>
      <assigned_to>46c6f9efa9fe198101dd5eed9ad6e7</assigned_to>
      <caller_id>46b673a6a9fe19810007ab03cbd5849d</caller_id>
      <category>network</category>
      <cmdb_ci>0c43f35dc61122750182c195a29e3243</cmdb_ci>
      <comments>Testing</comments>
      <contact_type>phone</contact_type>
      <due_date>2007-10-28 13:29:45</due_date>
      <escalation>0</escalation>
      <impact>3</impact>
      <incident_state>1</incident_state>
    </getResponse>
  </soap:Body>
</soap:Envelope>
```
aggregate

Query a table using an aggregate function including SUM, COUNT, MIN, MAX, LAST, and AVG.

⚠️ **Note:** Functionality described here requires the Aggregate Web Service plugin.

**Input fields**

Any element of the target table. In addition, one or more of the aggregate functions (SUM, COUNT, MIN, MAX, LAST, and AVG).

A GROUP BY and a HAVING clause may also be added.

**Output fields**

An aggregateResponse element encapsulating all field values for the record retrieved.

**Sample SOAP messages**

Sample SOAP request using COUNT aggregate function.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:m="http://www.service-now.com"
    xmlns:tns="http://www.service-now.com/map"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
```
The resulting response of a COUNT aggregate function call looks like this:

```xml
<?xml version="1.0" encoding="UTF-8"?>
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/
xmlns:m="http://www.service-now.com"
xmlns:tns="http://www.service-now.com/map"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<SOAP-ENV:Body>
<aggregateResponse>
<aggregateResult>
<avg>2.7200</avg>
</aggregateResult>
</aggregateResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Sample SOAP request using AVG aggregate function with a GROUP BY clause.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<aggregate xmlns="http://www.service-now.com">
<GROUP_BY>category</GROUP_BY>
<active>true</active>
<AVG>severity</AVG>
</aggregate>
```
The resulting response of a AVG aggregate function call with a GROUP BY clause looks like this:

```xml
<?xml version="1.0" encoding="UTF-8"?
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:m="http://www.service-now.com"
xmlns:tns="http://www.service-now.com/map"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <aggregateResponse>
      <aggregateResult>
        <avg>1.0000</avg>
        <category>database</category>
      </aggregateResult>
      <aggregateResult>
        <avg>3.0000</avg>
        <category>hardware</category>
      </aggregateResult>
      <aggregateResult>
        <avg>3.0000</avg>
        <category>inquiry</category>
      </aggregateResult>
      <aggregateResult>
        <avg>2.0000</avg>
        <category>network</category>
      </aggregateResult>
      <aggregateResult>
        <avg>2.6923</avg>
        <category>software</category>
      </aggregateResult>
    </aggregateResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Sample SOAP request using an encoded query to filter the aggregate:

```xml
<?xml version="1.0" encoding="UTF-8"?
<SOAP-ENV:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Body>
    <aggregateResponse>
      <aggregateResult>
        <avg>1.0000</avg>
        <category>database</category>
      </aggregateResult>
      <aggregateResult>
        <avg>3.0000</avg>
        <category>hardware</category>
      </aggregateResult>
      <aggregateResult>
        <avg>3.0000</avg>
        <category>inquiry</category>
      </aggregateResult>
      <aggregateResult>
        <avg>2.0000</avg>
        <category>network</category>
      </aggregateResult>
      <aggregateResult>
        <avg>2.6923</avg>
        <category>software</category>
      </aggregateResult>
    </aggregateResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```
Sample aggregate request using HAVING to narrow the results.

HAVING takes four fields. Each field is delimited by "^": the aggregate type, the field of the aggregate, the operation type, and the value to compare.

More than one HAVING can be added to the request, so you can use HAVING expressions, but there is no support for OR.
## Data Modification API

<table>
<thead>
<tr>
<th>Method Summary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>insert</td>
<td>Creates a new record for the table targeted in the URL.</td>
</tr>
<tr>
<td>insertMultiple</td>
<td>Creates multiple new records for the table targeted in the URL. To enable multiple inserts, activate the Web Service Insert Multiple Plugin.</td>
</tr>
<tr>
<td>update</td>
<td>Updates an existing record in the targeted table in the URL, identified by the mandatory <code>sys_id</code> field.</td>
</tr>
<tr>
<td>deleteRecord</td>
<td>Deletes a record from the targeted table by supplying its <code>sys_id</code>.</td>
</tr>
<tr>
<td>deleteMultiple</td>
<td>Delete multiple records from the targeted table by example values.</td>
</tr>
</tbody>
</table>

### Related reference

#### Data Retrieval API

**insert**

Creates a new record for the table targeted in the URL.

### Input fields

All fields from the targeted table, excluding system fields. Fields configured as mandatory in the System Dictionary are reflected in the WSDL with the attribute `minOccurs=1`.

### Output fields

**Insert method output fields**

<table>
<thead>
<tr>
<th>Table type</th>
<th>Output fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular</td>
<td>The <code>sys_id</code> field and the display value of the target table (<code>table</code>) are returned.</td>
</tr>
<tr>
<td>Import set</td>
<td>The <code>sys_id</code> of the import set row, the name of the transformed target table (<code>table</code>), the <code>display_name</code> for the transformed target table, the <code>display_value</code> of the transformed target row, and a <code>status</code> field, which can contain <code>inserted</code>, <code>updated</code>, or <code>error</code>. There can be an optional <code>status_message</code> field or an <code>error_message</code> field value when <code>status=error</code>.</td>
</tr>
</tbody>
</table>
## Insert method output fields (continued)

<table>
<thead>
<tr>
<th>Table type</th>
<th>Output fields</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>When an insert did not cause a target row to be transformed (skipped because a key value is not specified), the sys_id field will contain the sys_id of the import set row, rather than the targeted transform table.</td>
</tr>
<tr>
<td>Import set with multiple transforms</td>
<td>The response from this type of insert will contain multiple sets of fields from the regular import set table insert wrapped in a multiInsertResponse parent element. Each set will contain a map field, showing which transform map created the response.</td>
</tr>
</tbody>
</table>

### Sample SOAP messages for a regular table

The following example shows an insert that specifies the short description only:

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/
 xmlns:m="http://www.service-now.com"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
 <SOAP-ENV:Body>
  <insert xmlns="http://www.service-now.com">
   <short_description xsi:type="xsd:string">This is a test</short_description>
  </insert>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The resulting response looks like this:

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/
 xmlns:m="http://www.service-now.com"
 xmlns:tns="http://www.service-now.com/incident"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
 <SOAP-ENV:Body>
  <insertResponse xmlns="http://www.service-now.com">
  </insertResponse>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```
Language-specific sample messages

For language-specific insert samples, refer to the following topics:

- Perl SOAP::Lite
- Java Apache Axis2
- Microsoft .NET
- Python

**insertMultiple**

Creates multiple new records for the table targeted in the URL.

**Input fields**

The `insertMultiple` element may contain 1 or more record tags that contains all fields from the targeted table, excluding system fields. Limit the number of records inserted in a single operation to no more than 200. You can gradually increase this number with subsequent exports if the increase does not negatively impact instance performance.

**Output fields**

The `insertMultipleResponse` tag is followed by 1 or more record tags that contains:

**Insert method output fields**

<table>
<thead>
<tr>
<th>Table type</th>
<th>Output fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular</td>
<td>The <code>sys_id</code> field and the display value of the target table (<code>table</code>) are returned.</td>
</tr>
<tr>
<td>Import set</td>
<td>The <code>sys_id</code> of the import set row, the name of the transformed target table (<code>table</code>), the <code>display_name</code> for the transformed target table, the <code>display_value</code> of the transformed target row, and a <code>status</code> field, which can contain <code>inserted</code>, <code>updated</code>, or <code>error</code>. There can be an optional <code>status_message</code> field or an <code>error_message</code> field value when <code>status=error</code>.</td>
</tr>
</tbody>
</table>
Insert method output fields (continued)

<table>
<thead>
<tr>
<th>Table type</th>
<th>Output fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>When an insert did not cause a target row to be transformed (skipped because a key value is not specified), the <strong>sys_id</strong> field will contain the sys_id of the import set row, rather than the targeted transform table.</td>
<td></td>
</tr>
<tr>
<td>Import set with multiple transforms</td>
<td>The response from this type of insert will contain multiple sets of fields from the regular import set table insert wrapped in a <strong>multiInsertResponse</strong> parent element. Each set will contain a <strong>map</strong> field, showing which transform map created the response.</td>
</tr>
</tbody>
</table>

Sample SOAP messages for a regular table

The following example shows an insert that specifies the short description only:

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:insertMultiple>
      <record>
        <short_description>this is test 1</short_description>
      </record>
      <record>
        <short_description>this is test 2</short_description>
      </record>
      <record>
        <short_description>this is test 3</short_description>
      </record>
    </inc:insertMultiple>
  </soapenv:Body>
</soapenv:Envelope>
```

The resulting response looks like this:

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <insertMultipleResponse>
      <insertResponse>
        <sys_id>168160ad4a36231200a89091281dc803</sys_id>
      </insertResponse>
    </insertMultipleResponse>
  </soapenv:Body>
</soapenv:Envelope>
```
Sample SOAP messages for an import set table

The following example shows an insert that specifies the short description only:

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/
xmlns:imp="http://www.service-now.com/imp_notification">
  <soapenv:Header/>
  <soapenv:Body>
    <imp:insertMultiple>:
      <imp:record>
        <imp:message>one</imp:message>
        <imp:uuid>a</imp:uuid>
      </imp:record>
      <imp:record>
        <imp:message>two</imp:message>
        <imp:uuid>b</imp:uuid>
      </imp:record>
      <imp:record>
        <imp:message>three</imp:message>
        <imp:uuid>c</imp:uuid>
      </imp:record>
    </imp:insertMultiple>
  </soapenv:Body>
</soapenv:Envelope>
```

The resulting response looks like this:

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/
xmlns:imp="http://www.service-now.com/imp_notification">
  <soapenv:Header/>
  <soapenv:Body>
  </soapenv:Body>
</soapenv:Envelope>
```
update

Updates an existing record in the targeted table in the URL, identified by the mandatory sys_id field.

Input fields

All fields from the targeted table, excluding system fields, which will be used for updating the existing record. The sys_id field is used to locate the existing record.

Output fields

Returns the sys_id of the record that was updated.
Sample SOAP messages

Sample SOAP request

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:update>
      <sys_id>46e18c0fa9fe19810066a0083f76bd56</sys_id>
      <short_description>this is updated</short_description>
    </inc:update>
  </soapenv:Body>
</soapenv:Envelope>
```

Sample SOAP response

```xml
<soapenv:Envelope xmlns:inc="http://www.service-now.com/incident"
    xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <updateResponse>
      <sys_id>46e18c0fa9fe19810066a0083f76bd56</sys_id>
    </updateResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Language-specific sample messages

For language-specific update samples, refer to the following topics:

- **Perl SOAP::Lite**
- **Java Apache Axis2**
- **Microsoft .NET**
- **Python**

**deleteRecord**
Delete a record from the targeted table by supplying its `sys_id`.

**Input fields**
An element `<sys_id>` identifying the `sys_id` of the record to be retrieved.
Output fields

A `<count>` element within the `deleteRecordResponse` parent element indicating the number of records deleted, this will always equal to "1" for `deleteRecord`.

Sample SOAP messages

Sample SOAP request

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:deleteRecord>
      <sys_id>46e18c0fa9fe19810066a0083f76bd56</sys_id>
    </inc:deleteRecord>
  </soapenv:Body>
</soapenv:Envelope>
```

Sample SOAP response

```xml
<soapenv:Envelope xmlns:inc="http://www.service-now.com/incident"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <deleteRecordResponse>
      <count>1</count>
    </deleteRecordResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Language-specific sample messages

For language-specific `deleteRecord` samples, refer to the following topics:

- Perl SOAP::Lite
- Java Apache Axis2
- Microsoft .NET
- Python

**deleteMultiple**

Delete multiple records from the targeted table by example values.
Input fields
All fields from the targeted table, including system fields, are used in query-by-example (QBE) fashion to locate records to be deleted. Query example fields can have special prefixes to constrain the search function.

Output fields
A `<count>` element within the `deleteRecordResponse` parent element indicating the number of records deleted.

Sample SOAP messages

Sample SOAP request
```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"

xmlns:inc="http://www.service-now.com/incident">

<soapenv:Header/>

<soapenv:Body>

<inc:deleteMultiple>

  <category>hardware</category>

</inc:deleteMultiple>

</soapenv:Body>

</soapenv:Envelope>
```

Sample SOAP response
```
<soapenv:Envelope xmlns:inc="http://www.service-now.com/incident"

xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">

<soapenv:Header/>

<soapenv:Body>

<deleteMultipleResponse>

  <count>6</count>

</deleteMultipleResponse>

</soapenv:Body>

</soapenv:Envelope>
```

Language-specific sample messages
For language-specific `deleteRecord` samples, refer to the following topics:

Perl SOAP::Lite
Java Apache Axis2
Microsoft .NET
Python
SOAP web service import sets

Web service import sets complement direct web services and scripted SOAP web services by providing a web service interface to import sets tables.

By default, this type of web service synchronously transforms the incoming data based on the associated transform maps. If the associated import set mode is set to Asynchronous, the behavior is to save the data for transformation at a later time. Web service import sets tables publish all default web service functions in the WSDL.

You can access a web service import set WSDL by specifying the import set table name + ".do?WSDL" on the URL.

For example:

http://<instance name>.service-now.com/imp_notification.do?WSDL.

Related information
- Web service import sets
- Import sets key concepts

Scripted SOAP web services

Scripted SOAP web services allow a ServiceNow administrator to create custom SOAP web services.

You can define input and output parameters for the SOAP web service and use JavaScript to perform operations. Though this feature is very powerful, use direct web services or SOAP web service import sets whenever possible since they are simpler to implement and maintain.

Security

Scripted SOAP web services have the same base security options as all SOAP web services. For details on SOAP web services security, see SOAP web services security.

When strict security is enforced on a system, the HTTP authenticated user must have the soap_script role to execute the scripted web service.

WSDL

All ServiceNow tables and import sets dynamically generate Web Service Definition Language (WSDL) XML documents that describe its table schema and available operations.
Enforcing WSDL compliance

You can force the response to list output values in the same order as defined in the WSDL.

When you create a scripted SOAP web service, the generated WSDL is based on the Input Parameters and Output Parameters related lists. The actual SOAP response sent by the scripted service is determined by the Script. This behavior can cause the script to return output values in a different order than defined in the WSDL.

To enforce the order of output parameters as defined in the related list, select the WSDL Compliance check box. When this check box is selected, the web service reorders the parameters returned by the script to match the order in the WSDL.

**Note:** If additional response parameters are returned by the script, but are not defined in the Response Parameters related list, those parameters are excluded from the response when WSDL Compliance is selected.

### Output Parameters related list

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Param 1</td>
<td>200</td>
</tr>
<tr>
<td>Param 2</td>
<td>300</td>
</tr>
<tr>
<td>Param 3</td>
<td>100</td>
</tr>
</tbody>
</table>

The following is the script that sets values for the defined output parameters. Note that in this example script the parameters are set in a different order than defined in the Output Parameters related list. Also note the additional parameter `param4` that is not defined in the related list.

```java
Response.param1 = 1;
Response.param4 = 4;
Response.param3 = 3;
```

When the WSDL Compliance check box is false, the SOAP response generated by the script is the following:

```xml
<response>
  <param1>1</param1>
  <param4>4</param1>
  <param3>3</param1>
</response>
```
When the **WSDL Compliance** check box is **true**, the SOAP response generated by the script is the following:

```xml
<response>
  <param3>3</param3>
  <param1>1</param1>
</response>
```

**Static WSDL**

Some web service clients require SOAP access to your instance through a specific WSDL format. This required format may differ from the standard ServiceNow WSDL format. In these cases you can create a static WSDL that matches the required format. For details on creating a static WSDL, see **Global variables**

To facilitate custom processing of incoming SOAP requests, the following global variables are available in the script context:

- `soapRequestDocument`: Java org.w3c.dom.Document object representing the incoming SOAP envelope.
- `soapRequestXML`: String object representing the incoming SOAP envelope XML.
- `request`: Javascript object that contains mapped values (mapped to input parameter names) of the incoming SOAP envelope.
- `response`: Javascript object that allows you to customize the response values. See **Customize Response**

**Create a new scripted SOAP web service**

Follow these examples to create a new scripted SOAP web service.

When the Web Services Provider - Scripted plugin is activated, a new module **Scripted Web Services** is available under the **System Web Services** application.
**Example 1: Retrieving a system property**

The first step is to define the incoming and return parameters. This is done by adding an entry to the *Input Parameters* and *Output Parameters*. These parameters are used to construct and present a meaningful WSDL, and they do not add to the functionality of processing the actual Web Service itself.

The parameters are referenced in the script of the Web Service. Any of the input parameters are retrieved using the following syntax:

```javascript
var a = request.property;
```

The output parameters are set by using the following syntax:

```javascript
response.property = "ABC";
```
The following example demonstrates how to retrieve a system property and return it as part of the SOAP response. The example shows how to create a custom scripted web service to do something specific that the base ServiceNow system direct Web Services cannot.

GetProperty web service

Example 2: Ordering a Blackberry

Direct web services operate on tables and their data. The following example shows how to initiate a business solution, such as ordering a Blackberry, by invoking a scripted web service. The following input and output parameters support the Blackberry example:
This script shows how to use the above parameters to add a Blackberry to the service catalog shopping cart and order it. The request number is returned in the `request_number` field of the SOAP response.

```javascript
var cart = new Cart();
var item = cart.addItem('e2132865c0a8016500108d9cee411699');
cart.setVariable(item, 'original', request.phone_number);

// set the requested for
var gr = new GlideRecord("sys_user");
gr.addQuery("user_name", request.requested_for);
gr.query();
if (gr.next()) {
    var cartGR = cart.getCart();
cartGR.requested_for = gr.sys_id;
cartGR.update();
}

var rc = cart.placeOrder();
response.request_number = rc.number;
```

**Customize response**

Follow this example to customize and control the XML payload of a SOAP response.

**Procedure**

1. Create a customized XML document using the `XMLDocument` script include object.

   **Note:** When creating a scripted web service in a scoped application you must use the `XMLDocument2` API.
2. Set its document element to the variable `response.soapResponseElement` in a scripted web service.
For example, the following scripted web service script:

```javascript
var xmldoc = new XMLDocument2();
xmldoc.parseXML("<myResponse></myResponse>");
xmldoc.createElementWithTextValue("element_one", "test");
xmldoc.createElementWithTextValue("element_two", "new2 value");

var el = xmldoc.createElement("element_three");
xmldoc.setCurrentElement(el);
xmldoc.createElementWithTextValue("newChild", "test child element");

response.soapResponseElement = xmldoc.getDocumentElement();
```

Is used to accept the following request:

```xml
<soapenv:Envelope
    xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:tes="http://www.service-now.com/TestCustomResponse">
    <soapenv:Header/>
    <soapenv:Body>
        <tes:execute/>
    </soapenv:Body>
</soapenv:Envelope>
```

Which will respond with the following SOAP response:

```xml
<soapenv:Envelope
    xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:tes="http://www.service-now.com/TestCustomResponse">
    <soapenv:Header/>
    <soapenv:Body>
        <myResponse>
            <element_one>test</element_one>
            <element_two>new2 value</element_two>
            <element_three>
                <newChild>test child element</newChild>
            </element_three>
        </myResponse>
    </soapenv:Body>
</soapenv:Envelope>

WSDL support will need to be created externally. The SOAP endpoint will need to be referred back to the scripted web service in question.
Create a scripted SOAP web service using a static WSDL

Follow these examples to create a scripted SOAP web service using a static WSDL.

Create a scripted web service using a static WSDL

To use a static WSDL, create a scripted web service.

Before you begin
Role required: web_service_admin or admin

Procedure
1. Navigate to System Web Service > Scripted Web Services.
2. Click New.
3. Enter a Name for the scripted SOAP web service such as FakeStockValue.
4. Enter a Script for the web service to run.
5. Click Submit.

Scripted web service example

This example demonstrates the processing script for the FakeStockValue web service.

```
var vProcessor = new FakeStockValue(soapRequestXML);

var responseElement = vProcessor.process();
if(responseElement != null) {
    response.soapResponseElement = responseElement;
} else {
    response.soapResponseElement = vProcessor.generateSoapFault("unknown error");
}
```

Create a static WSDL

Create a static WSDL with the required format to override the standard WSDL for your scripted web service.

Before you begin
Role required: web_service_admin or admin

Procedure
1. Navigate to System Web Services > Static WSDL.
2. Create a static WSDL record using the same name as the scripted web service, such as FakeStockValue.
3. Enter the custom WSDL into the WSDL field.
4. Click Submit.

Static WSDL example
This example demonstrates the FakeStockValue WSDL.

```xml

<types><schema targetNamespace = "http://example.com/stockquote.xsd" xmlns = "http://www.w3.org/2000/10/XMLSchema" ><element name = "TradePriceRequest" ><complexType><all><element name = "tickerSymbol" type = "string" /></all></complexType></element><element name = "TradePrice" ><complexType><all><element name = "price" type = "float" /></all></complexType></element></schema></types>

<message name = "GetLastTradePriceInput" ><part name = "body" element = "xsd1:TradePriceRequest" /></message>

<message name = "GetLastTradePriceOutput" ><part name = "body" element = "xsd1:TradePrice" /></message>

<portType name = "StockQuotePortType" ><operation name = "GetLastTradePrice" ><input message = "tns:GetLastTradePriceInput" /><output message = "tns:GetLastTradePriceOutput" /></operation></portType>

<binding name = "StockQuoteSoapBinding" type = "tns:StockQuotePortType" ><soap:binding style = "document" transport = "http://schemas.xmlsoap.org/soap/http" /><operation name = "GetLastTradePrice" ><soap:operation soapAction = "" /><input><soap:body use = "literal" /></input><output><soap:body use = "literal" /></output></operation></binding>

<service name = "StockQuoteService" ><documentation>My first service</documentation><port name = "StockQuotePort" binding = "tns:StockQuoteSoapBinding" ><soap:address location = "https://myinstance.service-now.com/FakeStockValue.do?SOAP" /></port></service>

</definitions>
```
Create a static WSDL script include
Create a script include to define the majority of the code used to process static WSDL requests.

Before you begin
Role required: script_include_admin or admin

About this task
By implementing the majority of the custom functionality in a script include, you can reuse the script include in multiple areas.

Procedure
1. Navigate to System UI > Script Includes.
2. Click New.
3. Enter a Name for the script include that matches the name of the static WSDL, such as FakeStockValue.
4. Enter the script include code in the Script field.
5. Click Submit.

Static WSDL script include example
This example demonstrates the FakeStockValue script include that implements much of the static WSDL behavior.

```javascript
var FakeStockValue = Class.create();

FakeStockValue.prototype = { 
    initialize : function(requestXML) { 
        //Use some backend XML utilities...you could use string tools if you wish 
        this.xmlutil = Packages.com.glide.util.XMLUtil; 
        //converting the string to an XML Document 
        this.fSoapDoc = new XMLDocument(requestXML); },

    process : function() { 
        var soapBody = this.fSoapDoc.getNode("/Envelope/Body"); 
        //Our WSDL was formatted to have the only first child element be the function 
        var funcNode = this.xmlutil.getFirstChildElement(soapBody); 
        var nodeName = this.xmlutil.getNodeNameNS(funcNode); 
```
// If the function for this SOAP request is TradePriceRequest, perform the necessary actions
if (nodeName == "TradePriceRequest") {
    return this.fakeOutTradePriceRequest(funcNode);
}

// Couldn't find any supported functions in this SOAP request
return this.generateSoapFault("un-supported API call: " + nodeName);

fakeOutTradePriceRequest : function (funcNode) {

    // Create the beginnings of our XML response
    var r = new XMLDocument("<GetLastTradePriceOutput
xmlns='https://www.service-now.com/vws/FakeStockValue'/>");

    // Do the necessary actions here... we're going to get the USER ID of the user
    // used to make this SOAP call. Then we will return the stock symbol they were asking about
    var usersysid = gs.getUserID();
    var now_GR = new GlideRecord("sys_user");
    gr.get(usersysid);
    var username = gr.user_name;
    var quoteSymbol = this.xmlutil.getText(funcNode);
    // Create a "message" element to store our response message
    r.createElement("message", username + ", You were looking for a quote on " + quoteSymbol);
    return r.getDocumentElement();

}

generateSoapFault : function (str) {
    var f = "<SOAP-ENV:Fault>" +
             "<faultcode xsi:type='xsd:string'>SOAP-ENV:FakeStockValue</faultcode>" +
             "<faultstring xsi:type='xsd:string'>" + str +
             "</faultstring>" +
             "</SOAP-ENV:Fault>"
    var s = new XMLDocument(f);
    return s.getDocumentElement();
}
}
initialize function
The initialize function takes the XML request string and converts it to an XML Document object that you can navigate and manipulate using libraries. Alternatively, you can leave the XML request as a string and navigate it using regular expressions.

process function
The process function is called by the scripted web service. This function grabs the first child element in the XML after the body element. The WSDL uses this child element to determine which function to use. In this WSDL there is only one possible function but most WSDLs provide many functions. If more functions were available, there would be more "if" statements that tested the first child element for the various function names.

fakeOutTradePriceRequest function
The fakeOutTradePriceRequest function is the implementation of the only available function in the WSDL. This function looks up the user that the SOAP request authenticated as and retrieves the user_name then returns it to the SOAP client. The fakeOutTradePriceRequest function could be expanded to perform useful activities, such as looking up a stock symbol and returning the last traded price.

generateSoapFault function
The generateSoapFault function returns a SOAP error that can be called if there are problems.

Use the static WSDL
Load the static WSDL into a SOAP client to make requests to the SOAP web service.

The web service client provides
• The FakeStockValue project.
• The StockQuoteBinding web service.
• The GetLastTradePrice SOAP function. This function generates request records when run.
You can change the default request XML in the static WSDL to include a stock symbol.

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:stoc="http://example.com/stockquote.xsd">
  <soapenv:Header/>
  <soapenv:Body>
    <stoc:TradePriceRequest>IBM</stoc:TradePriceRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Submitting a SOAP request to this web service endpoint returns the following to the requesting SOAP client.

```xml
                    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
    <GetLastTradePriceOutput xmlns="https://www.service-now.com/vws/FakeStockValue">
      <message>admin2, You were looking for a quote on IBM</message>
    </GetLastTradePriceOutput>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

**AttachmentCreator SOAP web service**

Attach documents to records in ServiceNow by sending a SOAP message targeting the `ecc_queue` table.

**Important:** The AttachmentCreator SOAP web service is not recommended. Instead, use the REST Attachment API to manage attachments with web services.

Using the AttachmentCreator SOAP web service, you can attach a single document to a message that is a maximum of 5 MB. The following is an example of a URL or target end point: https://instance_name.service-now.com/ecc_queue.do?WSDL

**ecc_queue Fields for Attachment Creation**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent</td>
<td>The name of the agent sending in the request, this</td>
<td>AttachmentCreator</td>
</tr>
</tbody>
</table>
### ecc_queue Fields for Attachment Creation (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>can be any value since it is not used for processing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>topic</td>
<td>The topic of the queue record, this value must be set to &quot;AttachmentCreator&quot; to trigger the attachment creation</td>
<td>AttachmentCreator</td>
</tr>
<tr>
<td>name</td>
<td>This field must contain a &quot;:&quot; delimited value of the file name, and its content-type</td>
<td>file_name.xls:application/vnd.ms-excel</td>
</tr>
<tr>
<td>source</td>
<td>This field must contain a &quot;:&quot; delimited value of the target table and its sys_id</td>
<td>incident:dd90c5d70a0a0b39000aac5ae6704ae8</td>
</tr>
<tr>
<td>payload</td>
<td>This field must contain the Base 64 encoded string representing the object to be attached</td>
<td>the base 64 encoded string</td>
</tr>
</tbody>
</table>

Sending in the values described in the table above will attach an Excel file to the incident table for the record located by the sys_id `dd90c5d70a0a0b39000aac5ae6704ae8`

### Security

Like all other HTTP based web services available on the platform, the AttachmentCreator SOAP web service is required to authenticate using basic authentication by default. The user ID that is used for authentication will be subjected to access control in the same way as an interactive user.

To create attachments, the SOAP user must have any roles required to create Attachment [sys_attachment] records, as well as the soap_create role, and
any roles required to read and write records on the target table, such as the itil role to add attachments to incident records. By default there is no single role allowing you to add attachments. You can create a role to explicitly allow adding attachments, then assign this role to the SOAP user.

File type security
You can control what file types users can attach by setting the glide.attachment.extensions and glide.security.file.mime_type.validation properties.

For these properties to apply to the AttachmentCreator web service, the property glide.attachment.enforce_security_validation must be set to true. This property is true by default.

Example SOAP Message
The following is an example of a SOAP message that would take a text file “john1.txt” of mime-type: text/plain and attach it to an Incident with a GUID of: e6eed6950a0a3c59006f32c8e3ff3cf9. Note the payload is the base64 encoding of the file itself.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ecc="http://www.service-now.com/ecc_queue">
  <soapenv:Header />
  <soapenv:Body>
    <ecc:insert>
      <agent>AttachmentCreator</agent>
      <topic>AttachmentCreator</topic>
      <name>john1.txt:text/plain</name>
      <source>incident:e6eed6950a0a3c59006f32c8e3ff3cf9</source>
      <payload>SSB3b25kZXIgaWYgc2hlIGtub3ducyB3aGF0IHNoZSdzIGRvaW5nIG5vdy4K</payload>
    </ecc:insert>
  </soapenv:Body>
</soapenv:Envelope>
```

Example Node.js Script
The following example Node.js script adds an attachment to an incident record. Run this script from a client computer, not an instance.

```javascript
/**
 * Node.js to ServiceNow attachment upload via SOAP
 * 
 * Andrew Venables andrew.venables@servicenow.com
 * 
 * July 2014
 */
```
var soap = require('soap'), // https://github.com/vpulim/node-soap
mime = require('mime'), // https://github.com/broofa/node-mime
fs = require("fs");

var WSDL_FILENAME = 'ecc_queue.xml'; // Goto https://instancename.service-now.com/ecc_queue.do?WSDL and save a copy of the WSDL locally for simplicity
var DIRECTORY_CONTAINING_FILES = '/Users/andrew.venables/Documents/Uploads'; // Local path to the directory containing all the files we want to upload
var USERNAME = 'andy.venables'; // An admin user account on the instance
var PASSWORD = 'MY_PASSWORD'; // Password for above account
var TARGET_TABLE = 'incident'; // Target table to attach the files to
var TARGET_SYS_ID = '9d385017c611228701d22104cc95c371'; // Target record / sys_id to attach the files to. OOTB INC0000002

var files_to_upload; // Global that will contain our list of files to be uploaded
var pos = 0; // Global pointer for our position in the files_to_upload list

// Create a SOAP client to use to post to the instance
soap.createClient(WSDL_FILENAME, function(err, client) { // Node uses callbacks
if (err) console.error(err);

// Set the username and password
client.setSecurity(new soap.BasicAuthSecurity(USERNAME, PASSWORD));

// Read all the files in our source directory, will include . and ..
files_to_upload = fs.readdirSync(DIRECTORY_CONTAINING_FILES);

console.log('Files to upload: ' + files_to_upload.length + '\n');

// Start iterating through the list of files to upload
next(client);
});

// Process the next file in the files_to_upload array
// This is a neat way of dealing with Node and its expectation of callbacks
function next(client) {

// Check we haven't reached the end
if (pos >= files_to_upload.length) return;

// Get the next file to upload
var file_name = files_to_upload[pos];

// Increment the pointer to the next file
pos++;

console.log(pos + '/' + files_to_upload.length + ' - Uploading file: ' + file_name);

// A blank file is the end of the list
if (file_name == '') return;

// Skip to the next file as this one is invalid
if (file_name == '.' || file_name == '..' || file_name.indexOf('.') == 0)
  next(client);

// Get the file type using a module called mime
var file_type = mime.lookup(file_name);
console.log(' of type: ' + file_type);

var file_payload;
// Load the file into a buffer
fs.readFile(DIRECTORY_CONTAINING_FILES + '/' + file_name, function(err, the_data) {
  if (err) console.error(err);

  // Encode the buffer to base64
  file_payload = new Buffer(the_data, 'binary').toString('base64');

  // Set the parameters before we call the Web Service
  var parameters = {
    'agent': 'AttachmentCreator',
    'topic': 'AttachmentCreator',
    'name': file_name + ':+' + file_type,
    'source': TARGET_TABLE + ':TARGET_SYS_ID,'
    'payload': file_payload
  };

  console.log('      sending...')
  // Make the Web Service call, remember node likes callbacks
  client.insert(parameters, function(err, result) {
    if (err) console.error(err);
  });
Example Perl Script
The following example Perl script will create an attachment to an incident record.

```perl
use MIME::Base64;
use strict;
use SOAP::Lite;

# the ServiceNow instance
my $SNC_HOST = "https://instance_name.service-now.com";
my $base64;
my $buf;

# upload and attach a file on the local disk, base 64 encode it into a string first
open(FILE, "'/Users/davidloo/Desktop/test_files/number_test.xls") or die "$!";
binmode FILE; #preserves file formatting on Windows
while (read(FILE, $buf, 60*57)) {
    $base64 .= encode_base64($buf);
}

# call the sub routine to attach
attach_incident();

sub attach_incident {
    # target the ecc_queue table
    my $soap = SOAP::Lite->proxy("$SNC_HOST/ecc_queue.do?SOAP");
    $soap->(_transport)->(_proxy)->(ssl_opts)->(verify_hostname) = 0;
    my $method = SOAP::Data->name('insert')->attr({xmlns => 'http://www.service-now.com/'});

    # set the ecc_queue parameters
    my @params = (SOAP::Data->name(agent => 'AttachmentCreator'));
    push(@params, SOAP::Data->name(topic => 'AttachmentCreator') );

    # identify the file name and its mime type
    push(@params, SOAP::Data->name(name => 'number_test.xls:application/vnd.ms-excel') );
}
```
# attach to the incident, specifying its sys_id
push(@params, SOAP::Data->name(source => 'incident:dd90c5d70a0a0b39000aac5ae704ae8'));

# set the payload to be the base 64 encoded string representation of the file
push(@params, SOAP::Data->name(payload => $base64));

# invoke the web service
my $result = $soap->call($method => @params);

print_fault($result);

print_result($result);

sub print_result {
    my ($result) = @_; 

    if ($result->body && $result->body->{'insertResponse'}) {
        my %keyHash = %{ $result->body->{'insertResponse'} }; 
        foreach my $k (keys %keyHash) {
            print "name=$k   value=$keyHash{$k}\n";
        }
    }
}

sub print_fault {
    my ($result) = @_; 

    if ($result->fault) {
        print "faultcode=" . $result->fault->{'faultcode'} . "\n";
        print "faultstring=" . $result->fault->{'faultstring'} . "\n";
        print "detail=" . $result->fault->{'detail'} . "\n";
    }
}

# use the itil user for basic auth credentials
sub SOAP::Transport::HTTP::Client::get_basic_credentials {
    return 'itil' => 'itil';
}
Override a SOAP endpoint

The SOAP endpoint address where the SOAP message is posted is consistent with the endpoint of the WSDL.

In some cases, however, the WSDL may reference an incorrect endpoint URL. If this happens, it is necessary to over-ride the generated URL by creating the property `com.glide.soap_address_base_url` to contain the new URL. You may have to add the property manually as this is not an base system property.

For instance, a generated SOAP endpoint may look like this:

```
https://instance.service-now.com/incident.do?SOAP
```

You can specify a property to define the endpoint such that it goes through a proxy by setting the property:

```
com.glide.soap_address_base_url = "https://myproxy.mycompany.com/service-now/"
```

This will result in the endpoint being generated to appear as:

```
https://myproxy.mycompany.com/service-now/incident.do?SOAP
```

Enable HTTP compression

By default, the SOAP request is accepted un-compressed and the result of the request is returned un-compressed.

To enable HTTP compression using [gzip] when sending in your SOAP request, set the following HTTP header:

```
Content-Encoding: gzip
```

To receive the SOAP response compressed using [gzip] send in your SOAP request with the following HTTP header:

```
Accept-Encoding: gzip
```

Prevent empty elements in SOAP messages

By default, an instance does not omit empty elements, elements with NULL or NIL values, from SOAP messages.

To prevent SOAP responses from containing empty elements, an administrator can create a system property called `glide.soap.omit_null_values` and set the value to `true`. This behavior is compliant with the WSDL as all elements in a SOAP message have a minOccurs=0 attribute and are therefore optional. In addition, this behavior prevents the instance from creating inefficient SOAP messages containing a large number of empty elements.
Set this property to `false` to allow SOAP messages to search for existing fields with empty values. For example, if an administrator wants to search for incidents with an empty `Assigned to` field from a SOAP message, then the SOAP message must be able to send an empty value for this field.

**Note:** Changing the value of this property may cause unintended actions in existing web service integrations. Administrators are strongly encouraged to carefully test the new behavior to ensure that existing integrations process empty elements as expected.

**Insert related records using SOAP**

Support is available for inserting hierarchical data into tables or web service import set tables. The hierarchical data in the Insert API is automatically mapped to related records of the targeted table.

**Before you begin**

Role required: admin

**Procedure**

Create and set the property `glide.web_service.hierarchical` to `true`. The client of the API can also override this value by invoking the SOAP web service with the URL parameter `hierarchical=true`.

**Example:**

For example, when a related list is created for the incident table called `u_custom_comments`:

And `u_comment_items` is created as a related list for `u_custom_comments`:

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Related reference

WSDL Schema with related records
Hierarchical SOAP Message

WSDL Schema with related records

When a WSDL for the target Incident table is requested with an additional parameter of \texttt{hierarchical=true}, the WSDL schema for the \texttt{Insert} function will reflect available related records that may participate in the hierarchical data payload.

For example, the insert portion of the schema of the incident table, when requested with \texttt{hierarchical=true} displays its hierarchy as follows:

\begin{verbatim}
https://instance.service-now.com/incident.do?WSDL\&hierarchical=true
\end{verbatim}
The WSDL above shows the incident table having a related table `u_custom_comments` that itself has a related table `u_comment_items`.

**Hierarchical SOAP Message**

When the SOAP message is constructed from the hierarchical web service described by the WSDL and invoked, it will create the `incident`, `u_custom_comments`, and `u_comment_items` records.

**Endpoint URL**

https://instance.service-now.com/incident.do?SOAP&hierarchical=true

**Request**

```
<soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:inc="http://www.service-now.com/incident">
  <soapenv:Header/>
  <soapenv:Body>
    <inc:insert>
      <short_description>test hierarchical</short_description>
      <u_custom_comments>
        <u_comment>comment 1</u_comment>
        <u_comment_type>travel</u_comment_type>
      </u_custom_comments>
    </inc:insert>
  </soapenv:Body>
</soapenv:Envelope>
```
Specify requirement for signed SOAP requests

Use a SOAP security policy to specify whether the instance requires signed SOAP requests for all inbound SOAP traffic.

Before you begin
Role required: web_service_admin or admin

About this task
By default, all inbound SOAP traffic must be signed. Administrators may want to disable this policy and allow unsigned SOAP requests to ServiceNow web services.

Procedure
2. Select the Default Policy.
3. Clear the Required to Sign SOAP Request check box (selected by default) to disable the requirement.
4. Click Update.
Activate the Enhanced Web Service Provider - Common plugin

Administrators can activate the Enhanced Web Service Provider - Common plugin to enable unsigned WS-Security requests and specify what authentication requirements SOAP requests have.

Before you begin
Role required: admin

Procedure

1. Navigate to System Applications > All Available Applications > All.
2. Find the Enhanced Web Service Provider - Common plugin (com.glide.web_service_provider_v2) using the filter criteria and search bar.
   
   You can search for the plugin by its name or ID. If you cannot find a plugin, you might have to request it from ServiceNow personnel. For more information, see Request a plugin.
3. Click Install, and then in the Activate Plugin dialog box, click Activate.

   Note: When domain separation and delegated admin are enabled in an instance, the administrative user must be in the global domain. Otherwise, the following error appears: Application installation is unavailable because another operation is running: Plugin Activation for <plugin name>.

Installed with the Enhanced Web Service Provider - Common plugin

The following components installed with the Enhanced Web Service Provider - Common plugin.

The Enhanced Web Service Provider - Common plugin installs the following components:

<table>
<thead>
<tr>
<th>Component Type</th>
<th>Component Installed</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module</td>
<td>Web Services Security Profiles</td>
<td>The plugin adds this module to the System Web Services application.</td>
</tr>
<tr>
<td>System Property</td>
<td>glide.soap.default_security_policy</td>
<td>Specifies the default security policy to use when enforcing</td>
</tr>
</tbody>
</table>
Components installed with the Enhanced Web Service Provider - Common plugin (continued)

<table>
<thead>
<tr>
<th>Component Type</th>
<th>Component Installed</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Web Services-Security (WSS) for inbound requests.</td>
</tr>
</tbody>
</table>

**Configure SOAP security**

Administrators can configure web service security for inbound SOAP requests made to the ServiceNow instance.

**Before you begin**

Role required: admin

**About this task**

You can also set up web service security to use different certificates for different web service clients. By enabling web service security, you can prevent man-in-the-middle attacks.

ℹ️ **Note:** After you configure a WS-security profile or a security policy, validation is performed on all incoming SOAP requests, including from the MID Server or ODBC driver. Disable validation for these types of requests by marking the service accounts as internal integration users.

**Procedure**

1. Upload a certificate to the instance.
2. Create a WS-security profile.
3. Create a security policy.
   Security policies define which WS-security profiles are used to evaluate a particular web service request. If no policy is defined, all WS-security profiles are used to evaluate all requests.
4. Set the value of the property `glide.soap.default_security_policy` to the name of the new security policy.

**Set the SOAP default security policy**

Set the SOAP default security policy.

**Before you begin**

Role required: web_service_admin or admin
Procedure

1. Navigate to **System Web Services > Properties**.

2. In the **Security Policy to enforce if WS-Security is enabled** field, enter the default security policy to use when enforcing WS-security.

3. Click **Save**.

Create a new security policy

Administrators can specify which security profiles WS-Security communications must meet by creating a new security policy.

Before you begin

Role required: web_service_admin or admin

Procedure

1. Navigate to **System Web Services > SOAP Security Policies**.

2. Click **New**.

3. Fill out the SOAP Security Policy form (see table).

### SOAP Security Policy form fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique name for the security policy. Use this name to set the default security policy with the glide.soap.default_security_policy property.</td>
</tr>
<tr>
<td>Type</td>
<td>Select whether the SOAP security policy applies to inbound or outbound traffic.</td>
</tr>
<tr>
<td>Required to Sign SOAP Request</td>
<td>Select this checkbox to require signed SOAP requests. Clear the checkbox to allow unsigned SOAP requests. When enabled, the instance will produce an error for any SOAP request that does not include a valid signature. When disabled, the instance still verifies any signature included with a SOAP request.</td>
</tr>
<tr>
<td>Authenticate</td>
<td>Select if a SOAP request must authenticate against all security profiles or at least one security profile.</td>
</tr>
</tbody>
</table>
4. Click **Submit**.

**Create a new WS-Security profile**

Create a new WS Security profile to define how to authenticate a web services message when WS-Security is enabled.

**Before you begin**

Role required: web_service_admin or admin

**Procedure**

1. Navigate to **System Web Services > WS Security Profiles**.
2. Click **New**.
3. Fill in the WS-Security Profile form (see table).  

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security Profiles</td>
<td>Select the security profiles you want to apply to this SOAP security policy. You must select at least one security profile.</td>
</tr>
</tbody>
</table>

**WS-Security Profile form fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique name for the security profile.</td>
</tr>
<tr>
<td>Type</td>
<td>Select <strong>X509</strong> to verify the request's certificate. Select <strong>Username</strong> to verify the request's user credentials.</td>
</tr>
<tr>
<td>Run as user</td>
<td>Select the ServiceNow user the instance will impersonate if authentication succeeds and the <strong>Bind Session</strong> field is selected. All web services requests will be attributed to this user. For example, if you select the <strong>System Administrator</strong> user then the instance treats all web services operations as being done by the system administrator. Make sure the user you select has appropriate SOAP privileges if you are using the <code>glide.soap.strict_security</code> high security setting. This field is only visible when the type is <strong>X509</strong>.</td>
</tr>
<tr>
<td>Order</td>
<td>Enter the order in which the instance checks security profiles. The instance checks all security profiles when processing an incoming SOAP request. If a request fails any security profile authentication requirement, the instance stops processing additional security profiles and fails the request.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Bind Session</td>
<td>Select this check box to have the instance impersonate the user listed in the <strong>Run as user</strong> field. You should only set this field for one profile at a time. If multiple profiles have this field selected, ServiceNow impersonates the user from the last successfully authenticated WS-Security profile. If no profile has this field selected, ServiceNow impersonates the user provided with the basic authentication headers or impersonates the default user (Guest).</td>
</tr>
<tr>
<td>X509 Certificate</td>
<td>Select the certificate record containing the certificate for web service requests. ServiceNow only validates the request signature. It automatically trusts the certificate’s certificate authority (CA). This field is only visible when the type is <strong>X509</strong>.</td>
</tr>
<tr>
<td>Profile action</td>
<td>Select how you want the instance to authenticate the user credentials. Select <strong>Authenticate with user</strong> if you want the instance to authenticate the request based on an existing user record. The request’s credentials must match values in an existing user record. Select <strong>Specify user to authenticate</strong> if you want to list a user name and password combination that all web services requests must meet. The request’s credentials must match the user name and password you list. This field is only visible when the type is <strong>Username</strong>.</td>
</tr>
<tr>
<td>User field to match</td>
<td>Select the column from the User [sys_user] table containing the value you want match against the request’s UserName. For example, if you select <strong>Email</strong> then the request UserName header must contain an email address matching an existing ServiceNow user. This field is only visible when the profile action is <strong>Authenticate with user</strong>.</td>
</tr>
<tr>
<td>User name</td>
<td>Enter the user name that all web services requests must contain. This field is only visible when the profile action is <strong>Username</strong>.</td>
</tr>
<tr>
<td>User password</td>
<td>Enter the password that all web services requests must contain. This field is only visible when the profile action is <strong>Username</strong>.</td>
</tr>
</tbody>
</table>

4. Click **Submit**.
Related information
- SOAP web services security
- WS-Security
- WS-Security profiles

Enforce strict security for inbound SOAP
Strict security for web services requires that users meet Contextual Security requirements to access instance resources.

Before you begin
Role required: admin

About this task
Note: ServiceNow does not support digital certificates, digital signatures, or other digested token methods in SOAP web service calls.

To enforce strict security for web services connections:

Procedure
1. Navigate to System Properties > Web Services.
2. Select Yes for Enforce strict security on incoming SOAP requests.

Note: To learn more about this property, see SOAP request strict security (instance security hardening) in Instance Security Hardening Settings.

Enable WS-Security verification
Administrators can enable Web Services Security (WSS) verification from the Web Services system properties.

Before you begin
Role required: web_service_admin or admin
Procedure

1. Navigate to System Web Services > Properties.
2. For Require WS-Security header verification for all incoming SOAP requests, select Yes.

   ![Require WS-Security header verification for all incoming SOAP requests]

   Yes | No

   **Note:** Selecting this option enables WS-Security for all inbound SOAP requests. It is not possible to enable WS-Security for only some requests.

3. Click Save.

4. Create a WS-security profile.

5. Update the user record for the MID Server and ODBC driver to mark these users as internal integration users.

6. Download and install the latest MID Server and ODBC driver.

7. To validate SOAP request signatures, upload the remote web service's certificate as a JKS and create the web service's WSS Username Token Profile.

   **Note:** Because ServiceNow's WSS implementation does not verify the CA certificate, you do not need to upload the web service's CA certificate.

Related information

Basic authentication

Mark service accounts as internal integration users

Allow internal integration communications to bypass the WSS authentication requirement by marking their user accounts as internal integration users.

Before you begin

Role required: admin

About this task

When WS-Security is enabled, authentication is required for all SOAP requests including internal integration communications such as the MID Server, ODBC Driver, Remote Update Sets, and high availability cloning. SOAP requests for these internal integration communications cannot implement WS-Security due to technical implications. If your instance uses these SOAP interfaces, you can allow them to bypass the WS-Security authentication requirement by marking their user accounts as internal integration users.
Note: Any web services other than ODBC, MID Server, Remote Update Sets, or high availability cloning must implement WS-Security headers when WS-Security is enabled.

Procedure
1. Navigate to User Administration > Users.
2. Select the user account for the MID Server or ODBC Driver.
3. Configure the form to add the Internal Integration User field.
4. Select the Internal Integration User check box.
5. Click Update.

Related information
Enable WS-Security verification
WS-Security

Debug incoming SOAP envelope
To capture incoming SOAP envelope XML in the system log, add the property glide.processor.debug.SOAPProcessor with a value of true.

When enabled, this property adds the incoming SOAP envelope in the Message field of the system log (System Logs > All). Disable this debugging feature as soon as you are finished so that the log is not overwhelmed with excessive and unnecessary debugging information.

View a SOAP session log
You can view a user's log from a SOAP session.

Before you begin
Role required: admin

Procedure
1. Navigate to User Administration > Logged in users.
2. Open an active SOAP session to see the transactions log. The SOAP session is marked as inactive within 60 seconds of the last transaction.

**Basic authentication code samples**
Samples of basic authentication code for several programming languages and versions.

**Perl and the SOAP::Lite libraries**
To supply basic authentication when using Perl and the SOAP::Lite libraries, you can implement the following function:

```perl
sub SOAP::Transport::HTTP::Client::get_basic_credentials {
    return 'user_name' => 'password';
}
```

**C# .NET VS 2005 or older**
When using C# .NET VS 2005 or older, you can take advantage of the Credentials object. For example:

```csharp

service.ServiceNow proxy = new service.ServiceNow();
service.get getService = newservice.get();
service.getResponse getServiceResponse = new service.getResponse();

try {
    proxy.Credentials = cred;
    getService.sys_id = "bf522c350a0a140701972dbf876f1610";
    getServiceResponse = proxy.get(getService);
} catch (Exception ex) { }
```

**C# .NET VS 2008**
When using C# .NET VS 2008, you can take advantage of the ClientCredentials object. For example:

```csharp
Demo_Incident.ServiceNowSoapClient client = new
test08WebService.Demo_Incident.ServiceNowSoapClient();
client.ClientCredentials.UserName.UserName = "admin";
client.ClientCredentials.UserName.Password = "admin";
```
Then in your `app.config` file look for the following and change "None" to "Basic":

```
<transport clientCredentialType="None" proxyCredentialType="None" realm="" />
```

**VB .NET**

When using VB .NET taking advantage of the Credentials object would look like the following:

```vbnet
Sub Main()
    Dim cred As New System.Net.NetworkCredential("user_name", "password")
    Dim proxy As New VB_Democm.incident.ServiceNow
    Dim getIncident As New VB_Democm.incident.get
    Dim getResponse As New VB_Democm.incident.getResponse

    proxy.Credentials = cred

    getIncident.sys_id = "[your sysID here]"

    getResponse = proxy.get(getIncident)

End Sub
```

The resulting response when Basic Authentication is turned on and no credentials are supplied looks like this:

```
<html>
<head>
<title>Apache Tomcat/5.0.28 - Error report</title>
<style>  <!--H1
(font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:22px;)
H2
(font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:16px;)
H3
(font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:14px;)
BODY (font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;)
B (font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;)
P (font-family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;)
A (color: black;)
A.name (color: black;)
HR (color: #525D76;)-->
</style>
```
<SOAP-ENV:Header>
  <wsse:Security
    xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
    SOAP-ENV:mustUnderstand="1">
    <wsse:BinarySecurityToken
      xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
      EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary"
      ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3"
      wsu:Id="CertId-2D914AB929A6719E7F13068829874641" />
    <wsu:Id="CertId-2D914AB929A6719E7F13068829874641" />
  </wsse:Security>
</SOAP-ENV:Header>
### WS-Security properties

These properties control the behavior of WS-Security X.509 tokens.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>glide.soap.msg_digest.algorithm</td>
<td>Specifies the method digest algorithm. Possible values are SHA-1, SHA-256, and SHA-512.</td>
</tr>
<tr>
<td></td>
<td>- Type: String</td>
</tr>
<tr>
<td></td>
<td>- Default value: SHA-1</td>
</tr>
<tr>
<td></td>
<td>- Location: Add to the System Property [sys_properties] table</td>
</tr>
<tr>
<td>glide.soap.signature.algorithm</td>
<td>Specifies the signature algorithm. Possible values are RSA-SHA-1, RSA-SHA-256, and RSA-SHA-512.</td>
</tr>
<tr>
<td></td>
<td>- Type: String</td>
</tr>
<tr>
<td></td>
<td>- Default value: RSA-SHA-1</td>
</tr>
<tr>
<td></td>
<td>- Location: Add to the System Property [sys_properties] table</td>
</tr>
<tr>
<td>glide.soap.canonical.algorithm</td>
<td>Specifies the cannonicalization algorithm. Possible values are Canonical xml 1.0, Canonical xml 1.0 with Comments, Exclusive Canonical xml 1.0, and Exclusive Canonical xml 1.0 with Comments.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Type: String</td>
<td></td>
</tr>
<tr>
<td>• Default value:</td>
<td>Exclusive Canonical xml 1.0</td>
</tr>
<tr>
<td>• Location:</td>
<td>Add to the System Property [sys_properties] table</td>
</tr>
</tbody>
</table>

Each possible value for these properties represents a standard WS-Security algorithm.

Property value URLs

<table>
<thead>
<tr>
<th>Value</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method digest algorithms</td>
<td></td>
</tr>
<tr>
<td>SHA-1</td>
<td><a href="http://www.w3.org/2000/09/xmldsig#sha1">http://www.w3.org/2000/09/xmldsig#sha1</a></td>
</tr>
<tr>
<td>SHA-256</td>
<td><a href="http://www.w3.org/2001/04/xmlenc#sha256">http://www.w3.org/2001/04/xmlenc#sha256</a></td>
</tr>
<tr>
<td>SHA-512</td>
<td><a href="http://www.w3.org/2001/04/xmlenc#sha512">http://www.w3.org/2001/04/xmlenc#sha512</a></td>
</tr>
<tr>
<td>Signature algorithms</td>
<td></td>
</tr>
<tr>
<td>RSA-SHA-1</td>
<td><a href="http://www.w3.org/2000/09/xmldsig#rsa-sha1">http://www.w3.org/2000/09/xmldsig#rsa-sha1</a></td>
</tr>
<tr>
<td>RSA-SHA-256</td>
<td><a href="http://www.w3.org/2001/04/xmlenc-more#rsa-sha256">http://www.w3.org/2001/04/xmlenc-more#rsa-sha256</a></td>
</tr>
<tr>
<td>RSA-SHA-512</td>
<td><a href="http://www.w3.org/2001/04/xmlenc-more#rsa-sha512">http://www.w3.org/2001/04/xmlenc-more#rsa-sha512</a></td>
</tr>
<tr>
<td>Canonicalization algorithms</td>
<td></td>
</tr>
<tr>
<td>Canonical xml 1.0</td>
<td><a href="http://www.w3.org/TR/2001/REC-xml-c14n-20010315">http://www.w3.org/TR/2001/REC-xml-c14n-20010315</a></td>
</tr>
<tr>
<td>Canonical xml 1.0 with Comments</td>
<td><a href="http://www.w3.org/TR/2001/REC-xml-c14n-20010315#WithComments">http://www.w3.org/TR/2001/REC-xml-c14n-20010315#WithComments</a></td>
</tr>
<tr>
<td>Exclusive Canonical xml 1.0</td>
<td><a href="http://www.w3.org/2001/10/xml-exc-c14n#">http://www.w3.org/2001/10/xml-exc-c14n#</a></td>
</tr>
<tr>
<td>Exclusive Canonical xml 1.0 with comments</td>
<td><a href="http://www.w3.org/2001/10/xml-exc-c14n#WithComments">http://www.w3.org/2001/10/xml-exc-c14n#WithComments</a></td>
</tr>
</tbody>
</table>
WS-Security error messages

An instance produces one of the following error messages when it encounters an issue with a WS-security SOAP message.

### WS-security error messages

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invalid Security Policy Selected. Select an Inbound policy for Inbound Requests.</td>
<td>The default policy name is set to an outbound policy. Set the default policy name to an inbound security policy.</td>
</tr>
<tr>
<td>SOAP request not Signed. Policy requires signing.</td>
<td>The SOAP security policy requires signing and the inbound SOAP request is not signed. Either specify a different SOAP security policy or provide the SOAP request with a proper signature.</td>
</tr>
<tr>
<td>No profiles to authenticate.</td>
<td>The selected Security policy either does not have any security profiles (X509 or UserNameToken) or the security profiles are inactive. Verify at least one security profile is active.</td>
</tr>
<tr>
<td>Unable to verify signed request.</td>
<td>The inbound SOAP request contains an invalid signature. The SOAP request changed after being signed.</td>
</tr>
<tr>
<td>Failed to extract principal(s) from request.</td>
<td>The SOAP request has either been tampered or was not well formed. ServiceNow cannot extract credentials to authenticate the request.</td>
</tr>
<tr>
<td>Failed to authenticate WS-security, unknown type.</td>
<td>The SOAP request contains an unsupported security profile. Resend the request with a supported security profile type: X509 or UsernameToken.</td>
</tr>
<tr>
<td>Failed to authenticate WS-security.</td>
<td>ServiceNow failed to authenticate against the provided profile credentials. Verify that the SOAP request is using the proper credentials.</td>
</tr>
</tbody>
</table>

**LongRunningSOAPRequestProps**

The following properties are available for long-running SOAP requests.
### Long-running SOAP request properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>glide.http.connection_timeout</code></td>
<td>Specify the maximum number of milliseconds an outbound HTTP request (such as Web Services) will wait to establish a connection.</td>
</tr>
<tr>
<td></td>
<td>• Type: integer</td>
</tr>
<tr>
<td></td>
<td>• Default value: 10000 (10 seconds)</td>
</tr>
<tr>
<td></td>
<td>• Location: system properties [sys_properties] table</td>
</tr>
<tr>
<td><code>glide.http.timeout</code></td>
<td>Specifies the maximum number of milliseconds to wait before an outbound transaction times out.</td>
</tr>
<tr>
<td></td>
<td>• Type: integer</td>
</tr>
<tr>
<td></td>
<td>• Default value: 175000 (175 seconds)</td>
</tr>
<tr>
<td></td>
<td>• Location: Add to system properties [sys_properties] table</td>
</tr>
<tr>
<td><code>glide.soap.max_redirects</code></td>
<td>Specifies the maximum number of redirects the server sends to the client before the soap request is timed out.</td>
</tr>
<tr>
<td></td>
<td>• Type: integer</td>
</tr>
<tr>
<td></td>
<td>• Default value: 20</td>
</tr>
<tr>
<td></td>
<td>• Location: system properties [sys_properties] table</td>
</tr>
<tr>
<td><code>glide.soap.request_processing_timeout</code></td>
<td>Specify the maximum number of seconds an inbound SOAP request has to finish processing before the connection times out. This property computes a default value from the value of the</td>
</tr>
<tr>
<td></td>
<td>• Type: integer</td>
</tr>
<tr>
<td></td>
<td>• Default value: 60000 (60 seconds)</td>
</tr>
<tr>
<td></td>
<td>• Location: system properties [sys_properties] table</td>
</tr>
</tbody>
</table>
Long-running SOAP request properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>glide.http.timeout divided by 1000.</td>
<td>This property accepts values 5–1200 seconds (20 minutes). Customers might have network infrastructure (such as proxy servers) in place which implement a shorter timeout. In this case, a socket timeout may occur unless this property is set to a shorter value. Set this property to a value several seconds less than the shortest socket inactivity timeout in effect anywhere in the network path between the client application and the instance.</td>
</tr>
<tr>
<td></td>
<td>• Type: integer</td>
</tr>
<tr>
<td></td>
<td>• Location: system properties [sys_properties] table</td>
</tr>
<tr>
<td>glide.soapprocessor.allow_long_running_threads</td>
<td>Enables or disables a 307-Temporary Redirect when the request includes a redirectSupported=true parameter. The default setting is true (enabled).</td>
</tr>
<tr>
<td>glide.soapprocessor.max_long_running_threads</td>
<td>Controls the maximum number of long-running SOAP threads which can run at any one time. The default value for this property is determined dynamically based on the number of SOAP semaphores configured. It should not be</td>
</tr>
</tbody>
</table>
Long-running SOAP request properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>necessary to change this value.</td>
</tr>
</tbody>
</table>

Inbound web service examples

Inbound web service examples demonstrate how to access ServiceNow web services.

Java Apache Axis2 web services client examples

Examples demonstrating an integration with Axis2 Version 1.4.

Requirements

- An "elementFormDefault" value of qualified means that an unqualified element is in the default namespace defined on an ancestor. If it is "unqualified" then an unqualified element is in the empty namespace (xmlns=""). The default is "unqualified".
- To Resolve the Axis Client deserialization failure you should go to System Properties > Web Services and uncheck the property that sets the elementFormDefault attribute of the embedded XML schema to the value of unqualified. Save the property setting and regenerate your Axis2 client code if your client code was generated before changing this property.

Java Apache Axis2 web services client examples insert

An example class to insert an incident record.

```java
public class Insert {
    public static void main ( String args [ ] ) { try {
        HttpTransportProperties. Authenticator basicAuthentication  = new HttpTransportProperties. Authenticator ( ) ;
        basicAuthentication. setUsername ( "admin" ) ;
        basicAuthentication. setPassword ( "admin" ) ;
```
ServiceNowStub proxy = new ServiceNowStub () ; 

proxy._getServiceClient ( ) . getOptions ( ) . setProperty (org. apache. axis2. transport. 
http. HTTPConstants. CHUNKED, Boolean. FALSE ) ; 
proxy._getServiceClient ( ) . getOptions ( ) . setProperty (org. apache. axis2. transport. 
http. HTTPConstants. AUTHENTICATE, basicAuthentication ) ; 

ServiceNowStub. Insert inc = new ServiceNowStub. Insert ( ) ; 
ServiceNowStub. InsertResponse resp = new ServiceNowStub. InsertResponse ( ) ; 

inc. setAssigned_to ( "Christen Mitchell" ) ; 
inc. setCategory ( "hardware" ) ; 
inc. setPriority ( BigInteger. ONE ) ; 
inc. setDescription ( "The WI_FI in the reception area is down" ) ; 
inc. setCaller_id ( "Joe Employee" ) ; 

resp = proxy. insert (inc ) ; 

System. out. println ( "New Incident: " + resp. getNumber ( ) ) ; } catch ( Exception 
e ) { System. out. println ( e. toString ( ) ) ; }

} 

Java Apache Axis2 web services client examples update
An example of an Axis Client program that calls the getKeys function to query all incidents where the category is Hardware.

getKeys
A list of sys_id is returned as a result:

package com.service_now.www ;

public class DemoClient { 

public static void main ( String args [ ] ) { try { 
ServiceNowStub proxy = new ServiceNowStub () ; 
ServiceNowStub. GetKeys getInc = new ServiceNowStub. GetKeys ( ) ; 
ServiceNowStub. GetKeysResponse resp = new ServiceNowStub. GetKeysResponse ( ) ;

getInc. setActive ( true ) ; 
getInc. setCategory ( "hardware" ) ;
http. HTTPConstants. CHUNKED,  Boolean. FALSE ) ;

resp  = proxy. getKeys (getInc ) ;

String [ ] keys  = resp. getSys_id ( ) ;

System. out. println ( "Key: " + keys [ 0 ] ) ; } catch ( Exception e ) { System. out.
println (e. toString ( ) ) ; }

} }

getRecords

package com.service_now.www ;

import com.service_now.www.ServiceNowStub.GetRecordsResult_type0 ;

public class GetRecords {  

/**
 * @param args
 */
* / public static void main ( String [ ] args ) { try {
ServiceNowStub proxy  = new ServiceNowStub ( ) ;
ServiceNowStub. GetRecords incidents  = new ServiceNowStub. GetRecords ( ) ;
ServiceNowStub. GetRecordsResponse result  = new ServiceNowStub. GetRecordsResponse ( ) ;

incidents. setActive ( true ) ;
incidents. setCategory ( "hardware" ) ;
incidents. setSys_created_on ( "> 2009-06-08 10:30:00" ) ;

http. HTTPConstants. CHUNKED,  Boolean. FALSE ) ;

result  = proxy. getRecords (incidents ) ;

GetRecordsResult_type0 [ ] keys  = result. getGetRecordsResult ( ) ;

for ( int key  = 0 ; key  < keys. length ; key ++ ) { System. out. println ( "Key: 
" + keys [ 0 ]. getSys_id ( ) ) ; } catch ( Exception e ) { System. out. println (e.
toString ( ) ) ; }
Java Apache Axis2 web services client examples advanced

Examples showing how to construct and use an Axis2 client to consume a ServiceNow Web Service.

Axis is essentially a SOAP engine -- a framework for constructing SOAP processors such as clients, servers, or gateways. The current version of Axis is written in Java. This content is intended for system admins with a light development background in Java. To begin you would need Java JDK version 1.4.2 or higher and Axis2 version 1.0 or higher.

Create a Java Project
This example uses Eclipse SDK Version: 3.4.2 for managing the source code and executing the web request. Eclipse is not required.

• Open Eclipse and from the menu select File > New > Project > Java Project.
• Give the project a name.
• Verify that the correct JRE is specified.
◦ If using wsdl2java run "java -version" on the command line and this will be the version to specify for the project specific JRE.
◦ If using the Axis2 Codegen plugin use default JRE.
Generate your Axis2 client code

- From a command line in the bin directory of the axis folder:

```
./wsdl2java.sh -uri https://<instance name>.service-now.com/incident.do?WSDL
-o /glide/workspace/TestWebService/
```

- In the above example:
  - The "-uri" is either the path where you have saved a copy of the wsdl to either ".wsdl" or ".xml", or the URL the WSDL resides at.
  - The "-o" is the path where you want the files to be written out to. If not specified, the files will be written out to the current bin location.

- In Eclipse refresh the project and the generated Stub and CallbackHandler should now be displayed

<table>
<thead>
<tr>
<th>Axis Stub</th>
<th>src</th>
<th>com.service_now.www</th>
<th>ServiceNowCallbackHandler.java</th>
<th>ServiceNowStub.java</th>
</tr>
</thead>
</table>

Basic Authentication

```
HttpTransportProperties.Authenticator basicAuthentication = new HttpTransportProperties.Authenticator();
basicAuthentication.setUsername("admin");
basicAuthentication.setPassword("admin");
...
ServiceNowStub proxy = new ServiceNowStub();
...
proxy._getServiceClient().getOptions().setProperty(org.apache.axis2.transport.http.HTTPConstants.AUTHENTICATE, basicAuthentication);
```

Compatibility with Axis2 Versions 1.1 and higher

Chunking support is only available in HTTP Version 1.1. By default chunking is enabled in Axis2.xml for versions 1.1 and higher. ServiceNow does not support Chunking, so you will need to disable chunking at deployment time or at runtime.
• Deployment time: One can disable HTTP chunking by removing or commenting out the following element from Axis2.xml

```xml
<parameter name= "Transfer-Encoding" >chunked</parameter>
```

• Runtime: User can disable the chunking using following property set in Client or Stub, versions 1.1.1 and higher only

```java
options.setProperty (org.apache.axis2.transport.http.HTTPConstants.CHUNKED, Boolean.FALSE );
```

Creating Unique Packages
You can use the Axis2 parameter namespace2package (ns2p) to create unique package names. The parameter uses this format:

```bash
<Axis path>\bin\wsdl2java.bat -u -p cr2 -ns2p <namespace>=<package name> -uri <wsdl to convert>
```

For example:

```bash
<Axis path>\bin\wsdl2java.bat -u -p cr2 -ns2p
http://www.service-now.com/change_request=my.change_request -uri change_request
```

Microsoft .NET web services client examples
Examples demonstrating an integration with Microsoft .NET Web Services Client.

Requirements
.NET 2.0 Versions and Higher:

• An "elementFormDefault" value of qualified means that an unqualified element is in the default namespace defined on an ancestor. If it is "unqualified" then an unqualified element is in the empty namespace (xmlns=""). The default is "unqualified".

• To Resolve the .NET Client deserialization failure you should go to System Properties > Web Services and uncheck the property that sets the elementFormDefault attribute of the embedded XML schema to the value of unqualified. Save the property setting and recreate your WSDL Reference.cs class. See Also "Compatibility with Clients generated from WSDL" below.
Example insert using Visual Basic .NET

A sample Visual Basic .NET program that inserts a core_company record.

```vbnet
Public Class Class1
    Shared Sub Main()
        Dim proxyCompany As New core_company.localhost.ServiceNow()

        Dim companyInsert As New core_company.localhost.insert()

        Dim companyInsertResponse As core_company.localhost.insertResponse

        With companyInsert
            .name = "Test Company SKF 2"
            .contact = "SKF"
            .customer = True
            .customerSpecified = True
        End With

        companyInsertResponse = proxyCompany.insert(companyInsert)

        Console.WriteLine(companyInsertResponse.sys_id)
    End Sub
End Class
```

Sample Visual Basic .NET project
Perl web services client examples

Examples demonstrating an integration with a Perl web services client.

Note: The following examples require the usage of the Perl language and the SOAP::Lite package.

System Requirements

- Perl 5.8
  - SOAP::Lite (prerequisites http://soap-lite.com/prereqs.html)
  - Crypt::SSLeay
  - IO::Socket::SSL

insert

The following example will insert a record into the Incident table.
insert (With XML payload)
The following is an example of inserting a record into the ecc_queue table where the payload field is an XML document. This is done using the Perl language and the SOAP::Lite package, the XML document creation uses the XML::Writer package:

```
#!/usr/bin/perl -w
use SOAP::Lite { +trace => all, maptype => {} };use SOAP::Lite;
use XML::Writer;use XML::Writer::String;
```
## Get parameters passed by OVO

```perl
notification#$OVMSG{id}=$ARGV[0];$OVMSG{node_name}=$ARGV[1];$OVMSG{node_type}=$ARGV[2];$OVMSG{date_created}=$ARGV[3];$OVMSG{time_created}=$ARGV[4];$OVMSG{date_received}=$ARGV[5];$OVMSG{time_received}=$ARGV[6];$OVMSG{application}=$ARGV[7];$OVMSG{msg_group}=$ARGV[8];$OVMSG{object}=$ARGV[9];$OVMSG{severity}=$ARGV[10];$OVMSG{operator_list}=$ARGV[11];$OVMSG{msg_text}=$ARGV[12];$OVMSG{instruction}=$ARGV[13];
```

```perl
sub SOAP::Transport::HTTP::Client::get_basic_credentials{return 'itil' =>'itil';}
```

```perl
my $soap= SOAP::Lite->proxy('http://<instance name>.service-now.com/ecc_queue.do?SOAP');
```

```perl
my $method= SOAP::Data->name('insert')->attr({xmlns =>$http://www.service-now.com/});
```

```perl
# get all incidents with category Network
my @params=( SOAP::Data->name(agent =>'OVO_Notification'));push(@params, SOAP::Data->name(queue =>'input'));push(@params, SOAP::Data->name(name =>'HP Openview OVO Notification'));push(@params, SOAP::Data->name(source =>$OVMSG{id}));
```

```perl
my $s= XML::Writer::String->new();my $writer=new XML::Writer(OUTPUT =>$s);
```

```perl
#$writer->xmlDecl();$writer->startTag('notification');
```

```perl
write_element('id');
write_element('node_name');
write_element('node_type');
write_element('date_created');
write_element('time_created');
write_element('date_received');
write_element('time_received');
write_element('application');
write_element('msg_group');
write_element('object');
write_element('severity');
write_element('operator_list');
write_element('msg_text');
write_element('instruction');
```

```perl
$writer->endTag('notification');
```

```perl
$writer->end;
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc. in the United States and/or other countries.
Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
sub write_element
   (my$label=shift;my$value=GVMSG{$label};$writer->startTag($label);if($value){$writer->characters($value);$writer->endTag($label);})

push(@params, SOAP::Data->name(payload =>$s->value()));

print$soap->call($method=>@params)->result;</pre>

--- Response to the 'insert'---

<xml version="1.0" encoding="UTF-8">
  <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soap:Body>
      <insertResponse xmlns="http://www.service-now.com/ecc_queue">
        <sys_id>1a5ad50e0a0a021101bef2e07705f87a</sys_id>
        <name>HP Openview OVO Notification</name>
      </insertResponse>
    </soap:Body>
  </soap:Envelope>
</xml>

update

#!/usr/bin/perl -w

#use SOAP::Lite (+trace => all, maptype => {}); use SOAP::Lite;

sub SOAP::Transport::HTTP::Client::get_basic_credentials { return 'itil' =>'itil'; }

my$soap= SOAP::Lite->proxy('http://localhost:8080/glide/incident.do?SOAP');

my$method= SOAP::Data->name('update')->attr({xmlns =>'http://www.service-now.com/'});

# update incident by sys_idmy@params=( SOAP::Data->name(sys_id =>'e8caedcb0a80164017df472f39eae1'));
push(@params, SOAP::Data->name(short_description =>'this is a new description'));

my$result=$soap->call($method=>@params);

print_fault($result);
print_result($result);

sub print_result {my($result)=@_;}
sub print_fault {my($result)=@_;}

if($result->body&&$result->body->{'updateResponse'}){my%keyHash=%{$result->body->{'updateResponse'}};
foreach my$k (keys%keyHash){print"name=$k value=$keyHash($k)\n";}}

sub print_result {my($result)=@_;}
sub print_fault {my($result)=@_;}
if($result->fault){print"faultcode=".$result->fault->{'faultcode'}."\n";print"faultstring=".$result->fault->{'faultstring'}."\n";print"detail=".$result->fault->{'detail'}."\n";}

getKeys
The following is an example of retrieving a list of sys_ids for records of Incident where Category is Network.

#!/usr/bin/perl -w
#use SOAP::Lite ( +trace => all, maptype => {} );use SOAP::Lite;
sub SOAP::Transport::HTTP::Client::get_basic_credentials{return'itil'=>'itil';}
my$soap= SOAP::Lite->proxy('http://<instance name>.service-now.com/incident.do?SOAP');
my$method= SOAP::Data->name('getKeys')->attr({xmlns =>'http://www.service-now.com/'});
# get all incidents with category Network
my@params=( SOAP::Data->name(category =>'Network'));
print$soap->call($method=>@params)->result;

get
The following is an example of retrieving an Incident record using its sys_id value

#!/usr/bin/perl -w
#use SOAP::Lite ( +trace => all, maptype => {} );use SOAP::Lite;
sub SOAP::Transport::HTTP::Client::get_basic_credentials{return'itil'=>'itil';}
my$soap= SOAP::Lite->proxy('http://<instance name>.service-now.com/incident.do?SOAP');
my$method= SOAP::Data->name('get')->attr({xmlns =>'http://www.service-now.com/'});
# get incident by sys_id
my@params=( SOAP::Data->name(sys_id =>'9d385017c611228701d2104cc95c371'));
my$keyHash=\$soap->call($method=>@params)->body->{'getResponse'};
# iterate through all fields and print
foreachmy$k(keys$keyHash){print"$k=$keyHash{$k}\n";}

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
**getRecords**

To query for an Incident using its incident number value:

```perl
#!/usr/bin/perl -w
use SOAP::Lite ( +trace => all, maptype => {} );
use SOAP::Lite;

sub SOAP::Transport::HTTP::Client::get_basic_credentials{return 'itil' => 'itil';}

my $soap = SOAP::Lite->proxy('http://<instance name>.service-now.com/incident.do?SOAP');

my $method = SOAP::Data->name('getRecords')->{'xmlns' => 'http://www.service-now.com/'};

get incident by number

my @params = (SOAP::Data->name(number => 'INC10001'));

my %keyHash = %{$soap->call($method => @params)->body->{'getRecordsResponse'}->{'getRecordsResult'});

# iterate through all fields and print them
foreach my $k (keys %keyHash)
{print "$k=$keyHash{$k}\n";
}
```

**getRecords (Returning Multiple Results)**

The following is an example of retrieving and displaying an array of Incident records by querying all Incidents that have a "Category" of "Network"

```perl
#!/usr/bin/perl -w
use SOAP::Lite ( +trace => all, maptype => {} );
use SOAP::Lite;

sub SOAP::Transport::HTTP::Client::get_basic_credentials{return 'itil' => 'itil';}

my $soap = SOAP::Lite->proxy('http://<instance name>.service-now.com/incident.do?SOAP');

my $method = SOAP::Data->name('getRecords')->{'xmlns' => 'http://www.service-now.com/'};

# get incident by sys_id
my @params = (SOAP::Data->name(category => 'Network'));

my %keyHash = %{$soap->call($method => @params)->body->{'getRecordsResponse'});

my $i = 0;
my $size = @{$keyHash{'getRecordsResult'}};
for ($i = 0; $i < $size; $i++)
{
    my $record = @{$keyHash{'getRecordsResult'}}[$i];
    print "------------------------------ $i
------------------------------\n";
    foreach my $kk (keys %$record)
    {print "$kk=$record{$kk}\n";
    }
}
```

**deleteRecord**

```perl
#!/usr/bin/perl -w

use SOAP::Lite ( +trace => all, maptype => {} );
use SOAP::Lite;
```
sub SOAP::Transport::HTTP::Client::get_basic_credentials { return 'itil' => 'itil'; }

my $soap = SOAP::Lite->proxy('http://localhost:8080/glide/incident.do?SOAP');

my $method = SOAP::Data->name('deleteRecord')->attr({xmlns => 'http://www.service-now.com/'});

# delete incident by sys_idmy @params = ( SOAP::Data->name(sys_id =>'46f67787a9fe198101e06dfcf3a78e99'));

my $result = $soap->call($method => @params);

print_fault($result);
print_result($result);

sub print_result { my ($result) = @_; }

if ($result->body && $result->body->{'deleteRecordResponse'}) { my %keyHash = %{$result->body->{'deleteRecordResponse'}};
foreach my $k (keys %keyHash) { print "name=$k   value=$keyHash{$k}\n"; } }

sub print_fault { my ($result) = @_; }

if ($result->fault) { print "faultcode=".$result->fault->{'faultcode'}."\n"; print "faultstring=".$result->fault->{'faultstring'}."\n"; print "detail=".$result->fault->{'detail'}."\n"; }

---

**Python web services client examples**

Examples demonstrating an integration with a Python web services client.

**Requirements**
The following examples require the installation of the following Python modules:

- **fpconst** [http://pypi.python.org/pypi/fpconst/0.7.2](http://pypi.python.org/pypi/fpconst/0.7.2)
- **PyXML** [http://pyxml.sourceforge.net/topics/](http://pyxml.sourceforge.net/topics/)

**insert**

This is an example of inserting an incident.

```python
#!/usr/bin/python

from SOAPpy import SOAPProxy
import sys
```
def createincident (params_dict):
    
    # instance to send to
    instance = 'demo'
    
    # username/password
    username = 'itil'
    password = 'itil'
    
    # proxy - NOTE: ALWAYS use https://INSTANCE.service-now.com, not
    # https://www.service-now.com/INSTANCE for web services URL from now on!
    proxy = 'https://%s:%s@%s.service-now.com/incident.do?SOAP' % (username, password, instance)
    namespace = 'http://www.service-now.com/
    server = SOAPProxy (proxy, namespace)
    
    # uncomment these for LOTS of debugging output
    #server.config.dumpHeadersIn = 1
    #server.config.dumpHeadersOut = 1 #server.config.dumpSOAPOut = 1 #server.config.dumpSOAPIn = 1
    
    response = server.insert (impact = int (params_dict[ 'impact' ]), urgency = int (params_dict ['urgency'])), priority = int (params_dict ['priority'])),
    category =params_dict ['category'], location =params_dict ['location'], caller_id =params_dict ['user'], assignment_group =params_dict ['assignment_group'],
    assigned_to =params_dict ['assigned_to'], short_description =params_dict ['short_description'], comments =params_dict ['comments'])
    
    return response

values = { 'impact': 1, 'urgency': 1, 'priority': 1, 'category': 'High',
    'location': 'San Diego', 'user': 'fred.luddy@yourcompany.com', 'assignment_group':
    'Technical Support', 'assigned_to': 'David Loo', 'short_description': 'An incident created using python, SOAPpy, and web services.', 'comments': 'This a test making an incident with python. Isn\'t life wonderful?'}

new_incident_sysid =createincident (values)

print "Returned sysid: "+ repr (new_incident_sysid)
**getKeys**
This is an example of executing getKeys on the demo instance using basic authentication.

```python
#!/bin/env python

# use the SOAPpy module from SOAPpy import SOAPProxy

username, password, instance = 'admin', 'admin', 'demo'
proxy, namespace = 'https://username:password@www.service-now.com/'+instance+
'/incident.do?SOAP', 'http://www.service-now.com/

server = SOAPProxy(proxy, namespace)
response = server.getKeys(category = 'Network')

print response.sys_id.split(',')
```

**getRecords**
In this example, we get an incident, querying for category == "Network" (with basic authentication).

```python
#!/bin/env python

# use the SOAPpy module from SOAPpy import SOAPProxy

username, password, instance = 'admin', 'admin', 'demo'
proxy, namespace = 'https://username:password@www.service-now.com/'+instance+
'/incident.do?SOAP', 'http://www.service-now.com/

server = SOAPProxy(proxy, namespace)
response = server.getRecords(category = 'Network')

for record in response:
    for item in record:
        print item
```

**get**
In this example, we get an incident record by `sys_id` (with basic authentication).

```python
#!/bin/env python

# use the SOAPpy module from SOAPpy import SOAPProxy

username, password, instance = 'admin', 'admin', 'demo'
```

server = SOAPProxy(proxy,namespace)
response = server.get(sys_id = '9c573169c611228700193229fff72400')

for each in response:
    print each

---

Advanced
This is an example of advanced Python script that reads a log file for a keyword invalid spi and creates an ECC Queue record where the payload is set to an alert of XML format.

#!/bin/env python

# kevin.pickard@service-now.com 2008.07.03 initial creation

from SOAPpy import SOAPProxy
from xml.dom.minidom import getDOMImplementation
import sys, os, socket, pickle, re

# instance to send to
instance = 'demo'

# username/pass
username = 'admin'
password = 'admin'

# log file to watch
syslogfile = '/var/log/cisco.log.ksp'

# state file
statefile = '/tmp/syslog_ecc.state-test'

# ECC queue values
soapagent = 'SOAPpy'
cectopic = 'PIX Error: '
cecname = 'Invalid SPI: '
cecsource = 'Syslog'

# regex string to match
matchstring = 'invalid spi'
try:
state = open(statefile, 'r')
lastbyte = pickle.load(state)
state.close() except:
lastbyte = 0

#print 'DEBUG: lastbyte = '+'str(lastbyte)

try:
log = open(syslogfile, 'ro') except:
errortopic = 'Script Error'
errorname = 'Unable to open log file '+syslogfile+'.
errorpayload = 'This message was generated due to an error condition encountered in a script. The name of the script is '+os.path.basename(sys.argv[0])+ ' on server '+socket.gethostname()+ '.

proxy = 'https://'+username+':'+password+'@'+instance+'.service-now.com/ecc_queue.do?SOAP'
namespace = 'http://www.service-now.com/'
server = SOAPProxy(proxy, namespace)
server.config.dumpSOAPOut = 1
server.config.dumpSOAPIn = 1
response = server.insert(agent=soapagent, topic=errortopic, name=errorname, source=sys.argv[0], payload=errorpayload)

sys.exit(1)

if lastbyte != 0:
try:
log.seek(lastbyte) except IOError:
pass

loglines = log.readlines()

lastbyte = log.tell()

log.close()

state = open(statefile, 'w') pickle.dump(lastbyte, state)
state.close()
matchedlines. append (line )

# print 'DEBUG: len->loglines = '+str(len(loglines)) # print 'DEBUG: lastbyte ='
# print '+str(lastbyte) # print 'DEBUG: matchedlines = '+str(matchedlines)

if len (matchedlines ) == 0:
    sys. exit ( 0 )

proxy  = 'https://'+username+ ':'+password+ '@'+instance+'.service-now.com/ecc_queue.do?SOAP'
namespace  = 'http://www.service-now.com/'

server  = SOAPProxy (proxy , namespace ) #server.config.dumpSOAPOut = 1
#server.config.dumpSOAPIn = 1

entriestosend = {} for line  in matchedlines:
    device =line. split () [ 3 ]
    sourceip =line. split () [- 1 ]
    entriestosend [sourceip ] = [device , line ]

for key ,value  in entriestosend. iteritems ( ):
    #impl=getDOMImplementation() #newdoc = impl.createDocument(None, "log_line", None) #top_element = newdoc.documentElement #text = newdoc.createTextNode(value[1])
    #top_element.appendChild(text)

response  = server. insert (agent =soapagent , topic =ecctopic+value [ 0 ] , name =eccname+key , source =eccsource , payload =value [ 1 ] )

---

**Web services C Sharp .NET end to end tutorial**

Examples demonstrating how to use .NET to consume a ServiceNow web service.

This tutorial will show you how to configure ServiceNow correctly to receive a web service request from your .NET client, as well as how to consume our web services using C# .NET.

**Configure C sharp with .NET**

Configure web services within to receive web service requests from a .NET client..
Procedure

1. To configure web services within ServiceNow, access the **System Properties > Web Services** module.

   This module displays the system properties that are specific to web services within your instance. For security reasons, you will want to make sure that you require basic authorization for incoming SOAP requests. This ensures that only authenticated users will be able to make any web services calls, whether it be via web service import sets or inserting/deleting/querying via direct web services.

   ![Web Services](image)

   Please edit your changes and press Save

   **Customization Properties for Web Services**

   - **Require basic authorization for incoming RSS requests**
     - [ ] Yes | No

   - **Require basic authorization for incoming SOAP requests**
     - [ ] Yes | No

2. This next step is very important if you are using .NET as a client to connect to ServiceNow. You must set the **elementFormDefault** property to false.

   This property defines how the WSDLs are qualified. Of course, if you do not consume our WSDL and just create the XML manually, then this property is irrelevant.

   ![Element Form Default](image)

   This property sets the elementFormDefault attribute of the embedded XML schema to the value of unqualified, if set to true. This attribute indicates whether or not locally declared elements must be qualified by the target namespace in an instance document. If the value of this attribute is 'qualified', then locally declared elements must be qualified by the target namespace. For compatibility with clients generated from WSDL (.NET Web Reference, Axle2 stub, webMethods, etc.), set this value to false. This value defaults to true.


   [ ] Yes | No

**Call a web service in visual studio .NET**

Call a web service using Visual Studio 2008.

In this example, we will be using Visual Studio 2008. First, create a new Windows Form Application for this example.
On the resulting form, we created a richTextBox (which we named 'richTextBoxResult') and a button (named buttonResult).
**Use a service reference in a C Sharp integration**

Use a wizard to add a service reference for a C Sharp integration.

Go to the **Solutions Explorer** and select **Service References > Add Service Reference**. A wizard will appear asking for an address. Use: https://<instance name>.service-now.com/incident.do?WSDL. Accept the defaults for the rest of the wizard.

Open the app.config file and change the Security mode to "Transport" and the clientCredentialType and proxyCredentialType to "Basic"

```xml
  <security mode="Transport">
    <transport clientCredentialType="Basic" proxyCredentialType="Basic">
      <realm=""></realm>
      <extendedProtectionPolicy policy Enforcement="Never" />
    </transport>
    <message clientCredentialType="UserName" algorithmSuite="Default" />
  </security>
```

**Use a web reference in a C Sharp integration**

Use a wizard to add a web reference for a C Sharp integration.

Go to the **Solutions Explorer** and select **Service References > Add Service Reference**. A wizard will appear. At the bottom of the form, there is an **Advanced** button. Click on it and click on the **Add Web Reference** button at the bottom of the new wizard page. This will start the **Web Reference** wizard. For the URL, use: https://<instance name>.service-now.com/incident.do?WSDL and name the web reference, ‘WebReference1’. Accept the defaults for the rest of the wizard.

**C Sharp integration source code**

After defining the source code, insert it.

Now we are ready to insert the code. Double-click on the **Send Web Service** button on your form to open the backend code to the form that has been created. Here is the code to insert a record into the demo instance and to read the response.

```csharp
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
```
using System.Text;
using System.Windows.Forms;

namespace ExampleWebServiceForWiki
{
    public partial class FormMain : Form
    {
        public FormMain()
        {
            InitializeComponent();
        }

        private void buttonSend_Click(object sender, EventArgs e)
        {
            /* SERVICE REFERENCE-SPECIFIC CODE */
            ServiceReference1.ServiceNowSoapClient soapClient = new ServiceReference1.ServiceNowSoapClient();
            soapClient.ClientCredentials.UserName.UserName = "itil";
            soapClient.ClientCredentials.UserName.Password = "itil";

            ServiceReference1.insert insert = new ExampleWebServiceForWiki.ServiceReference1.insert();
            ServiceReference1.insertResponse response = new ExampleWebServiceForWiki.ServiceReference1.insertResponse();
            // END OF SERVICE REFERENCE CODE */

            // WEB REFERENCE-SPECIFIC CODE
            soapClient.Credentials = cred;

            WebReference1.insert insert = new WebReference1.insert();
            WebReference1.insertResponse response = new WebReference1.insertResponse();
            // END OF WEB REFERENCE CODE

            insert.category = "Category";
            insert.comments = "Comments";
            insert.short_description = "My short description";

            try
            {
                response = soapClient.insert(insert);
            }

            © 2021 ServiceNow, Inc. All rights reserved.
            ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
C Sharp integration results

If you have followed the tutorial correctly, you should receive the result whether you used a Service Reference or a Web Reference.

Dot net tutorial results

Troubleshoot a null response in a C Sharp integration

Receiving a null response from ServiceNow's web service.

If you are receiving a "null" response from your web service in your client code, then you may have missed the step in this tutorial for setting the elementFormDefault setting to "False".

Remember to recompile your code against the WSDL after you have changed this setting and saved it.
Retrieve a large number of records using SOAP

By default, a single SOAP request can retrieve a maximum of 250 records.

SOAP relies on Extensible Markup Language (XML) as its message format, and usually relies on other Application Layer protocols (most notably Remote Procedure Call (RPC) and HTTP) for message negotiation and transmission. SOAP can form the foundation layer of a web services protocol stack, providing a basic messaging framework upon which web services can be built.

Because of the verbose XML format, SOAP can be considerably slower than other transport methods. Therefore, sending a large amount of data via SOAP is inefficient and is discouraged. Because of this, ServiceNow has imposed a hard-limit of 250 records that can be retrieved at any time in a single query. You may find that this limit poses some technological challenges for your integration design.

SOAP strategies

Retrieve the information that you need and make your integration more efficient.

Use filters to limit the number of results

One way to make your web service calls fit within the 250 record limit is to think about the design of your integrating application.

For example, let's assume that we are making an incident form in C# to show a user the incidents that are assigned to him.

Problematic query approach

The C# application makes a soap call to retrieve all of the incidents within ServiceNow. The application would then store the results locally in memory. When the user decides to view the incidents that are assigned to him, the application loops the internal array and displays the incidents that are assigned to the user.

A better query approach

The C# application makes a soap call to retrieve all of the incidents within ServiceNow that are assigned to the logged-in user. The results are stored locally in memory. When the user decides to view the incidents that are assigned to him, the application shows all the results to the user.
A performance-optimized query approach

The C# application makes no SOAP call initially. When a logged-in user decides to view the incidents that are assigned to him, the application presents him with the choice of viewing active, closed, etc. It gives him the ability to filter the results that he wants to see before the SOAP call is even made. Then, the user is only presented with the results that he wished to view.

Use a local data store to pull data from

If a large amount of data needs to be queried often, and the data does not need to be real-time, perform a sync of the ServiceNow table that you're interested in with your integrating application's data store.

Data push

• Using a scheduled job, ServiceNow can generate a csv/xml from a report and have it emailed to a specific location. The receiver might have a trigger to take the email attachment, parse it, and populate an internal table from which the application can communicate when the data is needed.
• Using a schedule job, ServiceNow can generate a csv/xml from a report and FTP it to an public FTP/FTPS location. The integrating product would consume this csv file on a regular basis and populate an internal table from which the application can communicate when the data is needed.

Note: Currently, the platform does not provide a method for extracting very large amounts of data and sending the output to an FTP server. However, a customization to perform that function is described at here. The customization was developed for use in specific ServiceNow instances, and is not supported by ServiceNow Customer support. The method is provided as-is and should be tested thoroughly before implementation. Post all questions and comments regarding this customization to our community forum.

Data pull

Using a cron job, a machine internal to your network can make a wget call to pull csv/xml data from any table within ServiceNow. The integrating product would consume this csv/xml file on a regular basis and populate an internal table from which the application can communicate when the data is needed. Examples of the wget command that would be used:
• wget --user=itil --password=itil --no-check-certificate https://<instance name>.service-now.com/incident_list.do?CSV

• wget --user=itil --password=itil --no-check-certificate https://<instance name>.service-now.com/incident_list.do?XML

**Use Java/C#/PHP code to fetch the XML data using basic authentication**

If a local data store is not an option, another way to get the data is to call the CSV/XML processor directly and then parse the results.

Use the resulting data in a similar manner as you would a direct SOAP call. An example of this in PHP:

```php
<?php
//This example is in PHP

$user = "itil";
$pass = "itil";
$userPass = $user.':'.$pass;

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, 'https://<instance name>.service-now.com/incident_list.do?CSV');
curl_setopt($ch, CURLOPT_HEADER, 0);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_FOLLOWLOCATION, true);
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
curl_setopt($ch, CURLOPT_USERPWD, $userPass);

$data = curl_exec($ch);
$info = curl_getinfo($ch);

if ($output === false) {
    $output = "No cURL data returned for $addr [". $info['http_code']. "]";
    if (curl_error($ch))
        $output .= "\n". curl_error($ch);
    print $output;
} else{
    echo $data;
}
```
Exporting and converting records into complex data types

Use URL parameters to export table records and convert them into complex data types, such as JSON, XML, PDF, CSV, and XLS.

Exporting records as complex data types

You can use an HTTP GET request to retrieve records from a table and put them in a specified format. For example, use the PDF parameter in a GET request to export records from a table as PDF files; use the XLS parameter to export records from a table as XLS files. For example, to retrieve a list of incident records as XLS files, issue an HTTP GET using the following URL: https://instance_name.service-now.com/incident.do?XLS. The file returned is incident.xls. incident.do is basically a GET that returns a list of the records from the incident table. The XLS parameter converts those records into XLS files.

The general syntax is: https://<serviceNow-instance-name>/<table-name>.do?<Data-type-parameter>

URL parameters

The following table shows URL parameters you can use in GET requests, filters you can use to filter out unwanted table records in the return, and an indicator of whether you can POST the data type directly to a table. The parameter becomes the extension of the returned file, for example, using the XLS parameter returns a file in the form <table-name>.xls.

<table>
<thead>
<tr>
<th>Data type</th>
<th>Parameter</th>
<th>Valid filters</th>
<th>Directly POST to table?</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSV</td>
<td>CSV</td>
<td>sysparm_query, sysparm_view</td>
<td>Y</td>
</tr>
<tr>
<td>Excel</td>
<td>XLS, EXCEL, XLSX</td>
<td>sysparm_query, sysparm_view</td>
<td>Y</td>
</tr>
<tr>
<td>JSON</td>
<td>JSONv2</td>
<td>Various. See JSON data retrieval API.</td>
<td>Y</td>
</tr>
<tr>
<td>PDF</td>
<td>PDF</td>
<td>sysparm_query, sysparm_view</td>
<td>N</td>
</tr>
<tr>
<td>RSS</td>
<td>RSS</td>
<td>sysparm_query, sysparm_view and more.</td>
<td>N</td>
</tr>
</tbody>
</table>
For more information about retrieving and converting table records into the JSON file format, see JSONv2 Web Service.

For more information about retrieving and converting table records into the RSS file format, see RSS feed generator.

**Converting records to PDFs**

For PDF export, there is a distinction between targeting a table and targeting its list. To generate a PDF of a list of records, suffix the target with _list. To target a single record, you must specify the `sys_id` parameter to identify the record for which you are generating the PDF.

**Filters**

All URL parameters work with filters that enable you to export a subset of table records. For example, `sysparm_query=active=true` in a GET request exports only active records. The following example exports only active incident records in an Excel format: https://instance_name.service-now.com/incident.do?EXCEL&sysparm_query=active=true.

The general syntax is: https://<serviceNow-instance-name>/<table_list>.do?<Data-type-parameter>&<filter>

Filters include:

- `sysparm_query`—Filters the data using the encoded query before exporting files, for example, `sysparm_query=active=true` exports only active records.
- `sysparm_view`—Specify the name of a list view to control which fields are returned. For example, to return the ESS view, use `sysparm_view=ess`.
- `useUnloadFormat`—Indicates that the XML format returned is an unload format. The unload format is the same format you get when, from a list in the UI, you select Export > XML > ... You can import unload-formatted XML files back into the tables. To enable the unload format from a URL, use the `useUnloadFormat=true` URL parameter, for example, https://instance_name.service-now.com/incident.do?XML&useUnloadFormat=true.
Example GET queries

<table>
<thead>
<tr>
<th>Data type</th>
<th>Example query</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSV</td>
<td><a href="https://instance_name.service-now.com/incident.do?CSV&amp;sysparm_query=active=true">https://instance_name.service-now.com/incident.do?CSV&amp;sysparm_query=active=true</a></td>
</tr>
<tr>
<td>Excel</td>
<td><a href="https://instance_name.service-now.com/incident.do?XLS&amp;sysparm_query=active=true">https://instance_name.service-now.com/incident.do?XLS&amp;sysparm_query=active=true</a></td>
</tr>
<tr>
<td>PDF</td>
<td><a href="https://instance_name.service-now.com/incident.do?PDF&amp;sysparm_view=ess">https://instance_name.service-now.com/incident.do?PDF&amp;sysparm_view=ess</a></td>
</tr>
<tr>
<td>RSS</td>
<td><a href="https://instance_name.service-now.com/incident.do?RSS&amp;sysparm_view=ess">https://instance_name.service-now.com/incident.do?RSS&amp;sysparm_view=ess</a></td>
</tr>
<tr>
<td>XML</td>
<td><a href="https://instance_name.service-now.com/incident.do?XML&amp;sysparm_query=active=true">https://instance_name.service-now.com/incident.do?XML&amp;sysparm_query=active=true</a></td>
</tr>
</tbody>
</table>

Returned files

GET queries return records from a table in the format specified in the request. For example, a query that uses the XLS parameter returns a table record in a file with the .xls extension.

The Content-Disposition header in the response displays the file name and extension of the returned file. The file name is based on the table you export from, such as incident.xls, incident.pdf, or incident.xml.

Exporting data into tables

You can POST the following data types directly into tables:

- CSV
- Excel
- JSON

The file headers must match the field columns in the targeted table. For more information, see Post CSV or Excel files directly to an import set.

JSONv2 Web Service

The ability to describe sets of data in JSON format is a natural extension to the JavaScript language.
ServiceNow supports a web service interface that operates on the JSON object as the data input and output format.

The JSONv2 web service is provided by a platform-level processor similar to the services for SOAP, WSDL, CSV, Excel, and XML. Like those services, the JSON web service is triggered by a standalone JSONv2 URL parameter. For example:

```
https://<instance name>.service-now.com/mytable.do?JSONv2
```

Having the JSON object available as a data format for web services means that you can create (insert), update, and query any data in the system using the JSON object format, and get results in the JSON object format.

**Security**

Like all other HTTP-based web services available on the platform, the JSONv2 web service is required to authenticate using basic authentication by default. The user ID that is used for authentication is subjected to access control in the same way as an interactive user.

**Related information**

- SOAP web services security
- JSON object format

**JSON object format**

The JSON object is built in two structures.

- A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
- An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

In its simplest form, a JSON object is just a comma delimited set of name/value pairs. For example:

```
{"name one":"value one","name two":"value two"}
```

The following is a sample of a single record array of incidents in JSON:

```
{"records":
[["closed_by":null,
  "__status": "success",
  "category":"inquiry",
  "escalation":0,
  "state":1,
  "location":null,
  "reassignment_count":0,
  "time_worked":null],
```

© 2021 ServiceNow, Inc. All rights reserved. ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
The following is a record array of incident responses with an error.

```json
{
  "records": [
    {
      "__error": {
        "message": "Invalid Insert into: incident",
        "reason": "Data Policy Exception:  Short description is mandatory ",
      },
      "__status": "failure",
      "active": "true",
      "activity_due": "",
      "approval": "not requested",
      "approval_history": "",
      "approval_set": "",
      "assigned_to": "",
      "assignment_group": "d625dceca0a8016700a222a0f7900d06",
      "business_duration": "",
      "business_stc": ""
    }
  ]
}
```
JSON response status

JSONv2 requests may return one of several response statuses.

**JSON Success Response**

Each JSON success response includes a record array containing the records retrieved by the given action. Each JSON object contains one or more metadata elements, prefixed with __, regarding the status for the action on each record, as illustrated in the previous examples. The JSON success responses use the following syntax:

```
__status

"__status": "<value>"
```

where <value> is success or failure.
**JSON Failure Response**
When the _status element returns failure, the _error element is added to identify the error and reason.

```
"_error": { "message": "<error value>", "reason": "<reason value> "}
```

where `<error value>` is the error message text and `<reason value>` is the reason the error was triggered.

The JSON error response contains only the error and reason elements. Generally, this indicates that the whole JSON operation failed and no records can be processed.

For example:

```
{"_error":"Cannot update with empty sysparm_query","reason":null}
```

**Setting the number of rows returned**
The following system property controls how many rows JSON returns with each query.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>glide.processor.json.row_limit</code></td>
<td>Specify the maximum number of rows a JSON query returns.</td>
</tr>
<tr>
<td></td>
<td>• Type: integer</td>
</tr>
<tr>
<td></td>
<td>• Default value: 10,000</td>
</tr>
<tr>
<td></td>
<td>• Location: Add to the System Properties [sys_properties] table</td>
</tr>
</tbody>
</table>

**Requiring basic authentication for incoming JSONv2 requests**
The following system property controls whether basic authentication is required for incoming JSONv2 requests.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>glide.basicauth.required.jsonv2</code></td>
<td>Enables (true) or disables (false) requiring basic authentication for incoming JSONv2 requests.</td>
</tr>
</tbody>
</table>
Requiring Basic Authentication for Incoming JSONv2 Requests (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Type: true</td>
</tr>
<tr>
<td></td>
<td>• Default value: true</td>
</tr>
<tr>
<td></td>
<td>• Location: Add to the System Properties [sys_properties] table</td>
</tr>
</tbody>
</table>

Note: To learn more about this property, see Basic auth: JSONv2 requests in Instance Security Hardening Settings.

Action parameters

Action parameters are separate and different from data parameters because they specify the action to take when the JSON object parameter is part of an HTTP GET or POST request.

The parameters can also be specified as a field in the supplied JSON object. They have the effect of triggering an action in the case of sysparm_action, or filtering the results of an update or query in the case of sysparm_query.

sysparm_action

The following are the valid values for sysparm_action and the corresponding action triggered by the API.

<table>
<thead>
<tr>
<th>Method Summary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getKeys</td>
<td>Query the targeted table using an encoded query string and return a comma delimited list of sys_id values.</td>
</tr>
<tr>
<td>getRecords</td>
<td>Query the targeted table using an encoded query string and return all matching records and their fields.</td>
</tr>
<tr>
<td>get</td>
<td>Query a single record from the targeted table by specifying the sys_id in the sysparm_sys_id URL parameter, and return the record and its fields.</td>
</tr>
</tbody>
</table>
Data Modification

<table>
<thead>
<tr>
<th>Method Summary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>insert</td>
<td>Create one or more new records for the table targeted in the URL.</td>
</tr>
<tr>
<td>insertMultiple</td>
<td>Create multiple new records for the table targeted in the URL.</td>
</tr>
<tr>
<td>update</td>
<td>Update existing records in the targeted table in the URL, filtered by an encoded query string.</td>
</tr>
<tr>
<td>deleteRecord</td>
<td>Delete a record from the table targeted in the URL by specifying its sys_id in the sysparm_sys_id URL parameter.</td>
</tr>
<tr>
<td>deleteMultiple</td>
<td>Delete multiple records from the table targeted in the URL, filtered by an encoded query string.</td>
</tr>
</tbody>
</table>

**sysparm_query**

Specify an encoded query string to be used in get, getRecords, update or deleteMultiple sysparm_action value.

**sysparm_view**

Specify a form view to customize the return values for get and getRecords function calls. When using a view, the query returns only the fields defined in the view, including referenced values. If there is no view name, or if the view name is not valid, then the query returns all field names that are marked active in the dictionary.

**sysparm_sys_id**

Specify a target sys_id during a get or delete function call (sysparm_action value).

**sysparm_record_count**

Specify an integer value to limit the number of records retrieved for this request. Note that this value is capped by the glide.processor.json.row_limit system property.

**displayvalue**

Get the display value of a reference field, if any are in the record. For example, the Incident record can have an assigned_to field that is a reference to a user record. Instead of sending the sys_id of the user record, the user name is sent.

The displayvalue parameter can have three values: true, false, or all.
• **true**: All the references fields show the display value instead of `sys_id`.

• **false** (default): All reference fields show `sys_ids`.

• **all**: The display value and the `sys_id` are shown. For example, the `assignedto` field in the Incident record is sent back as `assigned_to:1234556, dv_assigned_to:Fred Luddy`.

**displayvariables**

Set this boolean value to **true** during a get or getRecords function call to retrieve all variables attached to this record.

**JSON Data Retrieval API**

Query for data by issuing an HTTPS GET request to the instance.

By default, a GET request is interpreted as a get function if a `sysparm_sys_id` parameter is present. Otherwise, it is interpreted as a getRecords function. You can also specify a URL parameter `sysparm_action=get`. Query responses are always encapsulated by a records hash of records, where each individual record's values are themselves hashed by field name.

**Return Display Value for Reference Variables**

When you are getting a record from a get or getRecords function, all the fields associated with that record are returned. The fields are often reference fields that contain a `sys_id` for another table. The base system behavior is to return the `sys_id` value for those fields. To have the display value for the field returned, use one of these options:

• Add the property `glide.json.return_displayValue` to the system properties, and every JSON request will return a display value for a reference field.

• Add the parameter `displayvalue=true` to the JSON request URL and JSON requests with that parameter will return a display value instead of the `sys_id` for a reference field. The JSON URL would look like this:

```
https://<instance_name>.service-now.com/incident.do?JSON&sysparm_action=getRecords&sysparm_query=active=true^category=hardware&displayvalue=true
```

• Add the parameter `displayvalue=all` to the JSON request URL and JSON requests with that parameter return a display value and the `sys_id` for a reference field. The response element name for the display value field will be prefixed with `dv_`, for example `dv_caller_id`.
Get variables

Use the displayvariables query parameter to return an array of variables associated with a Service Catalog item record. To get variables, add the parameter displayvariables=true to the JSON request URL. For example, here is a URL to retrieve a record in JSON format that includes Service Catalog variables:


Here is the example response that displays a multi-row variable set from the record:

```json
{
  "records": [
    {
      ...
      "variables": [
        {
          "display_value": [
            {
              "quantity": "1",
              "color": "Black",
              "device_type": "Apple iPhone 8",
              "storage": "64GB"
            },
            {
              "quantity": "1",
              "color": "Black",
              "device_type": "Apple iPhone 8",
              "storage": "64GB"
            }
          ],
          "columns_meta": [
            {
              "name": "device_type",
              "label": "Device Type",
              "id": "da7d3f3241411300964ff05369414eca",
              "type": 5,
              "order": 0
            },
            {
              "name": "storage",
              "label": "Storage",
              "id": "691e337241411300964ff05369414e31"
            }
          ]
        }
      ]
    }
  ]
}
```
"type": 5,
"order": "1"
},
{
  "name": "color",
  "label": "Color",
  "id": "e89fb7724141300964ff05369414e74",
  "type": 5,
  "order": "2"
},
{
  "name": "quantity",
  "label": "Quantity",
  "id": "2d5f73724141300964ff05369414eaf",
  "type": 5,
  "order": "3"
}
],
"max_rows": 50,
"name": "mobile_devices_set",
"id": "e84d3f324141300964ff05369414e3e",
"type": "one_to_many",
"value": [
  {
    "quantity": "1",
    "color": "black",
    "device_type": "iphone8",
    "storage": "64GB"
  },
  {
    "quantity": "1",
    "color": "black",
    "device_type": "iphone8",
    "storage": "64GB"
  }
],
"row_count": 2
},
{
  "question_text": "Department",
  "name": "department",
  "type": 8,
  "value": "Development",
  "order": 100
The keys in the response are defined as follows:

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>display_value</td>
<td>Multi-row variable set question display value. Only returned with multi-row variable sets.</td>
</tr>
<tr>
<td>columns_meta</td>
<td>Array of multi-row variable set metadata, such as the sys_id and name of the field. Only returned when the variable contains multiple fields.</td>
</tr>
<tr>
<td>max_rows</td>
<td>Maximum rows allowed in the multi-row variable set. Only returned with multi-row variable sets.</td>
</tr>
<tr>
<td>name</td>
<td>Question name.</td>
</tr>
</tbody>
</table>
### Key Description

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Sys_id of the multi-row variable set. Only returned with multi-row variable sets.</td>
</tr>
<tr>
<td>type</td>
<td>Type of question.</td>
</tr>
<tr>
<td>value</td>
<td>Question value.</td>
</tr>
<tr>
<td>row_count</td>
<td>Current number of rows in the multi-row variable set. Only returned with multi-row variable sets.</td>
</tr>
<tr>
<td>question_text</td>
<td>Question label. Only returned with single-row variable sets.</td>
</tr>
<tr>
<td>order</td>
<td>Order of the question.</td>
</tr>
</tbody>
</table>

### Control the order of records

You can control the order that records appear in the JSON response. To set an order, use the **ORDERBY** or **ORDERBYDESC** clauses in the URL encoded query. For example,

```
sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory
```

filters all active records and orders the results in ascending order by number first, and then in descending order by category. For more information, see Encoded query strings.

### getKeys

Get the **sys_id** of multiple records by specifying an encoded query string in the **sysparm_query** parameter.

```
https://<instance name>.service-now.com/incident.do?JSONv2&sysparm_action=getKeys&sysparm_query=active=true^ORDERBYnumber^ORDERBYDESCcategory
```

### get

Get a record directly by specifying the **sys_id** in a **sysparm_sys_id** parameter.

```
https://<instance name>.service-now.com/incident.do?JSONv2&sysparm_sys_id=9d385017c611228701d22104cc95c371
```

Optionally, you may also specify the **sysparm_action** parameter:

```
https://<instance name>.service-now.com/incident.do?JSONv2&sysparm_action=get&sysparm_sys_id=9d385017c611228701d22104cc95c371
```
getRecords
Get all records by specifying an encoded query string in the `sysparm_query` parameter.

```plaintext
https://<instance name>.service-now.com/incident.do?JSONv2&sysparm_action=getRecords&sysparm_query=active=true^category=hardware
```

JSON Data Modification API
Modify data using the JSON web service by sending an HTTPS POST request to the instance.

The HTTP POST must contain a `sysparm_action` parameter to indicate the type of action to be performed, with the incoming JSON object post in the body.

⚠️ Note: The content-type of the POST should be application/json. It cannot be application/x-www-form-urlencoded or multipart/form-data.

insert
Create a new record in ServiceNow. The JSON object has to be POSTed as the body (content-type is usually application/json, although not enforced). The response from the record creation is a JSON object of the incident that was created.

For example, posting the following JSON object:

```plaintext
{"short_description":"this is a test","priority":"1"}
```

to the following URL:

```plaintext
https://your_instance.service-now.com/incident.do?JSONv2&sysparm_action=insert
```
creates an incident.

Optionally, you may also specify the `sysparm_action` in the JSON object. The parameter inside the JSON object takes precedence over the URL parameter. For example:

```plaintext
{"sysparm_action":"insert","short_description":"this is a test","priority":"1"}
```

insertMultiple
To create multiple new records in ServiceNow, the input JSON object for the `insert` function must be an array. The response from the record creation is a JSON object of the incidents that were created. For example, the following JSON object:
posted to one the following URLs:

https://<instance name>.service-now.com/incident.do?JSONv2&sysparm_action=insert
https://<instance name>.service-now.com/incident.do?JSONv2&sysparm_action=insertMultiple

creates two incidents. Note the fields described as an array value for the **records** field.

**update**

Update a record or a list of records filtered by an encoded query string specified by the sysparm_query URL parameter. The JSON object has to be posted as the body (content-type is usually application/json, although not enforced). The response from the record creation is an array of JSON objects representing the records that were updated.

For example, posting the following JSON object:

```json
{ "short_description":"this was updated with python", "priority": "3", "impact":"1" }```

to the following URL:

https://instance_name.service-now.com/incident.do?JSONv2&sysparm_query=priority=3&sysparm_action=update

updates all incidents with priority 3, and sets the values specified by the JSON object.

**deleteRecord**

Delete a single record from the targeted table, identified by a sysparm_sys_id parameter. The parameter may be encoded in the input JSON object or given as a URL parameter.

For example, posting:

```json
{"sysparm_sys_id":"fd4001f80a0a0b380032ffa2b749927b"}"
```

to the following URL:

http://instance_name.service-now.com/incident.do?JSONv2&sysparm_action=deleteRecord

deletes the incident record identified by the sys_id fd4001f80a0a0b380032ffa2b749927b.
**deleteMultiple**

Delete multiple records from the targeted table, filtered by an encoded query string specified in the `sysparm_query` URL parameter. The filter may also be encoded in the input JSON object.

For example, posting:

```json
{"sysparm_query":"short_description=this was updated with python"}
```

to the following URL:

http://instance_name.service-now.com/incident.do?JSONv2&sysparm_action=deleteMultiple

deletes all incident records where the `short_description` field contains the value "this was updated with python".

**RSS web service**

RSS (Rich Site Summary) is a format for delivering web-based information that changes regularly.

**RSS feed generator**

ServiceNow supports the dynamic generation of RSS feeds.

Much like our Web Services implementation, the retrieval of an RSS feed representation of information is simply done by specifying an RSS parameter at the end of the URL to a table list. For example, the following will return a list of all incidents in RSS 2.0 format:

**Adding a Query**

To associate a query to the list so that a filtered list is returned, use the `sysparm_query` parameter. For example, the following will return a list of all incidents where the priority field is 1 (Critical):

```text
https://<instance name>.service-now.com/incident.do?sysparm_query=priority=1&RSS
```

If you have a multi part query then you would separate the parts with the `^` character. For example to get all priority 1 incidents with a category of software you would:

```text
https://<instance name>.service-now.com/incident.do?sysparm_query=priority=1^category=software&RSS
```

If you want to query on a field that is a reference to another file then you need to use javascript to resolve the reference to the other file. For example, the `assigned_to` field in incident is a reference to a user record. If you wanted to find all the incidents assigned to "ITIL User" then you would do the following:
Note: You can in most cases simply append "&RSS" to a URL that you generate in the U.I. or that of your favorite module. The easiest way to get the URL is to simply click the last breadcrumb from the list view. After appending "&RSS" then you can use this URL in your RSS feed reader.

Get Bread URL

Limiting results with a view

The description element in the returned RSS xml is constructed using the view as specified in the URL, when no view is specified, the default no-name view is used.

To change this format, specify the `sysparm_view` parameter on the URL. For example, the following request will return the incidents list. However the result will be restricted to only the fields available in the ess view:

https://<instance name>.service-now.com/incident.do?sysparm_query=priority=1&sysparm_view=ess&RSS

Additionally, the RSS item title can be modified using the `sysparm_title_view` URL parameter. When specified, the item title will be constructed using the fields specified in the view. For example:


Formatting results

The description element in the returned RSS xml can be formatted by setting the URL parameter `sysparm_format=true` and specifying the format string in the property `glide.rss.description_format`.

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
By default, when the URL parameter is present, the description element will be formatted to contain the field label and value using the following format string:

```
<b>{1}</b>: {2}<br/>
```

- `{0}` - field name
- `{1}` - field label
- `{2}` - field value

This default format string can be overridden using the property `glide.rss.description_format`. An example of the formatted RSS feed can be seen in the following screen capture from Firefox:

**RSS Format**

**Problem (All)**

<table>
<thead>
<tr>
<th>Problem (All)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PRB00001</strong></td>
</tr>
<tr>
<td>Sun, Aug 10, 2008 3:39 PM</td>
</tr>
<tr>
<td><strong>Number</strong>: PRB00001</td>
</tr>
<tr>
<td><strong>Escalation</strong>: Moderate</td>
</tr>
<tr>
<td><strong>Short description</strong>: Windows xp SP2 causing errors in Enterprise</td>
</tr>
<tr>
<td><strong>Problem state</strong>: Pending Change</td>
</tr>
<tr>
<td><strong>RFC</strong>: CHG00003</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>PRB00008</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sun, Aug 10, 2008 3:40 PM</td>
</tr>
<tr>
<td><strong>Number</strong>: PRB00008</td>
</tr>
<tr>
<td><strong>Escalation</strong>: Normal</td>
</tr>
<tr>
<td><strong>Short description</strong>: Hang when trying to print VISIO document</td>
</tr>
<tr>
<td><strong>Problem state</strong>: Open</td>
</tr>
<tr>
<td><strong>RFC</strong>:</td>
</tr>
</tbody>
</table>

**RSS basic authentication**

To enforce basic authentication on each request for an RSS feed, set the property `glide.basicauth.required.rss` to `true`.

RSS request would have to contain the Authorization header as specified in the Authentication protocol. Because the request is non-interactive, we always require the `Authorization` header during a request.

⚠️ **Note**: If you plan to disable RSS basic authentication, make sure that tables in the platform have the right ACL entries to protect from unauthorized access.
To specify basic authentication on the URL, put the username and password pair separated by a colon in front of the server name before an @ character. For example, to submit the demo credentials for the ITIL user, use the following URL.

```
https://itil:itil@<instance name>.service-now.com/incident.do?RSS
```

Some older browsers, such as Microsoft IE 7 do not support direct URL authentication. If the site uses basic authentication, Internet Explorer automatically prompts users for a user name and a password. In some cases, users can click the Remember my password box in the prompt to save their credentials for later visits to that site.

**Related information**
- [http://www.w3.org/Protocols/HTTP/1.0/draft-ietf-http-spec.html#BasicAABasic](http://www.w3.org/Protocols/HTTP/1.0/draft-ietf-http-spec.html#BasicAABasic)
- [http://support.microsoft.com/kb/834489](http://support.microsoft.com/kb/834489)
- RSS request authorization

**RSS title override**

You may optionally override the automatically generated title of the RSS feed by added the `sysparm_title` parameter to the request URL.

For example, you can specify the title **Priority One Incidents** using the following request URL.

```
```

This will produce results as follows:
RSS Out

<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE rss version="2.0">
<title>Priority One Incidents</title>
<channel>
  <link>http://www.service-now.com/demo/nav_to.do?url=incident.do?sysparm_query=priority=1</link>
  <description>Priority One Incidents</description>
  <copyright>2006 Service-now.com</copyright>
  <pubDate>Mon, 12 Jun 2006 11:04:44 PDT</pubDate>
  <lastBuildDate>Mon, 12 Jun 2006 11:04:44 PDT</lastBuildDate>
  <generator>rssGenerator by Enrique A. Vicelli</generator>
<item>
  <title>IN 000009</title>
  <link>http://www.service-now.com/demo/nav_to.do?url=incident.do?sys_id=46b66e409f1e19b101f243dfc79033d0%26sysparm_stack=incident_list_do%3Fsasysparm_query=active=true</link>
  <description>IN000009 2006-02-01 14:50:22 Reset my password</description>
  <author>glide maint</author>
  <guid>46b66e409f1e19b101f243dfc79033d</guid>
  <pubDate>Wed, 17 May 2006 18:20:57 PDT</pubDate>
</item>
</channel>
</rss>

RSS feed reader

Create a scrolling RSS feed reader using a UI page.

You must create a feed parser using an RSS API, such as the Google Feed API. Developer's guide: http://code.google.com/apis/ajaxfeeds/documentation/API: http://code.google.com/apis/ajaxfeeds/documentation/reference.html

Scrollable areas

A scrollable area is a div where contents scroll from the bottom up over time.

You can scroll any HTML content, and anything inside the scroller is operational HTML with functioning links and images.

To make a scrollable areas, wrap the scrolling content inside of a scrollable_area tag, likely in a UI Page:

```html
<g:scrollable_area height="100px">
  <g:evaluate var="jvar_temp" expression="var kb = new GlideRecord('kb_knowledge');"/>
  <g:inline template="kb_section.xml"/>
</g:scrollable_area>
```

The system will then create a 100 pixel high div and the contents will automatically scroll from bottom to top. If you have a 1000 pixel high block of text, for example, you'll see the top 100 pixels and then pixels 2-101, then 3-102, etc. Once it reaches the top it'll wrap back around to the bottom.
This sample code will create a scroller with a list of priority 1 incidents.

```xml
<?xml version="1.0" encoding="utf-8" ?>
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null" xmlns:g2="null">
  <g2:evaluate var="jvar_inc">
    var inc = new GlideRecord('incident');
    inc.addActiveQuery();
    inc.addQuery('priority',1);
    inc.query();
  </g2:evaluate>

  <g2:scrollable_area height="100px">
    <table border="0" cellspacing="2" cellpadding="0" width="100%">
      <j2:while test="$[inc.next()]">
        <j2:set var="jvar_inc_link" value="incident.do?sys_id=$[inc.sys_id]"/>
        <j2:set var="jvar_inc_list_link" value="incident_list.do?sysparm_query=active=true"/>
        <tr>
          <td>
            <a href="${jvar_inc_link}">
              <IMG SRC="images/services.png" style="padding-right:10px"></IMG>
            </a>
            <a href="${jvar_inc_link}" style="padding-right:10px; color:blue">${inc.number}</a>
          </td>
          <td>${inc.short_description}</td>
        </tr>
      </j2:while>
    </table>
    <tr>
      <td align="center" colspan="2"><a href="${jvar_inc_list_link}" style="color:blue">View all active incidents</a></td>
    </tr>
  </g2:scrollable_area>
</j:jelly>
```

Add scrolling elements in forms

You can add scrolling areas to forms as well as UI pages.

**Procedure**

1. Create a UI Macro with the script.
2. Create a Formatter to reference the script.
Priority 1 incidents example

This example scrolling element demonstrates how to create a UI macro to a scrolling list of priority 1 incidents.

Use the following example code:

```xml
<?xml version="1.0" encoding="utf-8" ?>
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null" xmlns:g2="null">
    <g2:evaluate var="jvar_inc">
        var inc = new GlideRecord('incident');
        inc.addActiveQuery();
        inc.addQuery('priority',1);
        inc.query();
    </g2:evaluate>

    <div style="background-color:DDDDDD; padding-left:10px; line-height:19px; border:2px white solid" width="100%" nowrap="true">
        Priority 1 Incidents:
        <input id="make_spacing_ok" style="visibility:hidden;width:0px:" title=""/>
    </div>

    <g2:scrollable_area height="100px" width="100%">
        <j2:while test="$[inc.next()]">
            <j2:set var="jvar_inc_link" value="incident.do?sys_id=$[inc.sys_id]"/>
            <j2:set var="jvar_inc_list_link" value="incident_list.do?sysparm_query=active=true"/>
            <span style="line-height: 10px; padding-left:10px">
                <a href="$[jvar_inc_link]">
                    <img src="images/services.png" style="padding-right:10px"></img>
                </a>
                <a href="$[jvar_inc_link]" style="padding-right:10px; color:blue">$[inc.number]</a>
            </span>
            $[inc.short_description]
            <br style="line-height:5px"/>
        </j2:while>
    </g2:scrollable_area>
</j:jelly>
```
Navigate to **System UI > Formatters** and create a Formatter that refers to the UI Macro above.

**Add a scrolling news panel to your homepage**
Display knowledge article content in a scrolling news panel.

**Before you begin**
Create knowledge content to display in the news panel.

**Procedure**
1. Navigate to a homepage.
2. Click **Add Content** to get a list of possible content types.
3. Click **Scrollers** in the leftmost box of the add content dialog.
4. Click the news scroller.
5. Click **Add** to add it on your page.

![News scroller example](image)

**News in the scroller**
The news scroller gets its news list by going to the Knowledge Base and querying the short descriptions of any items there with a topic of "News".

To add news, go to the Knowledge Base and add new entries with a topic of "News".

**Style Control**
If you want to make a headline standout in the news scroller, use HTML tags on the short description. For example, enter `<b>My Bold Test</b>` and **My Bold Test** will appear in the scroller.

ℹ️ **Note:** The property `glide.ui.escape_text` must be `false` to use HTML formatting.
Scrolling News Panel Notes and Limitations

Scrollable news panels offer a way for you to display high impact news items on your users home pages in an eye catching scrolling news display.

- Scrolling is currently always from the bottom to the top.
- Scrolling is 1 pixel a at a time every 100 ms.
- You can have as many scrollable areas on a screen as you want (they get system generated unique names).
- Works in Internet Explorer and Firefox.
- If the object you are scrolling is shorter than the scrollable area, it still scrolls.
- If you mouse over a scrollable area, it stops scrolling (so you can click something without chasing it as it scrolls).

RSS feed reader example

An example of how to set up an RSS feed reader using an RSS feed.
For an example RSS feed reader, refer to the Now Community.

REST & SOAP API analytics

REST & SOAP API analytics allow you to track and analyze web service API usage.

Use analytics to help answer questions such as
- Which APIs are used most?
- Which API versions are being used? Can I deprecate older versions?
- Which API methods are being used?
- What resources are being accessed?
- Who is using each API and resource?

The instance tracks analytics for all inbound web services, including platform web services such as the REST Table API or the SOAP API, and custom web services such as scripted REST APIs and scripted SOAP web services. Analytics are aggregated by each resource and HTTP action combination.

Outbound web services, such as REST Messages, are not tracked.

Collect analytics for an API

Select which APIs to collect analytics for and select if requestor information should be collected for each API.
Before you begin

Note: API analytics is not available for certain APIs used for internal platform functionality, such as the UI and Mobile app APIs.

Role required: api_analytics_read or admin

About this task

When a new API is added, an inclusion list record is created automatically. You can modify or create new inclusion list records to manually configure which APIs and requestor information are logged.

Procedure

Navigate to System Web Services > Inclusion List and create a new record.

API Transactions Stats Inclusion Lists fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API name</td>
<td>For REST APIs, enter the API name. This name should contain only the namespace and ID of the API. Do not include any request parameters in the API name. For example, for the Table API endpoints api/now/table/incident and api/now/table/problem, the namespace and ID are now/table. SOAP APIs do not support separate logging configurations for different APIs. You can configure logging for all SOAP APIs by modifying the SOAP APIs inclusion list record.</td>
</tr>
<tr>
<td>Collect API stats</td>
<td>Select this check box to track analytics for the specified API.</td>
</tr>
<tr>
<td>Collect API requestor stats</td>
<td>Select this check box to track which users make requests to this API. This option is available only if Collect API stats is selected.</td>
</tr>
</tbody>
</table>

REST & SOAP API analytics dashboards

You can view API analytics for the entire instance or for specific APIs using the included dashboards.

REST & SOAP API Analytics includes dashboards that present the overall analytics, analytics per API, and analytics per requesting user.
To access API Analytics dashboards, navigate to System Web Services and select Usage Overview, Usage by API, or Usage by Requestor. Users with the api_analytics_read or admin roles can view these dashboards.

Note: The Usage by Requestor dashboard uses the Responsive Canvas feature available with Istanbul. For the best experience on upgraded instances, you may need to enable responsive dashboards.

The following dashboards are available:

### Available API analytics dashboards

<table>
<thead>
<tr>
<th>Dashboard</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usage Overview</td>
<td>Provides general usage statistics for all REST and SOAP APIs.</td>
</tr>
<tr>
<td>Usage by Web API</td>
<td>Provides detailed usage statistics for each API.</td>
</tr>
<tr>
<td>Usage by Requestor</td>
<td>Provides detailed usage statistics for each requesting user.</td>
</tr>
<tr>
<td></td>
<td>You can additionally filter by API to view detailed stats for a specific user and API combination. If the selected user has not made requests to the selected API, no data is shown for that combination.</td>
</tr>
</tbody>
</table>

Note: To view API stats for all users, do not clear the user breakdown selection. Instead, use the Usage by Web API dashboard.

You can access API analytics from the REST API Explorer. When exploring an API, click the context menu icon (⋮) then select API analytics to view analytics for that specific API.

For custom web services, such as scripted REST APIs or scripted SOAP web services, you can access analytics for the API by clicking the API analytics related link on the scripted REST Service or scripted Web Service forms.

When you directly access the analytics for an API, if there are no analytics for that specific API, the analytics for all APIs appears.
Related information

Use the REST API Explorer

REST and SOAP API analytics collection and cleanup

API analytics uses scheduled jobs to collect and clean up analytics data. The instance tracks all web service transactions for APIs on the inclusion list and maintains a daily history, aggregated by resource and HTTP action combination. Requester information is aggregated per requester, resource, and HTTP action combination and tracked up to the daily limit defined by the property `com.glide.api.stats.daily_limit`.

Refer to the following table to determine which requests are logged.

<table>
<thead>
<tr>
<th>Whitelist API name</th>
<th>Example resource</th>
<th>Response code</th>
<th>Description</th>
<th>Logged</th>
</tr>
</thead>
<tbody>
<tr>
<td>now/table</td>
<td>/api/now/table/incident</td>
<td>Any except 401</td>
<td>Valid resource and table</td>
<td>Yes</td>
</tr>
<tr>
<td>now/table</td>
<td>/api/now/table/invalidResource</td>
<td>400</td>
<td>Valid resource but an invalid table</td>
<td>Yes</td>
</tr>
<tr>
<td>now/table</td>
<td>/api/now/table/incident</td>
<td>403</td>
<td>Requesting user has insufficient privileges</td>
<td>Yes</td>
</tr>
<tr>
<td>now/table</td>
<td>/api/now/table/incident</td>
<td>401</td>
<td>Requesting user is not authenticated</td>
<td>No</td>
</tr>
<tr>
<td>myApp/myScriptedApi</td>
<td>myApp/myScriptedApi/myResource</td>
<td>Any except 401</td>
<td>Valid resource</td>
<td>Yes</td>
</tr>
<tr>
<td>myApp/invalidApiName</td>
<td>• myApp/invalidApiName • myApp/invalidApiName/myResource</td>
<td>400</td>
<td>Invalid API, even with a matching inclusion list entry</td>
<td>No</td>
</tr>
</tbody>
</table>

On the 2nd of each month, the API Monthly Stats scheduled job calculates the monthly total for each resource and HTTP action combination. Each day the API Monthly Requestor Stats scheduled job calculates the monthly total for each
resource, requester, and HTTP action combination based on daily scores older than 2 days.

Daily statistics are maintained for 33 days. Monthly totals are maintained for 13 months. Table cleaners for the sys_api_stats, sys_api_stats_requestor, and sys_api_stats_requestor_monthly tables remove analytics records older than these limits.

**REST & SOAP API analytics naming**

The **API Name** used when tracking API analytics is determined by the type of API being described, such as a REST API or a Scripted SOAP service.

<table>
<thead>
<tr>
<th>API type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST</td>
<td>The API namespace and the first part of the URI following the namespace is used as the API name. For example, for the Table API endpoints api/now/table/incident and api/now/table/problem, the namespace and ID are now/table.</td>
</tr>
<tr>
<td>Direct SOAP (table does not extend Import Set Row table)</td>
<td>If the direct SOAP request accesses a table, Direct SOAP is used as the API name.</td>
</tr>
<tr>
<td>SOAP import (table extends Import Set Row table)</td>
<td>Import Set SOAP is used as the API name.</td>
</tr>
<tr>
<td>Scripted SOAP Services</td>
<td>The SOAP request endpoint page is used as the API name, such as my_service.do.</td>
</tr>
</tbody>
</table>

**API analytics properties**

Certain properties control the behavior of API analytics.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.glide.api.stats.enabled</td>
<td>When true, enables the collection of API usage statistics. When false, no analytics are collected even for domains on the inclusion list.</td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>--------------</td>
<td>----------------</td>
</tr>
<tr>
<td>com.glide.api.stats.max_bytes_nursery_size</td>
<td>The maximum amount of memory, in bytes, used to store transaction data before it is written to the log. Transaction data is written to the log regularly based on the value of <code>com.glide.api.stats.persist_interval</code>. If a large volume of transactions exceeds this memory limit before the log is written, some transactions may not be logged. The event <code>api.stats.cache.size.reached</code> is fired if this limit is reached. This property value must be between 1 and 3 megabytes.</td>
</tr>
<tr>
<td>com.glide.api.stats.persist_interval</td>
<td>The frequency, in seconds, for writing transactions stored in memory to the log. This property value must be between 30 and 120 seconds.</td>
</tr>
<tr>
<td>com.glide.api.stats.daily_limit</td>
<td>The daily limit of requestor stats records per instance node. As soon as the value is reached, data is no longer aggregated and</td>
</tr>
</tbody>
</table>
Outbound REST web service

ServiceNow outbound REST functionality allows you to retrieve, create, update, or delete data on a web services server that supports the REST architecture.

A REST message can be sent by a REST workflow activity or by using the RESTMessageV2 script API. You can run REST messages from a MID Server which allows the message to communicate with REST providers on an internal network.

ServiceNow REST functionality is flexible enough to accommodate many web service APIs. Be sure you are familiar with your web service and the parameters it accepts before attempting to define a REST message in ServiceNow.

REST message elements

An outbound REST message is composed of several elements, such as the endpoint and HTTP methods.

A REST message contains the following elements:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api.stats.requestor.daily.limit.reached</td>
<td>stored for that day. The event is fired if this limit is reached.</td>
</tr>
<tr>
<td></td>
<td>• Type: integer</td>
</tr>
<tr>
<td></td>
<td>• Default value: 20000</td>
</tr>
<tr>
<td></td>
<td>• Location: Add to the System Properties [sys_properties] table</td>
</tr>
<tr>
<td>glide.api.stats.debug</td>
<td>When true, enables debug logging for API stats.</td>
</tr>
<tr>
<td></td>
<td>• Type: true</td>
</tr>
<tr>
<td></td>
<td>• Default value: false</td>
</tr>
<tr>
<td></td>
<td>• Location: Add to the System Properties [sys_properties] table</td>
</tr>
</tbody>
</table>
### Elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Endpoint</td>
<td>The endpoint is the URL of the data to be retrieved, updated, or deleted. Every REST message must specify an endpoint.</td>
</tr>
<tr>
<td>Headers</td>
<td>HTTP headers in REST messages contain information about the request, such as the desired response format. A REST message may specify any number of headers.</td>
</tr>
<tr>
<td>Authentication</td>
<td>Authentication settings include which type of authentication to use, such as basic auth or OAuth, as well as the credentials to use.</td>
</tr>
<tr>
<td>settings</td>
<td>HTTP methods, such as GET, POST, or DELETE interact with the data at the endpoint.</td>
</tr>
<tr>
<td>HTTP methods</td>
<td>You can optionally override the parent REST message configuration in each HTTP method such as by specifying a different endpoint, authentication credentials, or headers.</td>
</tr>
<tr>
<td></td>
<td>HTTP methods that send content, such as POST, include a message body detailing this content.</td>
</tr>
<tr>
<td></td>
<td>A REST message may specify multiple HTTP methods. When sending a REST message, such as through a workflow activity or script, you must specify which HTTP method to use.</td>
</tr>
</tbody>
</table>

### Create a REST message

You can send requests to a REST web service endpoint by creating a REST message record.

#### Before you begin

Role required: web_service_admin

#### Procedure

1. Navigate to **System Web Services > REST Message**.
2. Click **New**.
3. Complete the following fields:
### REST Message form

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a descriptive name for this message.</td>
</tr>
<tr>
<td>Endpoint</td>
<td>Enter the endpoint that this REST message is sent to. The endpoint value may include variables using the format <code>${variable}</code>.</td>
</tr>
<tr>
<td>Authentication type</td>
<td>Select the type of authentication to use, if any, and the profile record that contains the user credentials.</td>
</tr>
<tr>
<td></td>
<td>Outbound REST supports basic authentication and OAuth 2.0. Outbound REST supports mutual authentication with basic authentication only.</td>
</tr>
<tr>
<td></td>
<td>Authentication configured here is inherited by the associated HTTP methods. You can configure authentication for each method which overrides any authentication setting at the message level.</td>
</tr>
<tr>
<td>HTTP Headers</td>
<td>Double-click a row in the HTTP Headers embedded list to define the header Name and Value. The web service provider determines which headers are supported or required.</td>
</tr>
</tbody>
</table>

4. Click **Submit**.

After creating the REST message, a GET HTTP method is created automatically using the values from the REST message record.

### What to do next
Create or edit HTTP methods and run a request.

### Define a REST message HTTP method

Define an HTTP method such as GET or POST to send a request to a web service provider.

**Before you begin**
Role required: web_service_admin

**About this task**
When you create a REST message record, several default HTTP methods are automatically created using settings inherited from the REST message record, such as the **Endpoint**. Subsequent changes to the REST message record are not
applied to the HTTP methods automatically. You can create additional HTTP methods or modify the default HTTP methods to implement new behavior.

**Procedure**

1. Navigate to **System Web Services > Outbound > REST Message**.
2. Select a REST message you want to define an HTTP method for.
3. In the **HTTP Methods** related list, click **New**.
4. Select the **HTTP method** you want to use, such as GET or POST.
5. Enter the **Endpoint** this HTTP method should access.
   The endpoint value may include variables using the format `${variable}`.
6. Right-click the form header and select **Save**.

**What to do next**

After creating the HTTP method, you can override the security settings from the parent REST message, configure HTTP headers, add variables, or test the method. For PUT, POST, and PATCH methods you can define a message body.

**Testing REST message HTTP methods**

After configuring an HTTP method for an outbound REST message, you can test it to ensure that the request is valid and the response returns as expected.

To test an HTTP method, click the **Test** related link on the HTTP Method form.

Each test run displays the response status, such as 200 for a successful GET request, the full endpoint URL, any parameters passed in the request, and the response body.

**Note:** Fields on the Test Runs form are for information only; changes to these fields do not apply to the REST message or HTTP method. Do not modify these values when testing different REST message configurations. Instead, update the REST message or HTTP method, then run a new test.

If the HTTP method includes variables, the **Test value** for each variable in the **Variable Substitutions** related list is used when testing the method.

Completed test runs for an HTTP method appear in the **Test Runs** related list. If there was an error during the request, the **Error Code** and **Error Message** fields appear.
Define a REST message HTTP header

Define an HTTP header for a REST message or HTTP method to send that header with REST requests.

Before you begin
Role required: web_service_admin

About this task
You can specify an HTTP header for a REST message, or for an HTTP method. Headers defined for a REST message apply to all HTTP methods for that REST message. If you specify the same header for both a REST message and a child HTTP method, the value defined for the HTTP method overrides the value from the parent REST message.

Procedure
1. Navigate to System Web Services > REST Message.
2. Select a REST message.
3. Optional: To specify a header for an HTTP method instead of the REST message, in the HTTP Method related list, select an HTTP method.
4. Select the HTTP Request tab.
5. In the HTTP Headers embedded list, click Insert a new row.
6. Enter the name of the header, such as Content-Type or Accept. Supported headers depend on the REST web service provider you want to connect to. Refer to the documentation for your web service provider to identify which headers are valid or required.
7. Click on the Value field for the new row and enter the value you want to assign this header.
   You can use a variable in the format `${variable}` instead of a static value. You can assign a value to the variable when sending a REST request.
8. Click Update.

Sending outbound REST messages through a MID Server

You can configure a REST message HTTP method to be sent through a MID Server.

By using a MID Server, the request can reach an endpoint that is behind a firewall or within a private network.
To configure an HTTP method to use a MID Server, select a MID Server in the **Use MID Server** field on the HTTP Method form. The instance must have an active MID Server to use this functionality.

**Using special characters in URIs**

A REST function URI or function variable may use special characters, such as pipe (|) characters.

When using these characters in a REST message, use URL encoding to escape these characters. For example, to use a parameter value of `user|title`, enter `user%7Ctitle`. Entering special characters directly may cause the REST message to fail and display the response *Invalid uri <URI>: Invalid query*.

**Outbound REST authentication**

Outbound REST messages support multiple types of authentication. Different web service providers may require a specific type of authentication. Outbound REST supports the following authentication formats.

- Basic authentication using a username and password
- OAuth 2.0 using an OAuth provider and profile
- Mutual authentication using protocol profiles

**Overriding REST authentication**

You can define authentication for a REST message, or individually for each HTTP method. HTTP methods inherit authentication from their parent REST message record when the HTTP method **Authentication type** is **Inherit from parent**, which is the default value.

You can disable authentication for a specific HTTP method by setting the **Authentication type** field to **No authentication**, or specify authentication that is different from the parent REST message by selecting basic auth or OAuth.

**Authentication requirements**

Outbound REST supports mutual authentication only when using basic authentication. Mutual authentication is not available with OAuth 2.0.

OAuth 2.0 can be used only with messages that are not configured to use a MID Server. You cannot send OAuth 2.0 authenticated messages through a MID Server.

When scripting new REST messages configured with authentication you must use the RESTMessageV2 API. The legacy RESTMessage APIs do not support current authentication formats.
Configure a REST message with basic auth

You can configure an outbound REST message to provide basic authentication credentials with each request.

**Before you begin**
Role required: web_service_admin
Before starting this procedure, ensure there is a REST Message record that you want to configure to use basic auth.

ℹ️ **Note:** Ensure any scripts that call this REST message use the RESTMessageV2 API. The RESTMessageV2 API is required to send authenticated REST messages via scripts.

**Procedure**

1. Navigate to **System Web Services > Outbound > REST Message**.
2. Select a REST message record.
3. In the **Authentication type** field, select **Basic**.

ℹ️ **Note:** The Basic (Simple) choice appears on REST message records configured to use basic authentication prior to the Geneva release. This choice is intended for compatibility with older REST messages and should not be used for REST messages created in the Geneva release or later.

4. In the **Basic auth profile** field, select the basic authentication profile that contains the credentials you want to send.
5. Click **Submit**.

**What to do next**
Test the REST message to ensure you receive the expected response. You can optionally specify different authentication settings for each HTTP method related to this REST message, overriding the parent REST message settings.

**Create a basic auth profile**
Create a basic auth profile to specify basic authentication credentials for one or more REST messages.

**Before you begin**
Role required: web_service_admin
Procedure

1. Navigate to **System Web Services > REST Message**.
2. Select a REST message record.
3. In the **Authentication type** field, select **Basic**.
4. In the **Basic auth profile** field, click the reference lookup icon.
5. Click **New**.
6. Enter a descriptive **Name** for the profile.
7. Enter the **Username** and **Password** you want to send as basic authentication credentials.
8. Click **Submit**.

**What to do next**
Configure a REST message to use this basic auth profile.

**Configure a REST message with OAuth**

You can configure an outbound REST message to send OAuth credentials with the request.

**Before you begin**
Role required: web_service_admin and oauth_admin
Before starting this procedure, ensure:

- There is a REST Message record that you want to configure to use OAuth.
- There is an OAuth provider set up in the OAuth application registry with the OAuth client information to use.
- The OAuth provider has an associated OAuth 2.0 profile.
- The REST message HTTP Methods are not configured to use a MID Server.

⚠ **Note:** Ensure any scripts that call this REST message use the RESTMessageV2 API. The RESTMessageV2 API is required to send authenticated REST messages via scripts.

Procedure

1. Navigate to **System Web Services > Outbound > REST Message**.
2. Select a REST message record.
3. In the **Authentication type** field, select **OAuth 2.0**.
4. In the **OAuth profile** field, select the OAuth 2.0 profile that specifies the credentials you want to send.

5. Right-click the form header and select **Save**.
   An info message appears at the top of the form indicating that you must request a new OAuth token.

6. Click the **Get OAuth Token** related link.
   Depending on your OAuth provider, a separate window from your provider may appear asking for confirmation before providing a token. Complete any steps required by the provider to obtain the token.

**What to do next**
Test the REST message to ensure you receive the expected response. You can optionally specify different authentication settings for each HTTP method related to this REST message, overriding the parent REST message settings.

**Use a third-party OAuth provider**
Each client application must register with the instance to participate in OAuth authorization.

**Before you begin**
Role required: admin

**About this task**

*Note:* ServiceNow only supports third-party OAuth providers to authorize requests from ServiceNow to third-party APIs.

For example, you might want to integrate with a third-party Calendar service which requires OAuth 2.0 access tokens to read a user's scheduled events and create events. Configure the Calendar service as a third-party OAuth provider. This configuration allows you to get an access token from the Calendar service and then use the token to make requests against the service from ServiceNow.

**Procedure**

1. Navigate to **System OAuth > Application Registry** and then click **New**.
2. On the interceptor page, click **Connect to a third party OAuth Provider** and then fill in the form.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A unique name that identifies the application to access.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Client ID</td>
<td>The unique ID of the application. The instance uses the client ID when requesting an access token. You must get the client ID from the authorization server.</td>
</tr>
<tr>
<td>Client Secret</td>
<td>[Required] The shared secret string that the instance and the application use to authorize communications with one another. If you do not enter the correct client secret, tokens are not issued.</td>
</tr>
<tr>
<td>OAuth API Script</td>
<td>This option enables you to reference an amended OAuthUtil script include. Copy and rename the default OAuthUtil script include file, and then amend this version for specific requests and responses to match your 3rd party OAuth provider. The amended script name must have the prefix OAuth. See OAuthUtil to add the required body parameter in the proper method.</td>
</tr>
<tr>
<td>Logo URL</td>
<td>The URL that contains an image to use as the application logo.</td>
</tr>
<tr>
<td>Default Grant Type</td>
<td>The type of grant:</td>
</tr>
<tr>
<td></td>
<td>• Authorization code: The code that is granted to the client to obtain an access token, which is then used to obtain access to the resource. If you select this option, then you need an authorization URL (the URL of the authorization server).</td>
</tr>
<tr>
<td></td>
<td>• Resource owner password credentials: The user name and password of the user that is trying to obtain access to the resource.</td>
</tr>
<tr>
<td></td>
<td>• Client Credentials: The client ID and client secret, which are both used to get the access token. This method does not provide refresh tokens.</td>
</tr>
<tr>
<td></td>
<td>• JWT Bearer: An authorization server validates a JWT token which enables identity and security information to be shared across security domains.</td>
</tr>
<tr>
<td>Refresh Token Lifespan</td>
<td>The refresh token lifespan in seconds.</td>
</tr>
<tr>
<td>Accessible from</td>
<td>The application scope that this registry is accessible from.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Active</td>
<td>A check box that indicates that the application registry is active.</td>
</tr>
<tr>
<td>Authorization URL</td>
<td>If you are using the authorization code grant type, the URL of the endpoint to authorize the user. If you are accessing another ServiceNow instance, append /oauth_auth.do to the URL.</td>
</tr>
<tr>
<td>Token URL</td>
<td>The location of the token endpoint that the instance uses to retrieve and refresh tokens. If you are accessing another ServiceNow instance, append /oauth_token.do to the URL.</td>
</tr>
<tr>
<td>Redirect URL</td>
<td>The application endpoint that receives the authorization code. Leave the field empty to have the instance auto-generate the URL. If you are accessing another ServiceNow instance, append /oauth_redirect.do to the URL.</td>
</tr>
<tr>
<td>Token Revocation URL</td>
<td>The location of the endpoint that the instance uses to revoke the token. If you are accessing another instance, append /oauth_revoke.do to the URL.</td>
</tr>
<tr>
<td>Comments</td>
<td>Additional information to associate with the application.</td>
</tr>
<tr>
<td>Embedded lists</td>
<td></td>
</tr>
<tr>
<td>OAuth Entity Profiles</td>
<td>The profiles that are associated with the OAuth provider. The profile includes the grant type. Click the profile name to go to the OAuth Entity Profile form.</td>
</tr>
<tr>
<td>OAuth Entity Scopes</td>
<td>The entity scopes associated with the OAuth provider. The scope identifies the services the application has access to. Click the scope name to go to the OAuth Entity Scope form.</td>
</tr>
</tbody>
</table>

3. Click **Submit**. The record is saved in the Application Registries [oauth_entity] table.

   The system creates a record in the Application Registries [oauth_entity] table of type **OAuth Provider**. The instance also auto-generates a default profile using the specified grant type, but without any scopes.

**What to do next**

You can create additional profiles, each with scopes.

**OAuth profiles and scopes**

In the OAuth provider scenario, profiles and scopes specify the grant type, authorization type, and level of access.
In the OAuth provider scenario, the OAuth profile refers to a combination of a grant type and at least one scope. The scope specifies the access that the user has to the protected resource, such as read or write. You can create a profile for each third-party provider and obtain the specific set of scopes from the provider. See Specify an OAuth profile and Specify an OAuth scope for more information. The instance also uses OAuth profiles when a REST call specifies OAuth 2.0 authentication. The instance auto-creates a default profile for each third-party provider record that you create. There can be only one default profile.

Specify the following parameters, which are saved in the OAuth Requestor Profile [OAuth_requestor_profile] table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oauth_requestor</td>
<td>The sys_id of the object, which can be a user record or an email account.</td>
</tr>
<tr>
<td>oauth_requestor_context</td>
<td>Descriptor that provides context for the OAuth requestor. As a good practice, use the name of the table where the oauth_requestor object is saved.</td>
</tr>
<tr>
<td>oauth_provider_profile</td>
<td>The sys_id of the OAuth profile record that is the default (see Specify an OAuth profile).</td>
</tr>
</tbody>
</table>

When the user attempts to authenticate, the provider accesses the OAuth Requestor Profile table to look for the user. If the user is found, the authentication is successful. If not, the provider accesses the default profile to determine the grant type and how to proceed with the authentication.

Specify an OAuth profile

An OAuth profile includes the grant type that the third-party OAuth provider needs to obtain access to the restricted resource.

Before you begin
Role required: admin
About this task

Procedure
1. Open a third-party OAuth provider record.
2. In the OAuth Entity Profiles embedded list, click **Insert a new row** and then enter a name for the profile.
3. Right-click the Application Registry form header and select **Save**. The system creates the profile record.
4. Click the name of the profile you created and then fill in the form fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a descriptive name.</td>
</tr>
<tr>
<td>OAuth provider</td>
<td>Verify the provider that is associated with the profile.</td>
</tr>
<tr>
<td>Grant type</td>
<td>Select the grant type:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Authorization code</strong>: The code that is granted to the client to obtain an access token, which is then used to obtain access to the resource. If you select this option, then you need an authorization URL (the URL of the authorization server).</td>
</tr>
<tr>
<td></td>
<td>• <strong>Resource owner password credentials</strong>: The user name and password of the user that is trying to obtain access to the resource.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>• Client Credentials</td>
<td>The client ID and client secret, which are both used to get the access token. This method does not provide refresh tokens.</td>
</tr>
<tr>
<td>• JWT Bearer</td>
<td>An authorization server validates a JWT token which enables identity and security information to be shared across security domains.</td>
</tr>
<tr>
<td>Is default</td>
<td>Select this option to make the profile the default option for the associated provider.</td>
</tr>
<tr>
<td>5. Click Update.</td>
<td></td>
</tr>
</tbody>
</table>

Specify an OAuth scope

Specify the OAuth scopes that you get from the provider. Scopes can be any level of access specified by the provider, such as read, write, or any string, including a URL.

Before you begin
Role required: admin

Procedure
1. Open a third-party OAuth provider record.
2. Open a profile associated with the provider.
3. In the OAuth Entity Profile Scopes embedded list, click Insert a new row and then enter a Name for the profile.
4. Right-click the OAuth Entity Profile form header and select Save. The profile record is created.
5. Click the name of the scope that you created and then fill in the form fields.
OAuth Entity Scope form fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a descriptive name.</td>
</tr>
<tr>
<td>OAuth provider</td>
<td>Verify the provider associated with this scope.</td>
</tr>
<tr>
<td>OAuth scope</td>
<td>The scope that you are granted by the provider. Typical scopes are read and write. Scopes can be any string that the provider specifies.</td>
</tr>
</tbody>
</table>

6. Click **Update**.

Outbound REST with OAuth 2.0 profile tutorial - integrating with Google Contacts API

This tutorial explains how to use an OAuth 2.0 profile to authenticate an outbound REST message with Google to retrieve contact information.

The procedures detailed in this tutorial require the oauth_admin and web_service_admin roles. Ensure you have both of these roles before starting this tutorial.

OAuth 2.0 tutorial - configure the Google service as an OAuth provider

Use the Google Developer Console to set up an OAuth 2.0 provider.

**Before you begin**

Role required: None

This procedure is performed within the Google Developer Console. You must have a Google account to access this console.

**About this task**

Configure the Google service in order to obtain a client ID and client secret, and specify your ServiceNow instance URL as the OAuth redirect URL.
Note: This information describes the state of the Google Developer Console and Contacts API as of July 22, 2015. Changes made after that date may not be included in this document.

Procedure

2. Log in using your Google credentials.
3. Click Select a project.
4. Click Create a project.
5. Enter a Project name.
6. Click Create.  
   After Google creates the project, the project dashboard appears.
7. Navigate to APIs & auth > APIs.
8. Select the Contacts API.
9. Click Enable API.
10. Navigate to APIs & auth > Credentials.
11. Click Create new Client ID.
12. Ensure the Web application radio button is selected and click Configure consent screen.
13. Enter a descriptive Product name.  
   This name appears when you authorize the OAuth token in your instance.
14. Click Save.
15. In the Create Client ID window, add the OAuth redirect URI for your instance to the Authorized redirect URIs field.  
   This URI follows the format https://<instance>.service-now.com/oauth_redirect.do
16. Click Create Client ID.  
   The client ID information appears.
17. Record the Client ID and Client secret values.  
   You will need these values to configure the Google service as an OAuth provider in your instance.

OAuth 2.0 tutorial - create an OAuth provider and profile

Set up the Google service as an OAuth provider in ServiceNow by entering your client information, Google API URLs, and configuring the OAuth profile.
Before you begin
Role required: oauth_admin
You must have configured the Google service as an OAuth provider and recorded your Client ID and Client Secret values.

Procedure
1. Navigate to System OAuth > Application Registry.
2. Click New.
3. Select Connect to a third party OAuth Provider.
4. Enter a Name for the OAuth provider. For this example, use Google.
5. Enter the Client ID and Client Secret that you obtained from Google.
7. In the Authorization URL field, enter https://accounts.google.com/o/oauth2/
   auth.
9. In the Redirect URL field, enter https://<instance>.service-now.com/
   oauth_redirect.do.
   This URL must match the redirect URL provided to Google.
10. In the Token Revocation URL field, enter https://accounts.google.com/o/oauth2/
    revoke.
11. Right-click the form header and select Save.
    A new OAuth Entity Profile record is created.
12. In the OAuth Entity Scopes embedded list, add a new row with the Name and
13. Right-click the form header and select Save.
14. In the OAuth Entity Profiles embedded list, select the automatically-created
    profile.
15. In the OAuth Entity Profile Scopes embedded list, add a new row and select
    the Google contacts API read-only scope.
16. Click Update.

OAuth 2.0 tutorial - create a REST message
Create a REST message and associated HTTP method to contact the Google service using the OAuth 2.0 profile.
Before you begin
Role required: web_service_admin and oauth_admin
You must have configured an OAuth provider and profile using the Google API information and your OAuth credentials.

Procedure
1. Navigate to System Web Services > REST Message.
2. Click New.
3. Enter a descriptive Name.
4. In the Endpoint field, enter https://www.google.com/m8/feeds/contacts/default/full.
   By using default instead of a specific username, the Google API uses the OAuth credentials to determine which account to get information from.
5. In the Authentication tab, set the Authentication type to OAuth 2.0.
6. In the OAuth profile field, select the Google contacts OAuth profile.
7. Right-click the form header and select Save.
8. Click the Get OAuth Token related link to request an authorization token from Google using the configured client ID and secret.
9. In the Request for Permission window that appears, click Accept to grant access to your Google contacts.
   The token acquired is not directly accessible in your instance.
10. In the HTTP Methods related list, select the GET method.
11. Leave the HTTP method Authentication type as -- None -- to use the OAuth profile from the parent REST message record.
12. On the HTTP Request tab, add a new row to the HTTP Headers related list with a Name of GData - Version and a Value of 3.0.
13. Right-click the form header and select Save.
14. Click the Test related link.
   The test result should display an HTTP Status of 200, and the result of the contacts API call.

Outbound REST mutual authentication
Mutual authentication causes the web service provider and consumer to authenticate with each other before communicating.
ServiceNow supports mutual authentication for outbound web services. Mutual authentication is not available for inbound web services or for outbound web services that use a MID Server.

**Related information**
- Outbound web services mutual authentication

### Variable substitution in outbound REST messages

You can use variables when creating outbound REST messages and assign values to those variables when performing a request.

Variables are allowed in the **Endpoint** URL, HTTP Header and HTTP Query Parameter **Value** fields, and the **Content** field for POST and PUT methods.

The syntax for variables is `${variable_name}`. The REST message substitutes this variable with the parameter values provided when the method runs. For example, if the REST message **Endpoint** is `http://myserver.mycompany.com/offices/${id}`, a parameter named `id` must exist and contain a value that can be used when the method runs.

You can assign a value to variables when running the request using the RESTMessageV2 API `setStringParameter` and `setStringParameterNoEscape` methods.

When testing an HTTP method that includes variables, the **Test value** for each variable in the **Variable Substitutions** related list is used.

### Generate REST message variables

You can automatically populate the list of variable substitutions, based on variables defined in several REST message HTTP method fields.

**Before you begin**

Role required: web_service_admin

Before starting this procedure, create a REST Message record with at least one HTTP method that uses variables.

**Procedure**

1. Navigate to **System Web Services** > **REST Message**.
2. Select a REST Message record.
3. Select a method from the **HTTP Methods** related list.
4. Click the **Auto-generate variables** related link.
The **Variable Substitutions** related list is automatically populated for any variables defined in the HTTP Method **Endpoint** field and the **HTTP Headers** and **HTTP Query Parameters** embedded lists. For POST and PUT messages, variables defined in the **Content** field are also used.

**What to do next**
You can use the REST Message workflow activity to send the message, or click **Preview Script Usage** to get a sample script. The sample script includes a `setStringParameter` call for each defined variable substitution that allows you to assign a value to the variable in your script.

**Scripting outbound REST**
You can send outbound REST requests from any place in the Now Platform where scripting is allowed.

For example, you can return data from a REST endpoint using a business rule when an event is triggered. Create a script from scratch or let the REST message preview feature create the script based on content and parameters you provide in the method record.

For detailed API information about the server-side RESTMessageV2 and RESTResponseV2 APIs, see the API documentation on developer.servicenow.com.

**Generate a REST message script preview**
You can generate an example script to send a REST message based on content and parameters you provide in the method record.

**Before you begin**
Role required: web_service_admin or admin

**About this task**
Generate an example script and use it as a starting point when scripting outbound REST messages.

**Procedure**
1. Navigate to **System Web Services > REST Message**.
2. Select a REST message record.
3. In the **HTTP Methods** related list, select an HTTP method record.
4. Ensure the HTTP method is configured as needed, including any variables.
5. Save the record.
6. In the Variable Substitutions related list, assign a value to each variable.

7. Under Related Links, click Preview script usage.
   The instance displays the script that the REST message generated for this method.

8. Copy this script and modify it as needed to use elsewhere in the instance.

What to do next
Refer to the RESTMessageV2 and RESTResponseV2 APIs for more information on available scripting methods. Outbound REST scripting examples are also available.

Direct RESTMessageV2 example
You can send an outbound REST message directly to the endpoint.

In this example, the script sends a REST message requesting a stock quote and waits for a response. If there is no response from the web service provider, or if the specified REST message record is unavailable, the script throws an error, handled in this example by the try-catch block.

```javascript
var requestBody;
var responseBody;
var status;
var sm;
try{
    var r = new sn_ws.RESTMessageV2('Yahoo Finance', 'get');
    //override authentication profile
    //authentication type = 'basic' / 'oauth2'
    //r.setAuthentication(authentication type, profile name);
    var response = r.execute();
    var responseBody = response.getBody();
    var httpStatus = response.getStatusCode();
}
catch(ex) {
    var message = ex.getMessage();
}
```
RESTMessageV2 MID server example

You can send an outbound REST message through a MID Server.

By sending the message through a MID Server, you can access endpoints that are behind a firewall or within a private network. All REST messages sent through a MID Server are asynchronous.

```
var requestBody;
var responseBody;
var status;
var sm;
try{
    sm = new sn_ws.RESTMessageV2("Yahoo Finance", "get"); // Might throw exception if message
doesn't exist or not visible due to scope.
    sm.setBasicAuth("admin","admin");
    sm.setStringParameter("symbol","NOW");
    sm.setStringParameterNoEscape("xml_data","<data>test</data>");
    sm.setHttpTimeout(10000); //In milliseconds. Wait at most 10 seconds for response from
http request.
    response = sm.execute(); //Might throw exception if http connection timed out or some issue
with sending request itself because of encryption/decryption of password.
    responseBody = response.haveError() ? response.getError Message() : response.getBody();
    status = response.getStatusCode();
} catch(ex) {
    responseBody = ex.getMessage();
    status = '500';
} finally {
    requestBody = sm ? sm.getRequestBody():null;
}
    gs.info("Request Body: " + requestBody);
    gs.info("Response: " + responseBody);
    gs.info("HTTP Status: " + status);
```
response.waitForResponse(60); // In seconds. Wait at most 60 seconds to get response from ECC Queue/Mid Server // Might throw exception timing out waiting for response in ECC queue.

responseBody = response.haveError() ? response.getErrorMessage() : response.getBody();
status = response.getStatusCode();
} catch(ex) {
  responseBody = ex.getMessage();
  status = '500';
} finally {
  requestBody = sm ? sm.getRequestBody():null;
}
gs.info("Request Body: " + requestBody);
gs.info("Response: " + responseBody);
gs.info("HTTP Status: " + status);

Note: This example uses waitForResponse to pause for a response, and then details how the response is handled. However, when using executeAsync, consider processing the response body in a separate business rule to take advantage of the asynchronous call rather than using waitForResponse.

Recordless RESTMessageV2 example

You can use the RESTMessageV2() constructor with no parameters to define a REST message entirely in the script.

When using this constructor you must provide an endpoint and HTTP method. In this example, the script creates an empty REST message and sets the values needed to insert an incident record.

```javascript
var restMessage = new sn_ws.RESTMessageV2();
restMessage.setBasicAuth("admin", "admin");
restMessage.setHttpMethod("post");
restMessage.setEndpoint("http://<instance>.service-now.com/api/now/table/incident");
restMessage.setRequestBody("{"short_description" : "Test incident"}");
var response = restMessage.execute();
```

Outbound SOAP web service

The SOAP Message module can be used to develop, prototype, and save outbound SOAP messages that can be reused in business rules and scripts.

You can use outbound SOAP messages in scripts using the SOAPMessageV2 API and the SOAPResponseV2 API. Examples detailing how to script outbound SOAP are available.
Outbound SOAP video tutorial
The following video tutorial demonstrates how to configure outbound SOAP web service messages to consume third-party web services from an instance.

Outbound SOAP video tutorial

SOAP message
Information needed to send SOAP requests is stored in SOAP message records. Each record specifies an endpoint for the request, the required format of the request as a web services description language (WSDL) file, authentication information, and a list of functions that can run against the endpoint.

Create a SOAP message
Create a SOAP message to define the remote endpoint, WSDL, and authentication settings.

Before you begin
Role required: web_service_admin

Procedure
1. Navigate to System Web Services > SOAP Message.
2. Click New.
3. Enter a Name to identify the SOAP message.
4. Specify a WSDL using one of these options:
   - To download and use an online WSDL source, select the Download WSDL check box and enter the URL for the WSDL in the WSDL field.
   - To enter the WSDL directly, clear the Download WSDL check box, and then copy and paste the WSDL XML into the WSDL XML field.
5. If the endpoint is protected by basic authentication, select the Use basic auth check box and enter the credentials.
6. If the endpoint requires mutual authentication, select the Enable mutual authentication check box and select a Protocol profile to use for mutual authentication.
7. Click Submit.
This image shows an example of a SOAP message that connects to a demo instance of ServiceNow.

**SOAP message functions**

After you create a SOAP message record, you can click **Generate sample SOAP messages** to populate the SOAP Message Functions related list.

The instance creates these functions by reading the supplied WSDL definition.
The SOAP action, SOAP endpoint, and Envelope fields should be populated automatically based on the WSDL definition. The Envelope defines the message to send to the endpoint. In this example, the Envelope values have this format:

```xml
<!-- optional --><short_description xsi:type="xsd:string">String</short_description>
```

To submit a specific value, enter the value directly in the appropriate XML tag. In this example, to set the Short description for a record, enter:

```xml
<short_description xsi:type="xsd:string">This is the short description</short_description>
```
Variable substitution in outbound SOAP

To use variable substitution, use the format `${<variable_name>}` instead of defining a specific value.

```xml
<short_description xsi:type="xsd:string">${short_desc}</short_description>
```

To test variable substitution after you have modified the SOAP envelope with the variables, define values for the variables in the SOAP Message Parameters related list. For example, click New and enter the following information:

**Soap message parameters**

| Name  | Value          | SOAP Function | soap message parameters
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>short_desc</td>
<td>this is the short description</td>
<td>insert</td>
<td><img src="image" alt="Soap message parameters" /></td>
</tr>
</tbody>
</table>

**Test the SOAP message**

Test a SOAP message to validate the configuration before using the message in an integration.

To test the SOAP message, click the Test related link. You are redirected to a test result form as shown below.

**Soap message test**

![Soap message test](image)
You can see the original SOAP request message, the resulting HTTP status code, and the SOAP response in this screen. You can also click the Rerun test related link to resubmit the SOAP request.

ℹ️ Note: A test SOAP message will time out after 60 seconds if a response is not received.

**Send a SOAP message through a MID server**

When creating SOAP message functions, you can configure the function to be sent through a MID Server.

There must be a running MID Server associated with your instance to use this functionality. All SOAP messages sent through a MID Server are performed asynchronously.

By specifying a MID Server, all SOAP requests that use this SOAP message are sent through that MID Server. You can override the selected MID Server by using the `setMIDServer(mid server)` API call in a script.

**Create a SOAP message from a WSDL that references an external XSD file**

Create a SOAP message from a WSDL and external XSD file.

**Before you begin**

Role required: `web_service_admin`
About this task
This task includes example WSDL and XSD files for a weather forecast SOAP message. Your WSDL and XSD file will vary.

Procedure
1. Navigate to System Web Services > SOAP Message and create a new record.
2. Clear the Download WSDL check box.
3. Paste the content of the WSDL into the WSDL XML field.
4. Save the record.
5. In the SOAP Message Imports related list, click New.
6. Paste the content of the XSD file into the External Document field.
7. Set the Schema Location field to db://<name of the referenced XSD file>.xsd. Specifying the schema location allows the instance to know the location of the referenced XSD file.
8. Click Submit.
9. Click Generate sample SOAP messages.

Example: Example WSDL and XSD files

```xml
<?xml version="1.0" encoding="utf-8"?>
 xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:s="http://www.w3.org/2001/XMLSchema"
 xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
 xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
targetNamespace="http://www.webservicex.net"
 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">  
 <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Get one week weather forecast for valid zip code or Place name in USA</wsdl:documentation>
 <wsdl:types>
 <s:schema elementFormDefault="qualified" targetNamespace="http://www.webservicex.net">
  <s:include schemaLocation="WeatherForecast.xsd" />
 </s:schema>
 </wsdl:types>
 <wsdl:message name="GetWeatherByZipCodeSoapIn">
  <wsdl:part name="parameters" element="tns:GetWeatherByZipCode" />
 </wsdl:message>
 <wsdl:message name="GetWeatherByZipCodeSoapOut">
```

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<wsdl:part name="parameters" element="tns:GetWeatherByZipCodeResponse" />
</wsdl:message>
<wsdl:message name="GetWeatherByPlaceNameSoapIn">
  <wsdl:part name="parameters" element="tns:GetWeatherByPlaceName" />
</wsdl:message>
<wsdl:message name="GetWeatherByPlaceNameSoapOut">
  <wsdl:part name="parameters" element="tns:GetWeatherByPlaceNameResponse" />
</wsdl:message>
<wsdl:message name="GetWeatherByZipCodeHttpGetIn">
  <wsdl:part name="ZipCode" type="s:string" />
</wsdl:message>
<wsdl:message name="GetWeatherByZipCodeHttpGetOut">
  <wsdl:part name="Body" element="tns:WeatherForecasts" />
</wsdl:message>
<wsdl:message name="GetWeatherByPlaceNameHttpGetIn">
  <wsdl:part name="PlaceName" type="s:string" />
</wsdl:message>
<wsdl:message name="GetWeatherByPlaceNameHttpGetOut">
  <wsdl:part name="Body" element="tns:WeatherForecasts" />
</wsdl:message>
<wsdl:message name="GetWeatherByZipCodeHttpPostIn">
  <wsdl:part name="ZipCode" type="s:string" />
</wsdl:message>
<wsdl:message name="GetWeatherByZipCodeHttpPostOut">
  <wsdl:part name="Body" element="tns:WeatherForecasts" />
</wsdl:message>
<wsdl:message name="GetWeatherByPlaceNameHttpPostIn">
  <wsdl:part name="PlaceName" type="s:string" />
</wsdl:message>
<wsdl:message name="GetWeatherByPlaceNameHttpPostOut">
  <wsdl:part name="Body" element="tns:WeatherForecasts" />
</wsdl:message>
<wsdl:portType name="WeatherForecastSoap">
  <wsdl:operation name="GetWeatherByZipCode">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsd1/">Get one week weather forecast for a valid Zip Code (USA)</wsdl:documentation>
    <wsdl:input message="tns:GetWeatherByZipCodeSoapIn" />
    <wsdl:output message="tns:GetWeatherByZipCodeSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="GetWeatherByPlaceName">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsd1/">Get one week weather forecast for a place name (USA)</wsdl:documentation>
    <wsdl:input message="tns:GetWeatherByPlaceNameSoapIn" />
    <wsdl:output message="tns:GetWeatherByPlaceNameSoapOut" />
  </wsdl:operation>
</wsdl:portType>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
<wsdl:portType name="WeatherForecastHttpGet">
  <wsdl:operation name="GetWeatherByZipCode">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Get one week weather forecast for a valid Zip Code(USA)</wsdl:documentation>
    <wsdl:input message="tns:GetWeatherByZipCodeHttpGetIn" />
    <wsdl:output message="tns:GetWeatherByZipCodeHttpGetOut" />
  </wsdl:operation>
  <wsdl:operation name="GetWeatherByPlaceName">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Get one week weather forecast for a place name(USA)</wsdl:documentation>
    <wsdl:input message="tns:GetWeatherByPlaceNameHttpGetIn" />
    <wsdl:output message="tns:GetWeatherByPlaceNameHttpGetOut" />
  </wsdl:operation>
</wsdl:portType>

<wsdl:portType name="WeatherForecastHttpPost">
  <wsdl:operation name="GetWeatherByZipCode">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Get one week weather forecast for a valid Zip Code(USA)</wsdl:documentation>
    <wsdl:input message="tns:GetWeatherByZipCodeHttpPostIn" />
    <wsdl:output message="tns:GetWeatherByZipCodeHttpPostOut" />
  </wsdl:operation>
  <wsdl:operation name="GetWeatherByPlaceName">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Get one week weather forecast for a place name(USA)</wsdl:documentation>
    <wsdl:input message="tns:GetWeatherByPlaceNameHttpPostIn" />
    <wsdl:output message="tns:GetWeatherByPlaceNameHttpPostOut" />
  </wsdl:operation>
</wsdl:portType>

<wsdl:binding name="WeatherForecastSoap" type="tns:WeatherForecastSoap">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
  <wsdl:operation name="GetWeatherByZipCode">
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="GetWeatherByPlaceName">
    <soap:operation soapAction="http://www.webservicex.net/GetWeatherByPlaceName" style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
<soap:operation soapAction="http://www.webservicex.net/GetWeatherByPlaceName"
    style="document" />
<wsdl:input>
    <soap:body use="literal"/>
</wsdl:input>
<wsdl:output>
    <soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:binding name="WeatherForecastSoap12" type="tns:WeatherForecastSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="GetWeatherByZipCode">
        <soap12:operation soapAction="http://www.webservicex.net/GetWeatherByZipCode"
            style="document"/>
        <wsdl:input>
            <soap12:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap12:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetWeatherByPlaceName">
        <soap12:operation soapAction="http://www.webservicex.net/GetWeatherByPlaceName"
            style="document"/>
        <wsdl:input>
            <soap12:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap12:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
<wsdl:binding name="WeatherForecastHttpGet" type="tns:WeatherForecastHttpGet">
    <http:binding verb="GET"/>
    <wsdl:operation name="GetWeatherByZipCode">
        <http:operation location="/GetWeatherByZipCode"/>
        <wsdl:input>
            <http:urlEncoded/>
        </wsdl:input>
        <wsdl:output>
            <mime:mimeXml part="Body"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
<wsdl:operation name="GetWeatherByPlaceName">
  <http:operation location="/GetWeatherByPlaceName" />
  <wsdl:input>
    <http:urlEncoded />
  </wsdl:input>
  <wsdl:output>
    <mime:mimeXml part="Body" />
  </wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:binding name="WeatherForecastHttpPost" type="tns:WeatherForecastHttpPost">
  <http:binding verb="POST" />
  <wsdl:operation name="GetWeatherByZipCode">
    <http:operation location="/GetWeatherByZipCode" />
    <wsdl:input>
      <mime:content type="application/x-www-form-urlencoded" />
    </wsdl:input>
    <wsdl:output>
      <mime:mimeXml part="Body" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="GetWeatherByPlaceName">
    <http:operation location="/GetWeatherByPlaceName" />
    <wsdl:input>
      <mime:content type="application/x-www-form-urlencoded" />
    </wsdl:input>
    <wsdl:output>
      <mime:mimeXml part="Body" />
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
<wsdl:service name="WeatherForecast">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Get one week weather forecast for valid zip code or Place name in USA</wsdl:documentation>
  <wsdl:port name="WeatherForecastSoap" binding="tns:WeatherForecastSoap">
    <soap:address location="http://www.webservicex.net/WeatherForecast.asmx" />
  </wsdl:port>
  <wsdl:port name="WeatherForecastSoap12" binding="tns:WeatherForecastSoap12">
    <soap12:address location="http://www.webservicex.net/WeatherForecast.asmx" />
  </wsdl:port>
  <wsdl:port name="WeatherForecastHttpGet" binding="tns:WeatherForecastHttpGet">
    <http:address location="http://www.webservicex.net/WeatherForecast.asmx" />
  </wsdl:port>
  <wsdl:port name="WeatherForecastHttpPost" binding="tns:WeatherForecastHttpPost">
    <http:address location="http://www.webservicex.net/WeatherForecast.asmx" />
  </wsdl:port>
</wsdl:service>
<http:address location="http://www.webservicex.net/WeatherForecast.asmx" />
</wsdl:port>
</wsdl:service>
</wsdl:definitions>

<s:schema elementFormDefault="qualified" targetNamespace="http://www.webservicex.net"
<s:element name="GetWeatherByZipCode">
<s:complexType>
<s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="ZipCode" type="s:string" />
</s:sequence>
</s:complexType>
</s:element>
<s:element name="GetWeatherByZipCodeResponse">
<s:complexType>
<s:sequence>
<s:element minOccurs="1" maxOccurs="1" name="GetWeatherByZipCodeResult" type="tns:WeatherForecasts" />
</s:sequence>
</s:complexType>
</s:element>
<s:complexType name="WeatherForecasts">
<s:sequence>
<s:element minOccurs="1" maxOccurs="1" name="Latitude" type="s:float" />
<s:element minOccurs="1" maxOccurs="1" name="Longitude" type="s:float" />
<s:element minOccurs="1" maxOccurs="1" name="AllocationFactor" type="s:float" />
<s:element minOccurs="0" maxOccurs="1" name="FipsCode" type="s:string" />
<s:element minOccurs="0" maxOccurs="1" name="PlaceName" type="s:string" />
<s:element minOccurs="0" maxOccurs="1" name="StateCode" type="s:string" />
<s:element minOccurs="0" maxOccurs="1" name="Status" type="s:string" />
<s:element minOccurs="0" maxOccurs="1" name="Details" type="tns:ArrayOfWeatherData" />
</s:sequence>
</s:complexType>
</s:element>
<s:complexType name="ArrayOfWeatherData">
<s:sequence>
<s:element minOccurs="0" maxOccurs="unbounded" name="WeatherData" type="tns:WeatherData" />
</s:sequence>
</s:complexType>
</s:element>
</s:complexType>
</s:element>
</s:complexType>
<s:complexType name="ArrayOfWeatherData">
<s:sequence>
<s:element minOccurs="0" maxOccurs="unbounded" name="WeatherData" type="tns:WeatherData" />
</s:sequence>
</s:complexType>
</s:element>
</s:complexType>
<s:complexType name="WeatherData">
<s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="Day" type="s:string" />
</s:sequence>
</s:complexType>
</s:element>
</s:complexType>
</s:element>
</s:complexType>
</s:element>
</s:complexType>
</s:element>
</s:complexType>
</s:element>
</s:complexType>
</s:element>
Connectivity details

For the ServiceNow-initiated SOAP requests to successfully communicate with the web service provider inside a remote network, the ServiceNow instance must have HTTP or HTTPS access to the SOAP endpoint at the provider.

Like any integration, such as LDAP, web services, or JDBC, the SOAP endpoint may reside behind a firewall that is blocking inbound communication from the instance. If this is the case, you need to make network changes to allow this connectivity into your network. You can either modify the firewall and ACL rules to allow the instance IP address, configure the SOAP message to use a MID Server, or implement a VPN tunnel to allow the instance communication into your network.

Note: A common misconception is that because asynchronous SOAP requests are routed through the ECC queue, they are always sent through a MID Server. This is not the case. Asynchronous SOAP requests only use a MID Server when configured to do so.
Outbound SOAP security

You can authenticate outbound SOAP messages using several different security protocols.

The security protocol you should use depends on the requirements of the web service provider. Mutual authentication is supported for outbound web services.

Related information

Outbound web services mutual authentication

Enable basic authentication for outbound SOAP

If the endpoint requires a user name and password, you can provide credentials using basic authentication.

Procedure

1. Navigate to System Web Services > Outbound > SOAP Message.
2. Select a SOAP message record.
3. In the SOAP Message Functions related list, select a function.
4. Select Use basic auth.
5. Enter a user name in the Basic auth user ID field.
6. Enter the password for that user in Basic auth user password.
7. Click Update.

Enable WS-Security for outbound SOAP

You can sign outbound SOAP messages using username and password or a key store and trusted server certificate saved on the instance.

Before you begin

Role required: admin

Procedure

1. Navigate to System Web Services > Outbound > SOAP Message.
2. Select a SOAP message record.
3. In the SOAP Message Functions related list, select a function.
4. In the WS-Security type field select the type of credentials to use, such as Username or X.509.
Some web service providers may require you to submit both types of credential.

5. In the **WS-Security x.509 profile** or **WS-Security Username profile** field, select the profile that contains the credentials you want to use. If the record was configured with x.509 security prior to the Helsinki release, you can migrate the security settings to a WS-Security x.509 profile record by clicking the **Migrate to X509 Security Profile** button. When you click this button, security configuration field values (**Key store**, **Certificate**, **Key store alias**, and **Password**) are migrated to a new WS-Security x.509 profile record and that record is referenced from the SOAP message function. If an x.509 profile record already exists with matching values, that record is referenced.

⚠️ **Note:** Existing SOAP message functions that use the legacy security configuration will still work, however new records can use only an x.509 profile record for x.509 security.

6. Click **Update**. Credentials from the selected profile are sent as part of the SOAP message header.

Related information

Create a new WS-Security profile

Configure SOAP with a proxy

Certain properties provide support for SOAP requests to use a web proxy server.

### SOAP properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>glide.http.proxy_host</td>
<td>The proxy server hostname or IP address</td>
<td>proxy.company.com, 192.168.34.54</td>
</tr>
<tr>
<td>glide.http.proxy_port</td>
<td>The port number for the proxy server</td>
<td>8080, 9100</td>
</tr>
<tr>
<td>glide.http.proxy_username</td>
<td>If the proxy server is authenticating using user name and password, enter a value for this property</td>
<td>proxyuser</td>
</tr>
<tr>
<td>glide.http.proxy_password</td>
<td>If the proxy server is authenticating using user</td>
<td>password</td>
</tr>
</tbody>
</table>
### SOAP properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>name and password</td>
<td>enter a value for this property</td>
<td></td>
</tr>
</tbody>
</table>

### Scripting outbound SOAP

You can send outbound SOAP requests from any place in the Now Platform where scripting is allowed.

For detailed API information about the server-side SOAPMessageV2 and SOAPResponseV2 APIs, see the API documentation on developer.servicenow.com.

### Preview a SOAP message script

Generate a sample script to send the SOAP message.

After you have developed and tested the SOAP message, click the **Preview script usage** related link in the SOAP Message Function form. The dialog box displays an example of how you can invoke the SOAP message from a script.

You can manipulate the resulting XML response body with `XMLDocument` or with `gs.getXMLText` and `gs.getXMLNodeList`.

### Direct SOAPMessageV2 example

You can send an outbound SOAP message directly to the endpoint.

In this example, the script sends a SOAP message requesting a stock quote and waits for a response. If there is no response from the web service provider, or if the specified SOAP message record is unavailable, the script throws an error, handled in this example by the try-catch block.

```javascript
var requestBody;
var responseBody;
var status;
var sm;
try{
    sm = new sn_ws.SOAPMessageV2("StockQuote", "GetQuote"); // Might throw exception if
    message doesn't exist or not visible due to scope
    sm.setBasicAuth("admin","admin");
    sm.setStringParameter("symbol", "NOW");
    sm.setStringParameterNoEscape("xml_data","<data>test</data>");
```
Asynchronous SOAPMessageV2 example

You can send an outbound SOAP message asynchronously.

When you send an asynchronous message the instance does not wait for a response before proceeding. You must handle waiting for a response within your code.
You can send an outbound SOAP message through a MID Server.

By sending the message through a MID Server, you can access endpoints that are behind a firewall or within a private network. All SOAP messages sent through a MID Server are asynchronous.

```javascript
var requestBody;
var responseBody;
var status;
var sm;
try{
    sm = new sn_ws.SOAPMessageV2("StockQuote", "GetQuote");  // Might throw exception if message doesn't exist or not visible due to scope
    sm.setBasicAuth("admin","admin");
    sm.setStringParameter("symbol", "NOW");
    sm.setStringParameterNoEscape("xml_data","<data>test</data>");
    sm.setMIDServer('mid_server_name');
    response = sm.execute(); // Might throw exception if http connection timed out or some issue with sending request itself because of encryption/decryption of password and stuff
    response.waitForResponse(60); // In Seconds, Wait at most 60 seconds to get response from ECC Queue/Mid Server // Might throw exception timing out waiting for response in ECC queue
    responseBody = response.haveError() ? response.getErrorMessage() : response.getBody();
    status = response.getStatusCode();
} catch(ex) {
    responseBody = ex.getMessage();
    status = '500';
} finally {
    requestBody = sm ? sm.getRequestBody():null;
}
gs.info("Request Body: " + requestBody);
gs.info("Response: " + responseBody);
gs.info("HTTP Status: " + status);
```
Recordless SOAPMessageV2 example

You can use the SOAPMessageV2() constructor with no parameters to define a SOAP message entirely in the script.

When using this constructor you must provide an endpoint and SOAP action. In this example, the script creates an empty SOAP message and sets the values needed to insert an incident record.

```javascript
var s = new sn_ws.SOAPMessageV2(); //create an empty SOAP message
s.setBasicAuth('admin','admin');
s.setSOAPAction('http://www.service-now.com/incident/insert'); //set the SOAP action to perform
s.setEndpoint('http://<instance>.service-now.com/incident.do?SOAP'); //set the web service provider endpoint
s.setRequestBody('<soapenv:Envelope
    xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:inc="http://www.service-now.com/incident"><soapenv:Header/>
    <soapenv:Body><inc:insert>
        <short_description>Test Dynamic SOAP</short_description>
    </inc:insert></soapenv:Body></soapenv:Envelope>');
var response = s.execute();
var xmldoc = new XMLDocument(response.getBody());
var incident_sysid = xmldoc.getNodeText('//sys_id');
```

Outbound web services mutual authentication

Mutual authentication establishes trust by exchanging Secure Sockets Layer (SSL) certificates.

Before connecting to a server, the client requests an SSL certificate. The server responds by requesting that the client send its own certificate. Both respond by validating the certificates of the other and sending acknowledgments before initiating an HTTPS connection. Mutual authentication is not available for inbound requests or for outbound web service calls through a MID Server.

As an administrator, you can enable mutual authentication by defining a protocol profile for connections that require mutual authentication. Protocol profiles allow you to associate a specific certificate record with a protocol, such as HTTPS.

For example, you can create a protocol profile called `mauth` with a certificate for mutual authentication. You can then make an HTTPS web service request...
by calling `mauth://<external-endpoint.com>` when the end point requires mutual authentication.

⚠️ **Note:** This feature enables mutual authentication only on outbound HTTPS connections, such as SOAP, REST, or direct HTTPS calls. ServiceNow does not support mutual authentication for inbound requests or for outbound requests sent through a MID Server.

## Create a protocol profile

You can create a custom HTTPS protocol profile to specify the credentials and certificates used for outbound web services. For example, you can create a custom HTTPS protocol profile to enable mutual authentication.

### Before you begin

- **Role required:** admin
- **Upload a certificate** to authenticate the client certificate of the instance.
- **Upload a trusted server certificate** to authenticate the server certificate of the web service provider.

### About this task

Mutual authentication causes the web service provider and consumer to authenticate with each other before communicating. You must configure a REST or SOAP message to support mutual authentication.

- **Outbound REST:** Create a REST message using basic authentication. Mutual authentication is not available with OAuth 2.0.
- **Outbound SOAP:** Create a SOAP message using mutual authentication.

⚠️ **Note:** Mutual authentication is not available when making outbound web service calls through a MID Server.

### Procedure

1. Navigate to **System Security > Protocol Profiles**.
2. Click **New**.
3. Fill in the fields on the form, as appropriate.
Name of form

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol</td>
<td>Enter a unique name to identify this HTTPS protocol, such as <code>mauth</code>. The protocol name allows you to differentiate between normal HTTPS connections and HTTPS connections that use this protocol profile. The name you enter becomes the protocol name in the URL. For example, <code>mauth://endpoint.service.com</code></td>
</tr>
<tr>
<td>Note:</td>
<td>You cannot create a custom protocol whose name matches as an existing protocol name such as HTTPS.</td>
</tr>
<tr>
<td>Default port</td>
<td>Enter the port number for connections that use this protocol.</td>
</tr>
<tr>
<td>Check box</td>
<td>Select the client certificate the protocol uses to authenticate itself with the web service provider. The certificate must have a Value of Java Key Store or PKCS12 Key Store.</td>
</tr>
</tbody>
</table>

**Enable mutual authentication**

You can configure a SOAP or REST message for mutual authentication using a protocol profile

**Before you begin**

Role required: web_service_admin or admin

**Procedure**

1. Navigate to System Web Services > SOAP Message or System Web Services > REST Message.
2. Select a message record.
3. Select the Use mutual authentication check box.
4. In the Protocol profile field, select a protocol profile configured for mutual authentication.
5. Click Update.

**Outbound web services logging**

Log requests and responses for outbound web services such as REST and SOAP.

Outbound request logging allows you to better understand what 3rd-party services your instance accesses and the volume of outbound requests.
Additionally, logging can provide valuable information when debugging outbound integrations.

Outbound web services logging tracks outbound REST and SOAP requests, as well as outbound requests made using the GlideHTTPRequest and GlideHTTPClient APIs.

To view outbound web service logs, navigate to System Logs > Outbound HTTP Requests.

All log information is read only.

**Configure outbound logging**

You can configure outbound request logging to log basic, elevated, or all HTTP request and response information for specific domains.

To configure the log level for a REST method or SOAP message function, navigate to the record you want to configure and click the **Set log level** related link, then select a log level for the current record.

To modify the log level for multiple outbound requests, navigate to System Web Services > HTTP Log Levels and change the log levels using the list. All outbound requests that have been configured with a specific log level are listed.

You can override the log level for all outbound requests using the properties `glide.outbound_http_log.override` and `glide.outbound_http_log.override.level`. Use these properties only for a limited time when troubleshooting.

You can set the log level in a script using the `setLogLevel()` function from the SOAPMessageV2 and RESTMessageV2 APIs. For more information about using these APIs, refer to the API documentation.

**Related information**

- RESTMessageV2 - `setLogLevel()`
- SOAPMessageV2 - `setLogLevel()`

**Outbound request log levels**

Certain elements are logged based on the configured log level.

**Logged elements**

The following elements from the request and response are logged depending on the configured log level.
Logged elements by level

<table>
<thead>
<tr>
<th>Field</th>
<th>Basic</th>
<th>Elevated</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>HTTP Method</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Protocol</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Scheme</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Hostname</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Path</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>HTTP response status</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Request length</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Response length</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Total call time</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>System ID</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Source table</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Source record ID</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

**Note:** You must have read access for records in the specified **Source table** to view this field.

<table>
<thead>
<tr>
<th>Field</th>
<th>Basic</th>
<th>Elevated</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Session</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Transaction</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>User</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>MID server</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Request headers</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Request query</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Response headers</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Request body</td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Response body</td>
<td></td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>
Outbound request logging exclusion domain requirements

When you exclude outbound request logging for a domain, the value you enter in the Domain field must meet certain requirements.

**Requirements**
The value you enter in the Domain field must meet the following requirements.

- Begins with `HTTP://` or `HTTPS://`.
- Is a domain pattern or IP address.
- Ends with alphanumeric characters preceded by a period, such as `.com`.
- Includes at most a single wildcard character immediately following the scheme and hierarchical portion of the domain pattern.

**Wildcard**
You can use a single wildcard character (`*`) in the domain pattern. Use this wildcard immediately following the scheme and hierarchical portion of the domain pattern, such as `http://*.domain.com` to include all subdomains. The wildcard must immediately follow the scheme and hierarchical portion of the domain pattern. If you use an IP address instead of domain pattern, you must enter the full IP address without a wildcard.

**Note:** You cannot use multiple wildcards, or specify a wildcard without a domain pattern. Values such as `*` or `*.*` are not supported.

**Domain matching**
When evaluating the Origin header in a request, ServiceNow prioritizes rules that match the domain pattern exactly. If no exact match is found, the next closest match is used.

For example, if there are rules for the domain patterns `http://*.blog.mysite.com` and `http://*.mysite.com`, a request from `http://alice.blog.mysite.com` will match the `http://*.blog.mysite.com` pattern.

**Examples of valid and invalid domains**

<table>
<thead>
<tr>
<th>Valid domain</th>
<th>Invalid domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>http://*.ms.net</td>
<td>https://*com</td>
</tr>
<tr>
<td>https://*.ms.com</td>
<td>http://*.com</td>
</tr>
</tbody>
</table>
Examples (continued)

<table>
<thead>
<tr>
<th>Valid domain</th>
<th>Invalid domain</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://192.168.1.1">http://192.168.1.1</a></td>
<td>http://*.168.1.126</td>
</tr>
<tr>
<td>http://*.service-now.com</td>
<td><a href="http://blog.*.service-now.com">http://blog.*.service-now.com</a></td>
</tr>
<tr>
<td>http://*.com</td>
<td>http://*com</td>
</tr>
</tbody>
</table>

**Outbound request logging exclusion**

You can exclude domains to allow only basic-level logging on those domains.

To exclude a domain, navigate to **System Web Services > Outbound HTTP Log Domain Exclusion List** and create a new record. You must have the **web_service_admin** role to exclude domains.

**Outbound web service logging properties**

These properties allow you to control the behavior of outbound web service request logging.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>glide.outbound_http.db.log</td>
<td>By default, log information is written to a log file and to the instance database. Set this property to false to disable database logging.</td>
</tr>
<tr>
<td></td>
<td>• Type: true</td>
</tr>
<tr>
<td></td>
<td>• Default value: true</td>
</tr>
<tr>
<td></td>
<td>• Location: Add to the System Property [sys_properties] table</td>
</tr>
<tr>
<td>glide.outbound_http.text.content_types</td>
<td>A comma-separated list of content types. The body of requests or responses with one of these content types is logged. By default, the content types text/*, application/json, and application/xml are always logged. Use this property to add additional content types.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>glide.outbound_http.content.max_limit</td>
<td>The maximum number of characters logged from a request or response body. The first characters of the body, up to this limit, are logged. This property has a maximum possible value of 1000.</td>
</tr>
<tr>
<td>glide.outbound_http_log.override</td>
<td>When this property is set to true, the property <code>glide.outbound_http_log.override.level</code> determines the log level for all requests and responses. This log level overrides any other log level settings. Only use this property for a limited time when troubleshooting.</td>
</tr>
<tr>
<td>glide.outbound_http_log.override.level</td>
<td>The log level to use for all requests and responses when <code>glide.outbound_http_log.override</code> is true. Valid values are <code>basic</code>, <code>elevated</code>, and <code>all</code>.</td>
</tr>
</tbody>
</table>

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Server Name Indication for outbound web services

Outbound HTTP requests, including outbound SOAP, REST, and GlideHTTPClient requests support Server Name Indication.

Server Name Indication (SNI) is a TLS extension that enables service providers to serve multiple host names from a single IP address. With SNI support, outbound web service requests specify the hostname to connect to as part of the TLS handshake. This information allows the server to present the correct certificate to the client.

SNI support is provided for outbound requests sent directly from an instance, as well as outbound requests sent through a MID Server. The hostname is automatically included in the HTTP request when SNI is enabled, it is not necessary to configure the request for SNI support.

For example, when SNI support is enabled, a REST message sent to the endpoint https://myhost.com/some/path will include the domain myhost.com in the TLS SNI extension.

Configure Server Name Indication

Server Name Indication support is disabled by default for all outbound HTTP requests.

To enable or disable SNI support for outbound HTTP requests sent directly from the instance, set the system property `glide.outbound.tls_sni.enabled` to true or false (default).

To enable or disable SNI support for outbound HTTP requests sent through a MID Server, create the property `glide.outbound.tls_sni.enabled` on the MID Server with a value of true or false (default), then restart the MID Server.

⚠️ Note: After you change the property, wait at least 30 seconds before testing a request to ensure that a cached connection is not reused.

HTTP Client Connection Management

Outbound HTTP(S) connections from a base system instance or inbound connections from MID Servers, the ODBC driver, and other clients are maintained and reused where possible.

Connection pooling is used to keep track of HTTP(S) client connections to determine if they are alive and available for reuse.

ServiceNow HTTP client code means:
• Any application or script which makes outbound HTTP(S) requests from a base system instance.
• ServiceNow code in the MID Server or the ODBC driver which makes HTTP(S) requests to one or more base system instances.

⚠️ Note: This discussion does not apply to browser-to-instance communication. No changes have been made with respect to the management of HTTP(S) connections for browser-based communication with ServiceNow. This discussion also does not apply to customer-developed Web Services clients making requests to ServiceNow.

What Should the Customer Do?

Users should monitor performance, such as the decreased time for loading Discovery data and improved ODBC driver performance. For systems with an unusually large amount of simultaneous outbound HTTP(S) activity, such as numerous third-party integrations or high-volume automated activities which generate HTTP(S) requests from the base system instance to other places, review the `max_connections` and `max_connections_per_host` properties to ensure that the settings are sufficient. This enhancement has no impact on end-user connections from browsers and no impact on connections from customer-developed Web Services client applications.

HTTP Connection Management Properties

Connection pooling is controlled by three properties.

The default values for these properties are appropriate for most customers. The Glide properties are dynamic, meaning that changes to these properties will take effect immediately. No outage or restart is required to update the values.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>glide.http.connection_mgr.max_connections</td>
<td>Controls the total number of permitted HTTP(S) connections outbound from the base system instance. This is an instance-wide value.</td>
<td>20</td>
</tr>
<tr>
<td>glide.http.connection_mgr.max_connections_per_host</td>
<td>Controls how many of the <code>glide.http.connection_mgr.max_connections</code> can communicate in</td>
<td>4</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Default Value</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>parallel with any particular host. If the maximum setting for any of these values is reached during normal operations, a script or background thread may have to wait briefly to obtain a connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>glide.http.outbound-max_timeout</td>
<td>The max timeout in seconds to wait for SOAPMessageV2 and RESTMessageV2 when using executeAsync method. The maximum value allowed is 30 seconds. If this value is set to something higher than 30 seconds, it will default to 30 seconds.</td>
<td>30</td>
</tr>
<tr>
<td>glide.http.outbound-max_timeout.enabled</td>
<td>Determines whether a configurable hard maximum wait time is enforced for SOAPMessageV2 and RESTMessageV2 when using executeAsync method.</td>
<td>true</td>
</tr>
<tr>
<td>glide.http.use_connection_mgr</td>
<td>Switches connection pooling on and off. To disable the new behavior (not recommended) set glide.http.use_connection_mgr to false.</td>
<td>true</td>
</tr>
</tbody>
</table>

**ODBC driver**

The ServiceNow open database connectivity (ODBC) driver provides read-only access to the database associated with your ServiceNow instance.
The ODBC driver is compliant to version 3.52 of the Microsoft ODBC core API conformance. The ServiceNow ODBC driver uses ServiceNow web services support for a query-only interface. The ODBC driver supports only select statements or read-only functions and does not modify your instance data. Because the ODBC driver uses the web services interface, platform-wide access control (ACL) is enforced and data security is in place.

⚠️ **Note:**
The ODBC driver has these limitations:

- The ODBC driver supports only select statements or read-only functions, and does not modify your instance data.
- There is no supported way to use the ODBC driver with a Java client application or with a Java JDBC-ODBC bridge.
- There is a hard-coded limit of 512 characters when accessing views through ODBC. Because of this limitation, a maximum of 11 table sys_ids can be included in an ODBC view query. Anything over 11 tables results in an error.

⚠️ **Note:** Versions of the ODBC driver older than 1.0.7.3 are no longer supported.

### Getting started with ODBC

Before installing the ODBC driver, view the video on this page, create an ODBC user account, assign the odbc role, and define an ACL rule for the odbc role.

A ServiceNow user with the admin role can perform these procedures.

Watch this seven-minute video for an overview of installing, configuring and testing the ServiceNow ODBC driver, which provides read-only access to the database associated with your ServiceNow instance. This video explains the ODBC prerequisites and demonstrates how to install, configure, and test the ServiceNow ODBC driver, which provides read-only access to the database associated with your ServiceNow instance.

Before downloading and installing the ODBC driver, review the requirements to ensure that your configuration is compatible.

### Create an ODBC user account and assign the odbc role

The ODBC driver communicates with your ServiceNow instance as a specific user. Create an ODBC user account and assign the odbc role to enable the user to communicate via ODBC.
Before you begin
Role required: admin

About this task
The odbc role contains various additional roles, including the soap_query role required to make ODBC requests and the itil role required to access core tables such as the incident table. You can modify the odbc role to allow access to other tables as needed by adding additional child roles.

Procedure
1. Navigate to User Administration > Users.
2. Click New.
3. In the User ID field, enter odbc.user.
4. Enter a Password for this user.
5. From the form context menu, select Save.
6. In the Roles related list, click Edit.
7. Use the slushbucket to add the odbc role, and then click Save.
8. Click Submit.

What to do next
In a separate browser session, confirm that the odbc.user is able to log in to your ServiceNow instance.

Define an ACL rule for the odbc role
Define an ACL rule for the odbc role to provide read access to the incident table. You can create other ACL rules for the odbc role to provide read access to other tables.

Before you begin
Role required: admin

Procedure
1. Elevate the session permissions to security_admin so you can create ACL rules.
3. Click New.
4. From the Operation choice list, select read.
5. From the Name choice list, select Incident [incident].
Leave the second **Name** choice list as **None**.

6. From the form context menu, select **Save**.

7. In the **Requires role** related list, click **Edit**.

8. Use the slushbucket to add the ODBC role, and then click **Save**.

9. Click **Submit**.

### Installing the ODBC driver

Review setup requirements, download the ODBC driver installer, and install the ODBC driver to a computer.

You can install the ServiceNow ODBC driver on Microsoft Windows computers. To install the ODBC driver, set up an ODBC user in your ServiceNow instance, then download and install the ODBC driver. If you already have the ODBC driver installed, you can upgrade to the newest version.

### ODBC driver installation requirements and supported software

Install the ServiceNow ODBC driver on Microsoft Windows computers.

### Installation requirements

Before installing the ODBC driver, ensure that your configuration meets these requirements.

<table>
<thead>
<tr>
<th>Category</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active user</td>
<td>The user record on the instance used to perform the queries.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>The account used to connect to the instance via the ODBC driver must be defined on the instance. Accounts using single sign-on are not supported by the ODBC driver.</td>
</tr>
<tr>
<td>The soap_query role</td>
<td>The user you use to query the database must have the soap_query role if the instance uses the glide.soap.strict_security high security setting.</td>
</tr>
<tr>
<td><strong>Warning:</strong></td>
<td>Do not enable WS-Security for all SOAP requests by setting the glide.soap.require_ws_security system property. It is incompatible with the ODBC driver. Enabling this setting blocks both ODBC driver and MID Server connections. Instead, use basic authentication.</td>
</tr>
</tbody>
</table>
## Installation requirements (continued)

<table>
<thead>
<tr>
<th>Category</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Table ACLs</td>
<td>The user you use to query the database must have read access for the tables that you want to query. For example, a user with the itil role can read task tables, such as Incident.</td>
</tr>
<tr>
<td>Target Table Web Service Access</td>
<td>The table you want to query must allow web service interaction. You can enable web service interaction using the application access settings.</td>
</tr>
<tr>
<td>Java Runtime Environment (JRE)</td>
<td>JRE must be enabled on your instance.</td>
</tr>
<tr>
<td>Operating System</td>
<td>The ServiceNow ODBC driver supports installation on the following operating systems:</td>
</tr>
<tr>
<td></td>
<td>• Windows XP</td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2003</td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2008</td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2012</td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2016</td>
</tr>
<tr>
<td></td>
<td>• Windows Vista</td>
</tr>
<tr>
<td></td>
<td>• Windows 7</td>
</tr>
<tr>
<td></td>
<td>• Windows 8</td>
</tr>
<tr>
<td></td>
<td>• Windows 10</td>
</tr>
<tr>
<td>Hardware</td>
<td>• RAM: 1 GB minimum</td>
</tr>
<tr>
<td></td>
<td>• Disk space: 135 MB for installation, 200 MB for writing cache files during usage</td>
</tr>
<tr>
<td>Account</td>
<td>The Windows account used for the installation must have local Administrator rights to install an ODBC driver.</td>
</tr>
<tr>
<td>Networking</td>
<td>During usage, the ODBC driver requires HTTPS (port 443) connectivity to the ServiceNow instance. The communication between the ODBC driver and the ServiceNow instance uses standard SOAP web services.</td>
</tr>
</tbody>
</table>
### Installation requirements (continued)

<table>
<thead>
<tr>
<th>Category</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>End User License Agreement</td>
<td>Read the End User License Agreement for the ServiceNow ODBC driver.</td>
</tr>
</tbody>
</table>

For more information, see Application access settings.

### Supported software

The following table lists the operating systems and reporting applications compatible with each version of the ODBC driver.

#### Supported software

<table>
<thead>
<tr>
<th>Driver Version</th>
<th>Operating System</th>
<th>Microsoft Excel</th>
<th>Microsoft SQL Server</th>
<th>Crystal Reports</th>
<th>Tableau</th>
<th>Informatica</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0.13 and later</td>
<td>Windows XP SP2</td>
<td>2007, 2010</td>
<td>2008</td>
<td>2008</td>
<td>8.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Windows 7</td>
<td></td>
<td></td>
<td>2013</td>
<td>8.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Windows 8.x</td>
<td></td>
<td></td>
<td></td>
<td>9.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Windows 10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Windows Server 2003</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Windows Server 2008</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Windows Server 2008 R2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The ODBC driver provides only basic support for Informatica. Use the ODBC driver with Informatica only for simple operations. Thoroughly test integrations with Informatica before using them in a production environment.
## Supported software (continued)

<table>
<thead>
<tr>
<th>Driver Version</th>
<th>Operating System</th>
<th>Microsoft Excel</th>
<th>Microsoft SQL Server</th>
<th>Crystal Reports</th>
<th>Tableau</th>
<th>Informatica</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Windows Server 2012</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Windows Server 2012 R2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Windows Server 2016</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.0.9 - 1.0.12</td>
<td>• Windows XP SP2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows Vista</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows 8.x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2003</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2008</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2008 R2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2012</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2012 R2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Supported software (continued)

<table>
<thead>
<tr>
<th>Driver Version</th>
<th>Operating System</th>
<th>Microsoft Excel</th>
<th>Microsoft SQL Server</th>
<th>Crystal Reports</th>
<th>Tableau</th>
<th>Informatica</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0.8 and earlier</td>
<td>• Windows XP SP2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows Vista</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows 8.x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2003</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2008</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2008 R2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Download and install the ODBC driver

Download the ODBC driver from the ServiceNow Knowledge Base and install the driver for the first time.

Before you begin

You must have administrator-level access for the Windows computer onto which you want to install the ODBC driver.

About this task

If this is the first time the driver is installed, the installer is in first-time installation mode and prompts for the driver to be installed. Install only one version of the ODBC driver on a computer. If the ODBC driver was previously installed, the installer is in upgrade mode and prompts for removal of the previous driver first.

Note: Versions of the ODBC driver older than 1.0.7.3 are no longer supported.
Procedure

1. To download the latest ODBC drivers, click the following links:
   - 1.0.14_01 (32 bit) download
   - 1.0.14_01 (64 bit) download

2. Download the ODBC driver version compatible with your computer’s operating system and the application you are using to query the database. The ODBC driver is available as 32-bit or 64-bit. Most applications require the 32-bit ODBC driver even if the operating system is 64-bit.

3. Right-click the executable and select Run as Administrator to launch the installer. The ODBC Driver Setup Wizard appears.

4. Click Next.

5. Read and accept the End User License Agreement.

6. The installer prompts for the Installation Directory. Select the target directory for installing the ServiceNow ODBC driver. The default directory is C:\Program Files\Service-now\ODBC.

7. The installer prompts for the Service Name. Enter a name to identify the service (for example, ServiceNow_ODBC) and press Next.

8. The installer prompts for the Java Virtual Machine Location.
• Browse to and select the directory where the jvm.dll file is located (usually C:\Program Files\Java\jre<version>\bin\server).

• If you do not want to enter the JRE location at the time of the installation, click **Next**.
  On the popup window, click **Yes** to enter the JRE location now, or click **No** to enter the JRE location later.

**Important:** You must specify the JRE location before the ODBC driver will work. After the installation, you can access the Management console as an administrator and navigate to **Services > <Service_Name> > Service Settings > IP Parameters** and enter the JRE location in the **ServiceJVMLocation** property.

**Tip:** For the configuration of a Java service, ServiceNow strongly recommends using the server jvm.dll file from the Java Development Kit (JDK).

9. The installer prompts for the **ODBC Data Source** details. Specify the following parameters, which are required to create an ODBC data source that can be used to create a DSN.

• **Data Source Name**: A short name to identify this data source.

• **Description**: A short description of the driver. The driver's version number is appended to the end of this value.

• **Service Name**: Enter the same **Service Name** entered at step 7.

• **Service Data Source**: The name that can be selected in the **Service Data Source** field of the ODBC Administrator. Usually the default values are appropriate.
10. Select the **Program Folder** to create links for the driver. This is the program folder that appears under the **Start** menu.
The installation creates the following links in the menu.

- **Interactive SQL (ODBC):** An interactive SQL command window for directly testing SQL statements.
- **Management Console:** A Microsoft MMC snap-in for configuring default properties for the ODBC driver.
- **ODBC Administrator:** A Microsoft ODBC Administrator program.

The driver code is copied to the target folder.

A progress bar appears.

11. When prompted, click **Finish** to complete the installation.
Upgrade the ODBC driver

If you have previously installed an older version of the ODBC driver, run the installer to uninstall the previous version, and then run the installer again to upgrade.

About this task
To check the build date and time of the ODBC driver, use CheckVersion located in the Service-Now\ODBC\ip\tools folder. This is an executable Windows host script that reports the build date and time of the current ODBC driver. Use it to assist ServiceNow Technical Support to determine which build of the ODBC driver is running. If the CheckVersion tool is absent, the ODBC driver is out of date; upgrade to the current version. To check the version of an older ODBC driver, see the previous version information.

Note: The ODBC installation also has a Service-Now\ODBC\tools folder, which is not the correct path for the CheckVersion tool.

Procedure
1. Right-click the executable and select Run as Administrator.

2. Click OK when prompted to uninstall the current driver, which is required for the upgrade.

   A list appears, displaying the existing ODBC DSN names that you have previously created. You have the option to delete them.

3. Select Yes to remove all previous DSNs or No to keep them for use with the upgraded driver.

   An ODBC DSN is a connection handle to use the ODBC driver in an application. For more information from Microsoft, see Administer ODBC data sources.
4. After removing the previous ODBC driver, double-click the executable again to run the installer. Then, follow the steps in Download and install the ODBC driver.

If you encounter errors when uninstalling the ODBC driver, see the Troubleshooting uninstalling ODBC driver knowledge article.

What to do next
After installing the ODBC driver, configure it to connect to your ServiceNow instance.

Install an ODBC driver patch
Use ServiceNow patches to install incremental ODBC fixes that occur between major ODBC releases.

Before you begin
Incremental ODBC patches contain only fixes to the ODBC driver and do not include fixes for third-party products such as Progress SDK. Third-party product fixes are included only in major ODBC release updates.
You must have administrator-level access for the Windows computer onto which you want to install the ODBC driver patch.
Before applying a new ODBC patch:
• Ensure that you have the Java Runtime Environment (JRE) already installed in your instance.
• Close any applications currently using the ODBC driver.

Procedure
1. Download and unzip the ODBC patch zip file from KB0540707 in the ServiceNow Knowledge Base.
2. Extract the zip files into a folder named odbc-patch.
3. From the Start menu, right-click Command Prompt and select Run as administrator.
4. In the Command Prompt window, cd to the odbc-patch folder created in step 2, type update.bat, and press Enter.
5. The installer prompts for the installation directory of the ODBC driver. The default installation location is C:\Program Files\ServiceNow\ODBC.
   • If the ODBC driver is installed in C:\Program Files\ServiceNow\ODBC, enter y and press Enter.
   • If the ODBC driver is not installed in C:\Program Files\ServiceNow\ODBC, enter the complete path to the ODBC installation directory and press Enter.
The installer validates the installation location. If the location specified is not valid, the installer prompts for the correct location of the installed ODBC driver.

Once you enter a valid installation location, the installer checks for the installed version and compares it with the version of the patch.

   • If the currently-installed version is the latest version, the installer aborts the installation process and displays the following message:

     The version you are trying to install is same as the existing version. Installation aborted.

   • If the currently-installed version is the same as the patch version, the installer aborts the installation process and displays the following message:

     The current installed version is the most recent version. Installation aborted.

   • If the patch version is the latest version, the installer continues with the patch installation. Once the installation is complete, you receive the following confirmation message:

     Patch is successfully installed.
Configuring the ODBC driver

After installing the ODBC driver, configure it to connect to your ServiceNow instance and to communicate through a proxy server, if applicable, and set properties to control ODBC behavior.

Configure the ODBC driver

Configure the ODBC driver to connect to your ServiceNow instance.

Before you begin

You must have administrator-level access for the Windows computer on which you want to configure the ODBC driver.

About this task

After the driver is installed, configure it for your instance. The driver is preconfigured to connect to https://demoodbc.service-now.com using the DSN ServiceNow. There are two ways to configure connectivity for the driver.

- Configure the global default used by all newly created DSNs.
- Configure each new DSN with its own connection.

Procedure

1. In Windows, navigate to Start > Programs > ServiceNow ODBC Management Console.
2. Expand the Console Root tree to: ServiceNow ODBC Manager\Manager \<installation location>\Services\ServiceNow_ODBC\Data Source Settings \ServiceNow\IP Parameters.
3. Double-click the DataSourceIPProperties attribute.
4. Change the Value to the URL of your instance, such as https://<instance>.service-now.com.
   If integrating the ODBC driver with Edge Encryption, change the Value to the URL of your encryption proxy. See Integrate Edge Encryption with the ODBC driver for more information.
5. Click OK.

Configure the global DSN default

Configure the global default used by all newly created DSNs.
Before you begin
You must have administrator-level access for the Windows computer on which you want to configure the global DSN default.

About this task
A default DSN is preloaded with the ODBC driver installation ServiceNow data source. This preloaded DSN connects using the default connection URL, which is set to https://demo.service-now.com. To change the global default for the instance URL, do the following.

Procedure
1. In Windows, navigate to Start > Programs > ServiceNow ODBC > Management Console.
2. Expand the Console Root tree using the following path:
   OpenAccess SDK 6.0 Manager\<installation location>\Services\ServiceNow_ODBC\Data Source Settings\ServiceNow\IP Parameters
3. Double-click the DataSourceIPProperties attribute for the ServiceNow data source setting to open the Properties dialog box.
4. Change the value to the URL of your instance, using the following format, and then click OK:
Create a new DSN

Use the ODBC driver and the ServiceNow data source to create an unlimited number of DSNs configured to connect with different instance URLs.

Before you begin
You must have administrator-level access for the Windows computer on which you want to create a new DSN.

About this task
Select the target instance for your ODBC connection by DSN name. As an option during installation or upgrade, you can elect to keep the DSNs when you uninstall.
Instead of creating a new DSN, you can Specify a connection string to connect with different instance URLs.
Procedure

1. In Windows, navigate to Start > Programs > Service-now ODBC > ODBC Administrator.

2. To create a system DSN, select the System DSN tab, and then click Add.
3. Select ServiceNow **ODBC driver 32-bit** from the list, and then click **Finish**.

4. Configure the driver and its connection URL by specifying the `url=` parameter value in the **Custom Properties** field. For example:
   
   `url=https://myinstance.service-now.com`

5. Click **OK**.

**Results**

You can now use the new driver.

**Specify a connection string**

You can specify a connection string instead of defining a DSN.

**Before you begin**

You must have administrator-level access for the Windows computer on which you want to specify a connection string.

**About this task**

This is an alternative method of connecting with different instance URLs. See also **Create a new DSN**.

A connection string must follow this format:

```
Driver=ServiceNow ODBC driver 32-bit;ServiceName=ServiceNow_ODBC;UID=youruser;PWD=yourpassword;ServerDataSource=ServiceNow;CustomProperties=url=https://<instance>.service-now.com
```
The driver name varies depending on whether you use the 32-bit or 64-bit version of the ODBC driver. To determine your driver name, do the following.

**Procedure**

1. In Windows, navigate to **Start > Programs > Service-now ODBC > ODBC Administrator**.
2. Select the **System DSN** tab.
3. Note the value in the **Driver** column for the ServiceNow data source.

**Configure the logging level of the ODBC driver**

Change the logging level of the ODBC driver.

**Before you begin**

You must have administrator-level access for the Windows computer on which you want to configure the logging level.

**Procedure**

1. In Windows, navigate to **Start > Programs > ServiceNow ODBC > Management Console**.
2. [ODBC version 1.0.8] Within the management console, navigate to  
   `<your_installation_directory> > Services > ServiceNow_ODBC > Service Settings > IP Parameters`.  
3. [ODBC version 1.0.8] Change the value of the **ServiceJVMOptions** attribute to the desired logging level.
4. Within the management console, navigate to `<your_installation_directory> > Services > ServiceNow_ODBC > Service Settings > Logging.

5. Change the value of the `ServiceDebugLogLevel` by selecting all available check boxes.
6. In Windows, navigate to **Start > Programs > ServiceNow ODBC > ODBC Administrator**.

7. In the ODBC Administrator, select the **Tracing** tab.

8. Navigate to the path in the **Log File Path** field and delete the old log file, if it exists.

9. Click **Start Tracing Now**.

10. Enable SOAP debugging for your ServiceNow instance.

**Related information**

- Enable debug logging
- Debug incoming SOAP envelope

**Configure the ODBC driver for large data sets**

You can set two ODBC driver properties to deal with errors you receive when using queries that return large amounts of data.
Before you begin
You must have administrator-level access for the Windows computer on which you want to configure the ODBC driver for large data sets.

About this task
These properties are set using the ODBC Management Console available on Windows operating systems. For more information, see ODBC management console properties.

Procedure
1. In Windows, navigate to Start > Programs > ServiceNow ODBC > Management Console.
2. Navigate to Data Source Settings > ServiceNow > IP Parameters.
3. Set the Timeout property to be more than the glide.soap.request_processing_timeout.odbc value.
4. Navigate to Services > Service Settings > SQL Engine Parameters.
5. Increase the ServiceSQLDiskCacheMaxSize property.
   Typically, when running a query that returns 50,000 rows, the default value of 200 must be increased.

Configure ODBC to use proxy servers
The ODBC driver can be configured to route its HTTP SOAP requests via an HTTP proxy server.

Before you begin
You must have administrator-level access for the Windows computer on which you want to configure ODBC to use proxy servers.

About this task
Setting up a proxy server gives you the option to control access to the ServiceNow instance from the proxy server, and potentially allows a network configuration that can monitor usage statistics. However, because the proxy server intercepts the ODBC driver's requests to your ServiceNow instance, it will degrade the performance of the driver.

Note: This feature is recommended for use with ODBC driver builds dated 7/15/2011 or later.

To enable the use of proxy servers, the custom properties for proxy server settings must be defined first for the data source. After that, these properties can be overridden by specific ODBC DSNs.
Procedure

1. In Windows, navigate to **Start > Programs > ServiceNow ODBC > Management Console**.

2. Expand the Console Root tree to: **ServiceNow ODBC Manager\Manager \<installation location>\Services\ServiceNow_ODBC\Data Source Settings \ServiceNow\IP Parameters**.

3. Double-click the **DataSourceIpCustomProperties** attribute.

4. Set the following custom properties to configure the ODBC proxy server.

   **Configure ODBC to use proxy servers**

<table>
<thead>
<tr>
<th>Property name</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>proxy_host</td>
<td>The proxy server host name or IP address.</td>
<td>proxy.company.com</td>
</tr>
<tr>
<td>proxy_port</td>
<td>The proxy server port number.</td>
<td>8080</td>
</tr>
<tr>
<td>proxy_user_name</td>
<td>The proxy server user name or id, used in an authenticating proxy configuration.</td>
<td>odbc_user</td>
</tr>
<tr>
<td>Property name</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>proxy_user_password</td>
<td>The proxy server user password, used with the proxy_name value in an authenticating proxy configuration.</td>
<td>*****</td>
</tr>
</tbody>
</table>

### Setting ODBC properties

The following properties customize connectivity and optimize the query behavior of the ODBC driver.

### ODBC administrator properties

These properties are specified in the ODBC Data Source Administrator for the DSN or in the **Custom Properties** field of the login dialog box.

### ODBC administrator properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>BatchSize</td>
<td>During fetching of results from the instance, this batch size configures the number of records to fetch for every request. Typically, the default is an optimal number for normal sized rows. If an error occurs during fetching of records that indicates this value should be lowered, you can modify it to optimize memory usage versus performance.</td>
<td>2000</td>
</tr>
<tr>
<td>url</td>
<td>This is the ServiceNow instance URL or endpoint. It should indicate the URL to the ServiceNow instance you want to connect to.</td>
<td><a href="https://demo.service-now.com">https://demo.service-now.com</a></td>
</tr>
<tr>
<td>EnablePassThrough</td>
<td>During processing of aggregate functions, enabling pass through mode allows directly calling Aggregate Web Service for optimized and speedy response. Whenever possible, this mode should be left enabled.</td>
<td>true</td>
</tr>
<tr>
<td>Property Name</td>
<td>Description</td>
<td>Default</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
<td>---------</td>
</tr>
<tr>
<td>debug</td>
<td>By default, debugging messages are not produced. Set <code>debug</code> to <code>true</code> when you operate the ODBC driver from the ISQL console window to write all HTTP-related network communication traffic to the console window. When using this option, set <code>gzip</code> to <code>false</code> so that data is not compressed. Otherwise, the data is unreadable.</td>
<td><code>false</code></td>
</tr>
<tr>
<td>gzip</td>
<td>By default, data sent over the network is compressed. Set <code>gzip</code> to <code>false</code> when using the <code>debug</code> parameter to write network communication to the ISQL console so that data is not compressed.</td>
<td><code>true</code></td>
</tr>
<tr>
<td>timeout</td>
<td>Specifies the socket inactivity timeout value in seconds.</td>
<td><code>175</code></td>
</tr>
<tr>
<td>retries</td>
<td>Number of times to retry the failing request in the event of a socket timeout error.</td>
<td><code>0</code></td>
</tr>
<tr>
<td>mode</td>
<td>The query mode used to parse complex where clauses. You can configure the ODBC driver query mode to use either AND or OR operators. While the OR operator provides the greatest compatibility with complex queries, the AND operator is usually more efficient and results in fewer database operations.</td>
<td><code>or</code></td>
</tr>
<tr>
<td>EnableDBSchema</td>
<td>The ODBC driver issues a database schema request to retrieve table names from the instance. This functionality is enabled by default so reporting applications such as Microsoft Excel can display a list</td>
<td><code>true</code></td>
</tr>
</tbody>
</table>
### ODBC administrator properties (continued)

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>of tables to query from. Disabling this property may improve the performance of the first query sent from a reporting application, especially if the instance has a large number of tables.</td>
<td></td>
</tr>
<tr>
<td>ExtendedSchemaCache</td>
<td>The ODBC driver caches the database schema for each connection. When a new connection is created, the driver clears the database cache and queries the database schema from the instance again. This behavior is beneficial when connecting to different data source, or when modifying the table schema. When querying a single data source with a consistent schema, enable this property to avoid sending unnecessary schema requests, including when <code>EnableDBSchema</code> is <code>true</code>.</td>
<td>false</td>
</tr>
<tr>
<td>LegacyDurationTimeZone</td>
<td>The ODBC driver returns timer and duration field values in the UTC timezone by default, starting with the 1.0.10 version. When this property is <code>true</code>, the ODBC driver returns timer and duration field values using the display value, as shown in the UI. This property can be used to preserve compatibility with legacy integrations that depend on the display value. See KB0583982 for details about this behavior.</td>
<td>false</td>
</tr>
</tbody>
</table>

If you need to use more than one of these properties in your connection, concatenate the settings with a semicolon (;) delimiter. For example, the
following string sets the URL to a specific instance and changes the batch size to 200 records.

`url=https://demo1234.service-now.com;BatchSize=200`

**ODBC management console properties**
You can access these properties from the ODBC Management Console available in the Windows Start menu at *ServiceNow ODBC > Management Console*.

<table>
<thead>
<tr>
<th>Property name</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServiceJVMOptions (Services\Service Settings\IP Parameters)</td>
<td>JVM command line properties and option. For example, to change the maximum Java heap size, modify the <code>-Xmx150m</code> parameter.</td>
<td><code>-Xms64m</code> <code>-Xmx150m</code></td>
</tr>
<tr>
<td>DataSourceIPProperties (OpenAccess SDK 6.0 Manager&lt;installation location&gt;\Services \ServiceNow_ODBC\Data Source Settings\ServiceNow \IP Parameters)</td>
<td>Global default of the instance URL for all ODBC connections. For more flexibility, you may also create new DSNs with default URL configurations.</td>
<td><code>https://demo.service-now.com</code></td>
</tr>
<tr>
<td>ServiceSQLDiskCacheMaxSize (Services\Service Settings \SQL Engine Parameters)</td>
<td>Specifies the maximum size of the disk cache files. Increase this value when you see <code>Disk Cache file size limit has reached</code> errors.</td>
<td>200</td>
</tr>
<tr>
<td>Timeout (Data Source Settings \ServiceNow\IP Parameters)</td>
<td>Specifies the socket inactivity timeout value in seconds. Increase this to a value greater than <code>glide.soap.request_processing_timeout.odbc</code> when you see <code>GetKeys failed (Socket timeout)</code> errors.</td>
<td>175</td>
</tr>
<tr>
<td>ServiceJVMLocation</td>
<td>Contains the JRE location used by the ODBC driver.</td>
<td>There is no default value for this property. You are prompted to enter the JRE location.</td>
</tr>
</tbody>
</table>
ODBC management console properties (continued)

<table>
<thead>
<tr>
<th>Property name</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>at the time of installation.</td>
</tr>
</tbody>
</table>

**Service JVM options**

You can specify these values within the ServiceJVMOptions parameter in addition to standard JVM arguments such as -Xmx.

**Service JVM options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>-DLOG_FILE_NAME</td>
<td>The location of the ODBC log file. This property is available starting with</td>
<td><code>${user.home}\AppData \Local\ServiceNow\odbc\logging\odbc.log</code></td>
</tr>
<tr>
<td></td>
<td>the ODBC driver 1.0.7.1 release.</td>
<td></td>
</tr>
<tr>
<td>-DLOG_LEVEL</td>
<td>The logging level used when writing to the ODBC log file. You can specify</td>
<td>INFO</td>
</tr>
<tr>
<td></td>
<td>the logging level using Logback levels, such as TRACE, INFO, or ERROR. This</td>
<td></td>
</tr>
<tr>
<td></td>
<td>property is available starting with the ODBC driver 1.0.8 release.</td>
<td></td>
</tr>
</tbody>
</table>

**Instance properties**

An administrator can configure these properties by adding a property or modifying an existing one in the ServiceNow instance.

**Instance properties**

<table>
<thead>
<tr>
<th>Property name</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>glide.db.max.aggregates</td>
<td>The maximum number of rows returned by aggregate functions.</td>
<td>100000</td>
</tr>
<tr>
<td>glide.db.max_view_records</td>
<td>The maximum number of rows returned by a database view.</td>
<td>10000</td>
</tr>
</tbody>
</table>
Test the ODBC driver

After configuring the ODBC driver, test that the driver can connect to the base instance as the ODBC user and can query data from a target table.

Before you begin
You must have administrator-level access for the Windows computer on which you want to test the ODBC driver.

About this task
To test the connection, run the ODBC Administrator program.

Procedure
1. In Windows, navigate to Start > Programs > ServiceNow ODBC > ODBC Administrator.
   The ServiceNow ODBC data source is installed as a system data source.
2. Select the System DSN tab, and then select the ServiceNow data source.
3. Click Configure.

4. Click Test Connect in the ODBC driver Setup dialog box.
5. Enter the login credentials.

These are the usual ServiceNow base system login credentials for the ODBC user you created.
6. Click **OK** to log in to the data source.

7. Click **OK** again when the success message appears.

**Related information**

Configure the ODBC driver

**Enable debug logging**

If you experience unexpected behavior when using the ODBC driver, you can enable debug logging and generate debug logs to help identify the issue.

**About this task**

Debug logs can be useful when submitting an incident with Customer Service and Support.

When you enable debug logging, note the version and bitness (32 bit or 64 bit) of the installed ODBC driver, the Windows operating system, and the client application you are using with the ODBC driver.

To generate debug logs, follow these steps.
Procedure

1. Close all active applications that may use the ODBC driver.
2. Navigate to one of these paths, based on your operating system.
   - For Windows 7: C:\Users\<user_name>\AppData\Local\ServiceNow\odbc\logging
   - For Windows XP and earlier: C:\Program Files\ServiceNow\ODBC\%LOCALAPPDATA\ServiceNow\odbc\logging
3. Delete any existing log data to ensure that you log only relevant information.
4. Run a query that produces the unexpected behavior, then immediately close the application and review the log files.

Related information

Configure the logging level of the ODBC driver

Test a query

To verify that the user has the appropriate permissions to send requests to the instance using ODBC, run a query using Interactive SQL.

About this task

For testing, use a query that returns exactly one record, such as a query using the **Number** value of a record.

Procedure

1. In the base system instance, navigate to **Incident > All**.
2. Record the **Number** of an incident record.
3. On the computer where the ODBC driver is installed, navigate to **Start > Programs > ServiceNow ODBC > Interactive SQL**.
4. Enter connect "odbc.user"**"password"@ServiceNow and press Enter.
5. Enter the following text, substituting the incident number you recorded.
   ```sql
   select short_description from incident where number='<incident number>';
   ```
6. Press Enter.

Results

The instance should respond with the short description of the incident record.

ODBC troubleshooting

Review these troubleshooting resources to resolve issues with the ODBC driver.
For troubleshooting information, see the Knowledge Base articles troubleshooting ODBC driver issues and troubleshooting common ODBC error messages.

**Related information**
- Getting started with ODBC
- Create an ODBC user account and assign the odbc role
- Define an ACL rule for the odbc role
- Configure the ODBC driver
- Test the ODBC driver

**ODBC behavior**
After testing the ODBC driver you can use it to query your instance database from a variety of client applications.

**ODBC aggregate functions**
The ODBC driver attempts to download data and apply aggregate functions locally. The ODBC driver supports the following aggregate functions.

- COUNT
- SUM
- MIN
- MAX
- AVG

Activate the Aggregate Web Service plugin to improve the performance of aggregate queries through the ODBC driver.

**ODBC date and time values**
The instance and the machine on which the ODBC driver is installed may use two different time zones. Date and time values returned by the ODBC driver are in the local time zone of the application using the driver, not the ServiceNow instance time zone.

Ensure that you query in accurate time zones for both the instance and the machine that hosts the ODBC driver. GlideRecord performs filtering based on the instance time zone, and the ODBC client is filtered based on the Windows time zone.
For example, an instance is in Central Standard Time (CST), and the ODBC driver is installed on a machine that is in Pacific Standard Time (PST). An incident is created on the instance at 2014-05-20 10:00:00, and the time that the incident was created is displayed in the UI as 10:00:00 for users in both time zones. However, in order to successfully query this incident by creation date and time, a user on the machine in PST must query 2014-05-20 08:00:00 instead of 2014-05-20 10:00:00.

Duration and timer type fields are returned using the UTC timezone, starting with ODBC version 1.0.10. See KB0583982 for details about this change.

**ODBC display values**

Some examples of how to use and work with ODBC display values are shown below.

- **Display values in Choice and Reference columns:**

  When querying a column of type Choice, Reference, Duration, or Timer, an additional column with the prefix `dv_` is available that contains the display value. For example, you can select `dv_caller_id` to return the `sys_user.name` display value of the reference field from an incident record without making another request to the sys_user table.

  ```sql
  ISQL> select number, dv_caller_id, caller_id from incident;
  number  dv_caller_id  caller_id
  INC00000009  Rick Berzle  5137153cc611227c000b0d1bdcd00d006
  INC00000010  Fred Luddy  5137153cc611227c000b0d1bdcd00d006
  INC00000011  Don Goodliffe 9e31b2d1d0611227d000b0d1bdcd00d006
  INC00000012  Don Goodliffe 9e31b2d1d0611227d000b0d1bdcd00d006
  INC00000013  Joe Employee 681c1c960a8016400b98a06818d57c7
  INC00000014  Bow Ruggeri f2982d2d2c611227d0105c6be7f1548c8
  INC00000015  Fred Luddy  5137153cc611227c000b0d1bdcd00d006
  INC00000016  Bow Ruggeri f2982d2d2c611227d0105c6be7f1548c8
  INC00000017  Joe Employee 681c1c960a8016400b98a06818d57c7
  INC00000018  Taylor Uveland 46b3a1e3d0a9e1f9810005f2994799b8f8
  INC00000019  Fred Luddy  5137153cc611227c000b0d1bdcd00d006
  ```

- **Display values in filter conditions:**

  Display values can also be used in a filter condition. The ODBC driver optimizes the query condition and processes the filter on the server, for example, querying on the display value of `sys_user` for the `caller_id` field of an incident by using the `dv_caller_id` field name.

  ```sql
  ISQL> select number, dv_caller_id, caller_id from incident where dv_caller_id = 'Fred Luddy';
  number  dv_caller_id  caller_id
  INC00000010  Fred Luddy  5137153cc611227c000b0d1bdcd00d006
  ```

- **Display values in aggregate queries:**

© 2021 ServiceNow, Inc. All rights reserved.
ServiceNow, the ServiceNow logo, Now, and other ServiceNow marks are trademarks and/or registered trademarks of ServiceNow, Inc., in the United States and/or other countries. Other company names, product names, and logos may be trademarks of the respective companies with which they are associated.
Aggregate queries can also take advantage of display values if you specify them in the group by or where clause, for example, grouping on the `caller_id` field of an incident, as well as specifying a filter for it. The query is optimized by passing through to the server.

**Display values in aggregate queries**

![SQL Query Example]

```sql
SELECT COUNT(*) AS dv_caller_id FROM incident WHERE dv_caller_id IS NOT NULL GROUP BY dv_caller_id;
```

**Querying table and column names**

You can get a list of accessible tables and columns based on the read ACLs for the querying user.

- The following query will return the names of all tables for which the querying user has read access:

  ```sql
  SELECT * FROM oa_tables;
  ```

- After you know the name of the table you want to query, you can query the names of all columns for which the user has read access. The querying user must have read access for both the table and the columns.

  ```sql
  SELECT * FROM oa_columns WHERE table_name = 'table_name';
  ```

**Note:** The `oa_tables` and `oa_columns` tables are internal ODBC tables. These tables are accessible only via the ODBC driver.

**Increase the field length in SQL queries**

The ODBC driver limits the field length in SQL queries to the maximum length defined by the ServiceNow dictionary entry. You can increase the maximum field length to avoid truncating data.

**Before you begin**

Role required: admin
About this task
If the data coming from the ODBC source exceeds the field size of the dictionary entry, ServiceNow truncates the query output to fit the field size. If your data is truncated, you can do the following.

Procedure
1. Increase the maximum length in the dictionary entry for the field in question.
2. Reconnect the ODBC driver to pick up the change.

Note: By default, the ODBC driver uses the VARCHAR data type to store query string output. When strings become very large (roughly 16000 characters), the ODBC driver uses the LONGVARCHAR data type instead. It is important to keep in mind, however, that the LONGVARCHAR data type has a more limited set of SQL commands that can be executed on it. For example, it does not support queries using scalar data.

Related information
Modify dictionary entries
Use Interactive SQL with ODBC

ODBC and client applications
See the following pages for examples of how to use the ODBC driver to create data sources from other applications.

Use Interactive SQL with ODBC
Run the Interactive SQL application for quick verification of connectivity and to test query results without using a full application.

Procedure
1. In Windows, navigate to Start > Programs > ServiceNow ODBC > Interactive SQL (ODBC).
2. Enter the following command to connect to the base instance. Select the appropriate user credentials in the format: ID*password@DSNName. The password cannot contain special characters.

CONNECT odbcuser*password@ServiceNow
3. Issue a `SELECT` SQL command, such as:

```sql
SELECT NUMBER, short_description FROM incident;
```

Make sure to include the semicolon at the end of your query statement. You will be presented with a `Cont>` prompt otherwise.

Specify the maximum number of rows returned

By default, ServiceNow only returns 100 rows of data with each iSQL query. If you need to return more rows of data, set the `maxrows` parameter for the iSQL session.

To return all rows set `maxrows` to zero:

```
maxrows 0
```

To return more than 100 rows set `maxrows` to a higher value. For example, to return 500 rows:

```
maxrows 500
```
Note: If running the Interactive SQL console from a shortcut, you must modify the shortcut Target to include the `-maxrows` parameter with the desired value.

SQL support
The ODBC driver embeds a third party SQL/ODBC engine from DataDirect, a division of Progress Software.

See the DataDirect SQL Reference for information on proper SQL syntax.

Note: The ServiceNow ODBC driver only supports `SELECT` statements. The driver ignores other SQL statements such as `CREATE` and `ALTER`.

ODBC driver in SQL Server
Use the ServiceNow ODBC driver in SQL Server as a Linked Server.
Using the ODBC driver in SQL Server as a Linked Server allows SQL Server to query tables from a ServiceNow instance directly via the ODBC driver. Only use the procedures described with SQL Server 2008 and 2012. Other versions of SQL Server may cause unexpected behavior. If you encounter unexpected behavior, refer to the troubleshooting linked server Knowledge Base article.

**Required Permissions**
Additional information on the required permissions for SQL Server Linked Servers can be found on the MSDN blog.

⚠️ **Note:** Review this information if you encounter permission errors with SQL Server.

**ODBC SQL Server video tutorials**
Watch video tutorials about configuring and troubleshooting the ODBC driver with a SQL Linked Server.

**Configuring Microsoft SQL Linked Server with the ODBC driver**
Configuring Microsoft SQL Linked Server with the ODBC driver

**Troubleshooting Microsoft SQL Linked Server permissions**
Troubleshooting Microsoft SQL Linked Server permissions

**Configure SQL Server**
The following example configuration was performed on SQL Server 2008, installed on Windows Server 2008.

**Before you begin**
The ODBC driver must be installed on the same computer on which SQL Server is installed.

**Procedure**
1. Right-click the SQL Server Management Studio application and select Run as Administrator.
2. Log in to the database to which you want to link.
3. Right-click Server Objects > Linked Servers.
4. Click New Linked Server.
5. Enter the following values in the dialog.
   - Linked server: SERVICENOW. This is the name of the Linked Server.
   - Provider: Microsoft OLE DB Provider for ODBC drivers
• Product name: ServiceNow. This is an identifier. Enter any value that is appropriate.

• Data source: ServiceNow. This is the name of your DSN.

6. Select **Security** from the Select a page list, and then enter the following security values:

a. For a login connection, select **Be made using this security context**.

b. Enter the user name and password for connecting to the ServiceNow instance.

c. Click **OK**.
7. Navigate to Server Objects > Linked Server > Providers and double-click Microsoft OLE DB Provider for ODBC drivers.

8. Select the following options.

- Nested Queries
- Level zero only
- Support 'Like' operator

**Tip:**

ServiceNow recommends running the third-party provider in the out-of-process mode setting (AllowInProcess=FALSE). If you run the provider in-process (within the same process as SQL Server), then any issues with the provider can affect the SQL Server process, which in turn could result in crashing SQL server.
9. Test your connection by selecting the newly created linked server SERVICENOW and selecting **Test connection**.
10. Execute the following query in a query builder window to retrieve some results.

```sql
SELECT * FROM OPENQUERY (SERVICENOW, 'select Cast(sys_mod_count as Decimal(38,0)), number, short_description from incident') 
```

### What to do next

**Number Precision Errors**

You may encounter precision errors querying for decimal or number field values using the `OPENQUERY` syntax with the ODBC driver. In this case, use the `Cast` syntax to convert the precision. For example:

```sql
SELECT * FROM OPENQUERY (SERVICENOW, 'select Cast(sys_mod_count as Decimal(38,0)), number, short_description from incident')
```

### SQL Server Connection String

To use the ODBC driver directly in SQL Server 2008, specify the connection string in the following format.

```
Dsn=ServiceNow;uid=username;pwd=password
```

**Note:** The latest SQL Server 2008 patches are required for the ability to specify a connection string in the user interface, via the SQL import wizard.

### Using sp_addlinkedserver

The following example creates a linked server named "ServiceNow ODBC" that uses the Microsoft OLE DB Provider for ODBC (MSDASQL) and the data_source parameter.
EXEC sp_addlinkedserver
    @server = N 'ServiceNow ODBC' ,
    @srvproduct = N '' ,
    @provider = N 'MSDASQL' ,
    @datasrc = N 'ServiceNow';
GO

After creating the linked server, you must update its properties to specify the login credentials.

**Use the ODBC driver in Excel**

After installing the ODBC driver and its associated DSN, use it in Excel as a data source provider.

**Procedure**

1. In Excel open the **Data** tab.
2. Under **From Other Sources** open **From Microsoft Query**.
3. Select **ServiceNow** as your database (the default DSN name).
4. Clear the **Use the Query Wizard to create/edit queries** check box.
5. Supply the ServiceNow user name and password.

6. Select a table from the ServiceNow instance and click Add.
7. Close the dialog box.

8. Select the table columns from which the Query Builder will retrieve data. Use the list above the table, or type the names directly into the columns, and then press Enter.

9. To retrieve the data and create the Excel record, click the Return Data icon or select File > Return Data to Microsoft Office Excel.
The requested data is brought into Excel.

Use the ODBC driver in Crystal Reports

After installing the ODBC driver and its associated DSN, use it in Crystal Reports as
da data source provider.

About this task

>Note: Crystal Reports includes the configuration file CRConfig.xml that
contains the JVM minimum heap size (Xms) and maximum heap size (Xmx)
values. When configuring the ODBC driver with Crystal Reports, ensure that
the ODBC driver uses the same minimum and maximum JVM heap size
as Crystal Reports. If these values do not match, update the ODBC driver
design settings, not the Crystal Reports settings.
Procedure


2. Create a new connection using the ServiceNow DSN.
3. Select a table from the list of available tables.

4. Select the available fields from the selected table.
5. Click **Finish** to render the report.

Domain separation and ODBC driver

This is an overview of domain separation and ODBC drivers. Domain separation enables you to separate data, processes, and administrative tasks into logical groupings called domains. You can control several aspects of this separation, including which users can see and access data.

**Support level: Basic**

- There is business logic to ensure data goes into the proper domain for the application’s service provider use cases.
- In the application, the user interface, cache keys, reporting, rollups, aggregations, and so on, all consider domain at run time.
- The owner of the instance needs to be able to set up the application to function normally across multiple tenants.

**Use case:** As a service provider when I use chat to respond to a tenant-customer’s message, the client must be able to see my response.

**Related information**

Domain separation for service providers
Domain separation and Web Services

Domain separation is supported in Web Services. Domain separation enables you to separate data, processes, and administrative tasks into logical groupings called domains. You can control several aspects of this separation, including which users can see and access data.

Support level: Standard*

The support level is Standard but has some exceptions or special conditions.

- Includes Basic level
- Business logic: The service provider (SP) creates or modifies processes per customer. The use cases reflect proper use of the application by multiple SP customers in a single instance.
- The instance owner must be able to configure minimum viable product (MVP) business logic and data parameters. This configuration is done per tenant, as expected for the specific application.

Sample use case: An admin must be able to make comments required when a record closes for one tenant, but not for another.

For more information on support levels, see Application support for domain separation.

Related information

Domain separation for service providers

Developer guides

This section contains developer guides that help software developers to integrate their applications with the ServiceNow platform.

Each developer guide addresses a single ServiceNow product such as Financial Services Operations.

Available ServiceNow developer guides

<table>
<thead>
<tr>
<th>Developer guides</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Financial Services Operations</td>
<td>ServiceNow® Financial Services Operations (FSO) enables financial services institutions (FSIs) to securely automate core operational processes through pre-built digital applications and workflows. FSO connects operations across all departments—front,</td>
</tr>
<tr>
<td>developer guide</td>
<td></td>
</tr>
</tbody>
</table>
## Available ServiceNow developer guides (continued)

<table>
<thead>
<tr>
<th>Developer guides</th>
<th>Description</th>
</tr>
</thead>
</table>
|                  | middle, and back-office—for financial institutions with a single platform.  
This developer guide provides information on how to extend the FSO base capabilities as well as integrate your existing applications with ServiceNow FSO. |
Index

Special Characters
.net
7457
A
AbstractDBObject
isValid() 1104
AbstractDBObject API 1104
AbstractReconciler
getReconciliationField() 1106
getReconciliationKey() 1105
hasChanged() 1106
markAbsent() 1107
markPresent() 1108
process() 1107
readDatabaseFields() 1108
readDiscovered() 1108
resolveReferenceFields() 1109
setDatabaseFields() 1110
setup() 1110
track(String sys_id) 1109
AbstractReconciler API 1105
AbstractScriptProcessor
initialize(String request, String response, String process) 1111
process() 1111
AbstractScriptProcessor API 1111
AbstractTimelineSchedulePage
add() 1113
addSeparator() 1113
elememntMoveX() 1114
elementMoveXY() 1115
elementMoveY() 1114
elementSuccessor() 1115
elementTimeAdjustEnd() 1116
elementTimeAdjustStart() 1116
getitems() 1117
inputBox() 1117
itemMove() 1118
setDoReRenderTimeline() 1118
pageTitle() 1119
setStatusError() 1119
setStatusPrompt() 1119
setStatusSuccess() 1121
access enforcing 72
to Update Set picker 451
workflow scratchpad
from business rules
4541
access control rules
ACL rules
71, 71
ActivityHistoryRecord
1152
addArrivedActivityToJoin()
1152
addJoinFromActivityIds()
1153
addTransition()
1153
dump()
1154
doesJoinContainActivity()
1154
gLogString()
1154
gSatisfiedJoinActivities()
1155
gTransitionCount()
1155
gUnSatisfiedJoinActivities()
1156
isARollback()
1156
isADestination()
1156
isJoin()
1157
isJoinSatisfied()
1157
isJoinWaitingForActivity()
1158
isRolledBack()
1158
isTurnstile()
1159
add role to every user
4545
add test step
535
adding
1172
Aggregate API
5160
AJAX
4518
AJAXClientTiming
4523
AngularJS service
4021
APCHandler
classifyAndIdentify()
1175
APCHandler API
1175
ApcPduOutletReconciler
getReconciliationField()
1176
getReconciliationKey()
1176
hasChanged()
1176
readDatabaseFields()
1177
readDiscovered()
1177
setDatabaseFields()
1178
ApcPduOutletReconciler API
1176
API
3384
APIs
Glide Server
4404
app install
6146
app rollback
6146
application
import from source control
348
link to source control
351
application administration
71, 71

Application Administration
ACL Inheritance
sysScoped_admin_acl_inheritance
71
application development
117
application files
363, 366, 367
adding
317
searching
by name
324
by record values
325
by type
324
application scope
11
application service API
5210
applications
9, 10, 11, 14, 14, 15, 15, 15, 25,
26, 28, 29, 31, 55, 57, 63, 64,
77, 81, 363, 366, 367
access settings
26, 28
design time
38, 38, 39
denying
40
granting
41
runtime
33, 34
default permissions
33
denying
35
granting
36
application administration
65
application files
form
protecting
relationships
viewing
application scope
configure layout
creating lists and forms
form design
namespace identifier
creation options
create a custom application
start from a template
start from an existing service
start from global
start from scratch
custom application record
deleting
developers
development environment
fix scripts
running
testing
include data
install
install updates
managing
moving
parts of
publish
publishing
removing
repository
selecting
sharing
studio
uninstall
update set
updating
approval ui macro
approve execution plan
use script
arbitrary JavaScript
ArrayUtil
concat(Array parent, Array child)
contains(Array array, Object element)
  1207
convertArray(Object a)
  1208
diff(Array a, Array b)
  1208
ensureArray(Object object)
  1209
indexOf(Array array, Object element, Number startIndex)
  1210
indexOf(Array array, Object element)
  1210
intersect(Array a, Array b)
  1211
union(Array a, Array b)
  1212
unique(Array a)
  1212
ArrayUtil API
  1206
AssessmentCreation
  createAssessments()
  1213
AssessmentCreation API
  1213
ATF for Service Catalog in Service Portal
  783, 784, 786, 787, 788, 789, 790, 792, 794, 798, 800, 803, 880, 893
Attachment API
  5244
AuthCredential
  1216
authentication portal
  create
    asp.net
    4565
Authorization credentials
  3837
autofill
  4580
automated test administration
  614
automated test framework
Automated Test Framework
  492, 533, 534, 535, 538, 592, 595, 637, 641, 646, 648, 826, 828, 879, 880, 893, 894, 895
  Add Item to Shopping Cart
    798, 838
  Add row to multi-row variable set
    786
  admin properties
    924
  administering REST
    625
  assert json response payload element
    867, 869
  assert JSON response payload element
    870
  assert response header
    864
  assert response JSON payload
    865
  assert response time
    863
  assert response XML payload
    866
  assert status code
    860
  assert status code name
    862
  assert value
auto flush
944
basic authentication profiles
625
category
943
click modal
824
execute server script
888
field validation
821
filter rest headers
626
Open a Catalog Item
784, 831
Open a Record Producer
783, 832
open existing form
816
open new form
815
Order Catalog Item
800, 839
record delete
886
record insert
883
record query
882
record update
884
record validation
887
replay request item
890
REST
577
REST API Explorer
856
REST properties
626
retention policy
944
Save current row of multi-row variable set
787
Search for a Catalog Item
880
send REST
856
send REST Request - Inbound
858
Set Catalog Item Quantity
794, 833
set value
817
Set Variable Values
788, 832
step categories
943
step config
929
Submit a Record Producer
803, 840
table cleanup
944
test administration
924
test client runner
917, 918
test queue
919, 920
test record form
897
test results
898, 899
test suite record
898
test suite results
908, 915, 916
test template
943
UI visibility
822
Validate Price and Recurring Price
792, 837
Validate Variable Values
789, 834
Variable State Validation
790, 835
Automated Test Framework>
891
Automated Test log step>
891
automated test step order
537, 930
automated test suite
596, 601, 602, 603
AutomationException
1219
getMessage() 1219

B
back reference
595, 635
background scripts
4452
baseline version
431
batch cost plans
1549
breakpoints
remove
4611
set
4611
business
4414
Business rule
637
Business Rule and Workflow
order
4381
Business Rule order
4381
business rules
4414
creating
4419
how they work
4414
scoped applications
4417
BusinessServiceManager
1219
addCI(String service_id,
String source_ci, String
target_ci)
1219
addManualConnection(String
target_id, Object
manualEndpoint, String
bs_id)
1220
BusinessServiceManager()
1221
migrateManualToApplicationService(String
sys_id)
1221
populateApplicationService(String
sys_id, Number levels, String
blackListedRelationTypes)
1222
removeManualCi(String
targetSysId, String bs_id)
1222
removeManualConnection(String
ci_sys_id, String
endpoint_sys_id, String
bs_id)
1223
C
c sharp integration
7457
Cabrillo
1229
isNative()
1229
cabrillo button styles
1224
cabrillo close button styles
1224
cabrillo constants
1224
cabrillo data types
1225
CatalogItemVariableSet
1281
create(Boolean standardUpdate)
1281
deleteRecord(Boolean standardUpdate)
1282
read(Object columns, Boolean standardUpdate)
1283
setAttributes(Object attributes)
1283
update(Object columnValues, Boolean standardUpdate)
1284
CatalogItemVariableSetM2M
1284
create(Boolean standardUpdate)
1284
deleteRecord(Boolean standardUpdate)
1285
read(Object columns, Boolean standardUpdate)
1285
setAttributes(Object attributes)
1286
update(Object columnValues, Boolean standardUpdate)
1286
CatalogJS
1261
canView(boolean mobile)
1261
Catalog(Glide Record now_GR)
1261
Catalog(String sys_id)
1261
getAvailableCatalog()
setTableName(String tableName)
update(Object columnValues, Boolean standardUpdate)

Category 7447, 7448

CatItem API 1301
create(Boolean standardUpdate)
deleteRecord(Boolean standardUpdate)
getFirstAccessibleCategoryForSearch(String catalogId)
getRecordClass()
isVisibleServicePortal()
setAttributes(Object attributes)
setCatalogs(String catalogs)
setCategories(String categories)
setImage(String dbImageSysId, String type)
setTableName(String tableName)

update(Object columnValues, Boolean standardUpdate)
CertificateEncryption CertificateEncryption()
generateMac(key, algorithm, data)
getThumbPrint(String certificateID, String algorithm)
getThumbPrintFromKeystore(String certificateID, String alias, String algorithm)
sign(String certificateID, String alias, String aliaspassword, String algorithm, String datatosign)
certificates encryption APIs 1974, 1975
Change Management API 5632
ChangeCollisionHelper
addCiToChgAffectedCis(String ci, String changeld)
getAffectedCisByChangeld(String changeld)
getBlackoutsByDate(GlideDateTime startDate, GlideDateTime endDate)
getChangesWithAffectedCi(String ci, GlideDateTime startDate, GlideDateTime endDate)
j
getChangesWithCi(String ci, GlideDateTime startDate, GlideDateTime endDate)
getCiMaintenanceSchedule(String ci)
getCiMaintenanceSchedule(String ci)
getDependants(String ci)
getDependants(String ci)
getDependencies(String ci)
isCiInAffectedCis(String ci, String changeld)
isDateInCiMaintenanceWindows(GlideDateTime startDate, GlideDateTime endDate, String maintenanceWindow)

ChangeCollisionHelper API
ChangeConflict
ChangeConflict(String cId, String changeld, String type)
toString()

ChangeConflict API
ChangeConflictHandler
addChangeConflict(String chgConflict)
ChangeConflictHandler()
deleteConflictsByChangeld(String changeld)
getConflicts()
getConflicts()

ChangeConflictHandler API
Checkout Shopping Cart
CIData API
1347
CIIdentification
CIIdentification(String ciData, String logger)
1353
d debug(String msg)
1354
process()
1354
CIIdentification API
1353
CIIdentifierResult
CIIdentifierResult(Array matched, Boolean matchable, Boolean terminatedChain)
1354
CIIdentifierResult API
1354
CimCIData
addSerial(String type, String serial)
1355
SerialRecord(String type, String serial)
1355
setMakeAndModel(String make, String model)
1356
CimCIData API
1355
CimIDSensor
configureTriggeredProbe(String probe)
1356
getInstanceHashToken()
1357
prepare()
1357
process(Object results, CimCIData cimData, CimIDSensor sensor)
1357
runMultiProbeScript(String script, Object probeResult, String probeRecord)
1358
updateDeviceCount()
1358
CimIDSensor API
1356
CimInstanceToken
gerHashToken(String cimomip)
1359
parse(XMLObj instance)
1359
CimInstanceToken API
1359
CimProbe
gerNamedQueries()
1360
gerQueries()
1360
gerQueryRecords()
1360
CimProbe API
1360
CimProbeResult
gerNamedInstances(String namedQueries)
1361
gerQueries()
1361
CimProbeResult API
1361
CIRelatedList
addRec(String record)
1362
addRecs(String records)
1362
appendXMLChildFld(String parent, String name, String datum)
1363
appendXMLChildRecord(String parent, String name, String record)
CIRelatedList(String table, String refField, String cmdb_ci, String debugFlag)
CIRelatedList API

ClientUtils
servicesAffectedByCI(String CI_sys_id)
servicesAffectedByTask(GlideRecord task)
ClientUtils API

ClassifierProbes
launch()
launchSupplementary()
size()
ClassifierProbes API

ClassName
methodName()
getEncodedQuery()
getLimit()
getTableName()
hasNext()
insert(Function responseFunction)
next()
orderBy(String column)
query(Function responseFunction)
setLimit(Number maxQuery)
Clientside GlideRecord API
Clientside GlideRecordV3 API
addOrderBy(String column)
addQuery(String fieldName, Object operator, Object value)
addQuery(String fieldName, Object value)
deleteRecord(Function responseFunction)
get(String sysId)
getEncodedQuery()
getTableName()
GlideRecord(String tableName)
hasNext()
insert(Function responseFunction)
next()
orderBy(String column)
query()
Clientside GlideRecordV3 API
close form
CMDB ingest
CMDB instance API
CMDB meta API
CMDB Query Builder
CMDBDuplicateTaskUtils
createDuplicateTask(String sysIDs)
CMDBGroupAPI
getAllCI(String groupld, Boolean requireCompleteSet)
getAllCIFromQueryBuilder(String groupld, Boolean requireCompleteSet)
getManualCIList(String groupld, Boolean requireCompleteSet)
getSavedQueryIdList(String groupId, Boolean requireCompleteSet)
    1479
setManualCIList(String groupId, String ciSysIds)
    1481
setSavedQueryIdList(String groupId, String queryIds)
    1482
CMDBQueryBuilderAPI
    1484
getSavedQueryExecutionDetails(String savedQueryName, Boolean executeQuery)
    1485
CMDBTransformUtil
    1486
    CMDBTransformUtil()
    1487
setError()
    1487
getOutputPayload()
    1488
getOutputRecordSysId()
    1499
hasError()
    1525
identifyAndReconcile (Object source, Object map, Object log)
    1526
logTransformStats( Object log)
    1528
CMDBTransformUtil -. . .
    setDataSource(String source)
    1529
CMDBUtil
    baselineProposedChangesApplyChanges(GlideRecord proposedChanges)
    1530
    bootstrap(String dictFile)
    1531
createCIRelationship(String tableName, String parentField, String childField, String parentDesc, String childDesc)
    1532
getAllChildrenOfAsCommaList(String baseTable)
    1532
getCMDBViews()
    1533
getTables(String tableName)
    1533
isExcludedFromBR(String className)
    1533
processCIChange(GlideRecord event, GlideRecord target)
    1534
processRelChange(GlideRecord event, GlideRecord current, String relType, String triplet)
    1535
removeCIRelationship(String tableName, String parentField, String childField, String parentDesc, Object childDesc)
    1535
reParentTable(String table, String oldParent, String newParent)
    1536
CMDBUtil API
    baselineProposedChangesGenDiff(GlideRecord current, String changeSet)
424 collisions
limitations while resolving
413 resolving
412 com.glide.scope.access.restricted_caller
44 comma separated value
parser
1554 Compare to Current
435 Compatibility Mode
4595 Configuration management
data ingest
5992 conflict
resolving
412 conflicts
367 ConnectActionResponse
newRecord(String table,
Object values, String view)
1540 ConnectActionResponse API
1539 ConnectionInfo
1540
getAttribute(name)
1541
getCredentialAttribute()
1542
dataMap()
1542
getExtendedAttributes()
1543
ConnectionInfoProvider
1543, 4043 ConnectionInfoProvider()
1544
getConnectionInfo()
1545
getConnectionInfoByDomain(String
aliasID, String domainID)
1545 console
4606 contextual development
environment
25, 64 application list
55 application picker
57 application version
9 context
26 warning messages
63 Conversation
1546 create(String name)
1546 sendMessage(String body,
String field)
1548 copy automated test step
538 CostPlanBatchOperations
1549
add(Object costPlan)
1549
clear()
1551, 1552
create automated test
534 create automated test suite
598 Create cost plans
1549 Create Import Set tables
2293 CreateNow Studio
316 keyboard shortcuts
createOrUpdateService
POST
5225
Creating transform entries
2303
Creating transform maps
2303
CriteriaEvaluator
  addCriterion(String lhComparand, String operator, String rhComparand)
    1553
evaluate(Object lhValues, String rhValues, Boolean any)
    1554
CriteriaEvaluator API
  1553
cross-scope access
    43, 45, 46
CSM API Case API
  5302
CSVParser
  1554
    parseLineToArray(String cvsLine, String delimiter, String quoteCharacter)
      1554
    parseLineToObject(String cvsLine, Array headers, String delimiter, String quoteCharacter)
      1556
currency configuration
  1980
Currency Value
  2008
currency values
  1996
Currency2
  2177
custom
  4550
custom category
  618

Custom Chart Data
Generation
  4092
custom processor
    create
      4451
custom scripted step config
  894
custom step
  615
custom step config
  616
CustomEvent
  1624
    fireAll(String event, String qualifier)
      1624
customizations
    load from XML
      468

Data
  1624
Data Classification REST API
  6319
Data ingest
  5992
data policy
  641
Date
  validate
    4585
DateTimeUtils
  int8ToGlideDateTime(Number int64)
    1644
  msToGlideDateTime(Number milliseconds)
    1648
DateTimeUtils API
  1647
debug
  4597, 4619
debug logs
  4622
debug script
4638
debugger
4597
   User interface
   4602
debugging
4979
   session
   4620
   UI policies
   4633
debugging classifications
4627
decision answers
1648
decision input
1648
DecisionTableAPI
1648
   getDecision(String
decisionID, Map inputs)
   1658
   getDecisions(String
decisionID, Map inputs)
   1659
   getInputs(String decisionID)
   1665
   getQuestions(String
decisionID)
   1667
Deep Link
3359
Deep Linking
3359
Deep-linking
3359
delegated development
78
dependencies
15
design access
31
developer permissions
78
developers
81
       group
       77
       user
       77
DevOps API
6330
DHTMLX
3596
Discovery
   script includes
   using
     4436, 4438
DSCScriptableAPI
1708
   chkDataInvalidDomainForTables()
   1708
   chkDataInvalidPathForTables()
   1708
   DSCScriptableAPI()
   1709
DSN
   global default
   7560
Duration Field Value
   setting
   4412
DurationCalculator
1709
   DurationCalculator()
   1714
   relative duration
   4458
   simple duration
   4456
DurationCalendar
   calcDuration()
   1709
   calcRelativeDueDate()
   1710
   calcRelativeDuration()
   1711
   calcScheduleDuration()
   1712, 1713
getEndDateTime() 1715
getSeconds() 1715
getTotalSeconds() 1716
isAfter() 1717
setSchedule() 1718
setStartDateTime() 1719, 1719
setTimeZone() 1720
DynamicTranslation 1753
getDetectedLanguage(String
text, Object parms) 1721, 1753
getTranslation(String
textToTranslate, Object
parms) 1733, 1764
isEnabled(Object parms) 1750
isEnabled(Object translator) 1781
DynamicTranslation Client 1720

E
edit test template 620
elements
adding
scrolling 7486
scrolling
adding 7486
email 3301
Email API 6444
enable tests 621
enforce subscriptions 72
ES5 standards mode 4596
ES5 Standards Mode 4595
escaping types
jelly 4530
Event 1813
every example
incident
priority 1 7487
priority 1
incident 7487
Examples
end to end tutorial
7454, 7454, 7455, 7457,
7457, 7457, 7459, 7459
web services c sharp .net
7454, 7454, 7455, 7457,
7457, 7457, 7459, 7459
Excel, XLS, XSD, XLSX 7463
execution plan
script 4470
execution plan scripts
limitations 4470
write 4470
ExpenseAllocation 1819
createAllocation() 1819
initialize() 1819
ExpenseLine 1820
() 1821
createExpense()
1820
methodName()
1823
processCIParents()
1821
setCostSource()
1822
setDescription()
1822
setRecurring()
1822
setSummaryType()
1823
extensions to Jelly syntax
4531, 4532, 4532, 4532, 4533,
4533, 4535, 4536, 4536, 4537,
4537, 4537, 4538, 4538, 4538,
4539, 4539, 4539, 4540
ExtractTermsFromAttachment
API
1824
F
fetch XML
use code
7462
field messages
display
4554
field types
scripting
4382
field value
null
4425
field values
business rules and client
scripts for controlling
4423
Field Watcher
4628, 4629, 4630, 4630, 4630,
4631
FlowAPI
1828
executeActionQuick(String
name, Map inputs, Number
timeout)
1831
executeFlowQuick(String
name, Map inputs, Number
timeout)
1836
startActionQuick(String
name, Map inputs)
1845
startFlowQuick(String name, Map inputs)
1848
startSubflowQuick(String
name, Map inputs)
1851
formatting currency
2000
FormInfoHeader
1827
addMessage(String
message)
1827
free-form script
4619
freeform script
4619
fulfillment tables
24, 25, 72
Fullcalendar
3596
FX Currency
2177
G
g_aw
1885
g_form
2233, 2273
addDecoration()
2273
getLabel()
2274
hasField()
2274
removeDecoration()
2275
setLabel()
2276
General Scripting
execution order of scripts
and engines
4381
glide class overview
4377
glide stack
4379
GlideScriptable object
name
4646
GlideScriptable object
replacement list
4647
Packages Call Removal tool
4641
script functions
4371
generate choice list
AJAX
4522
GenericUPSAlarmsReconciler
getReconciliationField()
1859
getReconciliationKey()
1860
hasChanged()
1860
readDatabaseFields()
1860
readDiscovered()
1861
setDatabaseFields()
1861
GenericUPSAlarmsReconciler
API
1859
GenericUPSReconciler
getReconciliationField()
1862
getReconciliationKey()
1862
hasChanged()
1862
readDatabaseFields()
1863
readDiscovered()
1863
setDatabaseFields()
1864
GenericUPSReconciler
API
1862
GenericUPSReconciler
getReconciliationField()
1864
getReconciliationKey()
1864
hasChanged()
1865
readDatabaseFields()
1865
readDiscovered()
1866
setDatabaseFields()
1866
GenericUPSReconciler
API
1866
GenericUPSReconciler
getReconciliationField()
1865
getReconciliationKey()
1866
hasChanged()
1867
readDatabaseFields()
1868
readDiscovered()
1868
setDatabaseFields()
1868
GenericUPSReconciler
API
1866
Get authorization credentials
GetElement
  getBooleanAttribute(String attributeName)
    2138, 2165

Getting started with REST
  access the REST API Explorer
    4964
  create an incident record
    4966
  delete the incident
    4971
  read the inserted incident
    4968
  retrieve existing incidents
    4965
  update the incident
    4969

Glide currency
  1980

Glide import
  2288, 2292, 2293, 2296, 2827

Glide Server APIs
  GlideAggregate
    4404, 4405

GlideAgentWorkspace
  1885
    openRecord(String table, String sysId, Object params)
      1885

GlideAgentWorkspace
  (g_aw)
    1885

GlideAggregate
  1887
    addAggregate(String agg, String name)
      1888
    addEncodedQuery(String query)
      1889
    addHaving(String name, String operator, String value)
      1890
    addTrend(String fieldName, String timeInterval)
      1891
    getAggregate(String agg, String name)
      1893
    getQuery()
      1894
    getTotal(String agg, String name)
      1895
    getValue(String name)
      1896
    groupBy(String name)
      1897
    orderBy(String name)
      1898
    orderByAggregate(String agg, String name)
      1899
    query()
      1900
    setGroup(Boolean b)
      1901

GlideAjax
  1923, 4518
    addParam()
      1924
    example asynchronous
      4519
    example synchronous
      4522
    getAnswer()
      1924
    getXML()
      1925
    getXMLAnswer(Function callback)
      1926
    getXMLWait()
      1927
    GlideAjax()
      1928
    GlideAjaxV3
addParam(String name, String value)
    1928
getJSON(Function callback)
    1929
getParam(String name)
    1929
getParams()
    1930
getProcessor()
    1930
getURL()
    1930
getXML(Function callback)
    1931
getXMLAnswer(Function callback)
    1931
GlideAjax(String processor, String targetURL)
    1932
GlideAjax(String processor)
    1932
setErrorCallback(Function callback)
    1933
setProcessor(String serverScript)
    1933
GlideAjaxV3 API
    1928
GlideappCalculationHelper API
    1933
GlideCalendarDateTime
    1935
    add(GlideTime time)
    1935
    add(Number milliseconds)
    1936
    addDaysLocalTime(Number days)
    1936
    addDaysUTC(Number days)
    1937
    addMonthsLocalTime(Number months)
    1938
    addMonthsUTC(Number months)
    1938
    addSeconds(Number seconds)
    1939
    addWeeksLocalTime(Number weeks)
    1939
    addWeeksUTC(Number weeks)
    1940
    addYearsLocalTime(Number years)
    1941
    addYearsUTC(Number years)
    1941
    compareTo(GlideCalendarDateTime dateTime)
    1944
    equals(Object GCDT)
    1944
    getDate()
    1945
    getDayOfMonthLocalTime()
    1945
    getDayOfMonthUTC()
    1946
    getDayOfWeekLocalTime()
    1947
    getDayOfWeekUTC()
    1947
    getDaysInMonthLocalTime()
    1948
    getDaysInMonthUTC()
    1948
    getDisplayValue()
    1949
    getDisplayValueInternal()
    1949
    getDSTOffset()
getErrorMsg()
getInternalFormattedLocalTime()
getLocalDate()
getMonthLocalTime()
getMonthUTC()
getNumericValue()
getTime()
getTZOffset()
getUserFormattedLocalTime()
getValue()
getWeekOfYearLocalTime()
getWeekOfYearUTC()
getYearLocalTime()
getYearUTC()
GlideCalendarDateTime()
GlideCalendarDateTime(GlideCalendarDateTime gcDT)
GlideCalendarDateTime(String dateTime)
hasDate()
isDST()
isValid()
setDayOfMonthLocalTime(Number day)
setDayOfMonthUTC(Number day)
setDisplayValue(String asDisplayed)
setDisplayValue(String dateTime, String format)
setDisplayValueInternal(String dateTime)
setGlideCalendarDateTime(GlideCalendarDateTime gcDT)
setMonthLocalTime(Number month)
setMonthUTC(Number month)
setValue(Object dateTime)
setValueUTC(String dateTime, String format)
setYearLocalTime(Number year)
setYearUTC(Number year)
subtract(GlideCalendarDateTime start, GlideCalendarDateTime end)
subtract(GlideTime time)
subtract(Number milliseconds)
toString()
GlideCertificateEncryption
1325, 1974
    generateMac(key, algorithm, data)
    GlideCertificateEncryption()
    GlideCertificateEncryption()

GlideConversation
description
    GlideConversation API

GlideCurrencyConfig
    GlideCurrencyConfig()
    GlideCurrencyConfig(Object ed)

GlideCurrencyConverter
    GlideCurrencyConverter

GlideCurrencyParser
    GlideCurrencyParser()

GlideCurrencyValue
    GlideCurrencyValue

GlideDateTime
    GlideDateTime

GlideDate API
    GlideDateScoped

GlideDateTime
    GlideDateTime

getReferenceCurrencySource() 1989
GlideCurrencyConfig() 1990
GlideCurrencyConverter 1996
GlideCurrencyValue 2008
GlideCurrencyValue 2009
GlideDateTime 2015, 4411
GlideDateScoped 2009
GlideDateTime 2015, 4411
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDateTime time)
    GlideDate...
addMonthsLocalTime(Number months)
addMonthsUTC(Number months)
addSeconds(Number seconds)
addWeeks(Number weeks)
addWeeksLocalTime(Number weeks)
addWeeksUTC(Number weeks)
addYears(Number years)
addYearsLocalTime(Number years)
addYearsUTC(Number years)
compareTo(GlideDateTime dateTime)
equals(Object GDT)
getDate()
getDayOfMonth()
getDayOfMonthLocalTime()
getDayOfMonthUTC()
getDayOfWeek()
getDayOfWeekLocalTime()
getDayOfWeekUTC()
getDaysInMonth()
getDaysInMonthLocalTime()
getDaysInMonthUTC()
getDisplayValue()
getDisplayValueInternal()
getDSTOffset()
getErrorMsg()
getInternalFormattedLocalTime()
getInternalMidnight(Number dayOfTheWeek)
getLocalDate()
getLocalTime()
getMonth()
getMonthLocalTime()
getMonthUTC()
getNumericValue()
getSpanTime(Number dayOfWeek)
getTime()
getTZOffset()
getUserFormattedLocalTime()
getUserTimeZone()
getUTCMidnight(Number dayOfWeek)
getUTCMidnight(Number dayOfWeek)
getValue()
2044
getWeekOfYearLocalTime()
2045
getWeekOfYearUTC()
2046
getYear()
2046
getYearLocalTime()
2047
getYearUTC()
2048
GlideDateTime()
2048
GlideDateTime(GlideDateTime gDT)
2049
GlideDateTime(String dateTime)
2048
hasDate()
2050
isDST()
2051
isValid()
2052
setDayOfMonth(Number day)
2052
setDayOfMonthLocalTime(Number day)
2053
setDayOfMonthUTC(Number day)
2054
setDisplayValue(String asDisplayed)
2054
setDisplayValue(String dateTime, String format)
2055
setDisplayValueInternal(String dateTime)
2056
setDisplayValueInternalWithAlternates(String dateTime)
2057
setGlideDateTime(GlideDateTime gDT)
2057
setInitialValue(String dateTime)
2058
setMonth(Number month)
2058
setMonthLocalTime(Number month)
2059
setMonthUTC(Number month)
2060
setNumericValue(Number milliseconds)
2060
setTZ(TimeZone timeZone)
2061
setValue(Object dateTime)
2061
setValueUTC(String dateTime, String format)
2063
setYear(Number year)
2064
setYearLocalTime(Number year)
2065
setYearUTC(Number year)
2065
subtract(GlideDateTime start, GlideDateTime end)
2067
subtract(GlideTime time)
2066
subtract(Number milliseconds)
2066
toString()
2068
GlideDateTime API
GlideDialogWindow
2111
adjustBodySize()
2112
destroy()
2112
GlideDialogWindow()
2113
render()
2114
setPreference()
2115
setSize()
2115
setTitle()
2116

GlideDocumentV3
getElement(String selector, Element context)
2123
getElements(String selector, Element context)
2123

GlideDocumentV3 API
2122
GlideElement
1112, 1152, 1709, 1828, 1853, 1923, 2111, 2132, 2196, 2277, 2281, 2338, 2602, 2648, 2656, 2807, 2831, 3261, 3307, 3893, 4090, 4163, 4267, 4271, 4279, 4298, 4304, 4313, 4315, 4315, 4336, 4360
canCreate()
2132, 2160
canRead()
2132, 2160
canWrite()
2133, 2161
changes()
2133
changesFrom(Object value)
2134
changesTo(Object value)
2135
debug(Object o)
2136
getAttribute(String attributeName)
2137, 2164
getBaseTableName()
2138
getChoices(String value)
2139
getChoiceValue()
2139
debugCount()
2140
dependent()
2140
dependentTable()
2140
getDisplayValue(Number maxChar)
2141
displayValueExt(Number maxChar, String nullSub)
2141
ed()
2142
getElementValue(String value)
2143
geror()
2143
getEscapedValue()
2144
getFieldStyle()
2144
glideObject()
2145
getGlideRecord()
2145
getHTMLValue(Number maxChars)
2146
getHTMLValueExt(Number maxChar, String nullSub)
2147

ab
getJournalEntry(Number mostRecent)
    2147, 2169
getLabel()
    2148, 2170
getName()
    2149, 2171
getRefRecord()
    2149
getStyle()
    2150
getTableName()
    2151, 2172
getTextAreaDisplayValue()
    2152
getXHTMLValue()
    2152
getXMLValue()
    2152
hasAttribute(String attributeName)
    2153
hasRightsTo(String operationName)
    2154
hasValue()
    2154
nil()
    2154
setDateNumericValue(Number milliseconds)
    2155
setDisplayValue(Object displayValue)
    2156
setError()
    2156
setInitialValue(Object value)
    2157
setValue(Object value)
    2158
toString()
    2159
GlideEncrypter
    2196
decrypt(String encryptedString)
    2197
encrypt(String clearString)
    2197
GlideEncrypter()
    2198
GlideEncrypter(String key)
    2198
GlideEvaluator
    2198
GlideExcelParser
    close()
    2204
columnHeaders()
    2205
gErrorMessage()
    2205
getRow()
    2206
gTableInfo()
    2208
GlideExcelParser()
    2210
next()
    2210
parse(GlideScriptableInputStream inputstream)
    2211
setHeaderRowNumber(Number headerRowNumber)
    2213
setNullToEmpty(Boolean empty)
    2214
setSheetName(String sheetName)
    2214
setSheetNumber(Number sheetNumber)
    2214
GlideFilter
    2217
checkRecord(GlideRecord
now_GR, String filter,
Boolean match) 2221, 2223
GlideFlow 2225
GlideForm 2233
addDecoration(String
fieldName, String icon,
String title, String color) 2234
addDecoration(String
fieldName, String icon,
String title) 2233
addErrorMessage(String
message) 2235
addInfoMessage(String
message) 2236
addOption(String
fieldName, String
choiceValue, String
choiceLabel, Number
choiceIndex) 2237
addOption(String
fieldName, String
choiceValue, String
choiceLabel) 2237
clearMessages() 2238
clearOptions(String
fieldName) 2240
clearValue(String
fieldName) 2240
disableAttachments() 2240
enableAttachments() 2241
flash(String widgetName,
String color, Number count) 2241
getActionName() 2242
getBooleanValue(String
fieldName) 2243
getControl(String
fieldName) 2243
getDecimalValue(String
fieldName) 2244
getElement(String id) 2244
getFormElement() 2245
getHelpTextControl(String
fieldName) 2245
getIntValue(String
fieldName) 2245
getLabelOf(String
fieldName) 2246
getOption(String fieldName,
String choiceValue) 2246
getReference(String
fieldName, Function
callBack) 2248
getRelatedListNames() 2247
getSectionNames() 2249
getSections() 2249
getTableName() 2250
getUniqueValue() 2250
getValue(String fieldName)
removeOption(String fieldName, String choiceValue)
2260
save()
2260
setDisabled(String fieldName, Boolean disable)
2261
setDisplay(String fieldName, Boolean display)
2261
setLabelOf(String fieldName, String label)
2262
setMandatory(String fieldName, Boolean mandatory)
2263
setReadOnly(String fieldName, Boolean readOnly)
2263
setSectionDisplay(String sectionName, Boolean display)
2264
setValue(String fieldName, String value, String displayValue)
2266
setValue(String fieldName, String value)
2264
setVisible(String fieldName, Boolean display)
2267
showErrorBox(String input, String message)
2268
showErrorBox(String name, String message, Boolean scrollForm)
2269
showFieldMsg(String field, String message, String type, Boolean scrollForm)
2270
showFieldMsg(String field, String message, String type)
2269
showRelatedList(String listTableName)
2271
showRelatedLists()
2271
submit()
2272
submit(String verb)
2272
GlideGuidV3
  generate(Number stringLength)
2277
GlideGuidV3 API
2277
GlideHTTPRequest
2277
GlideHTTPResponse
2281
GlideImpersonate
2285
  canDebug()
2285
  canImpersonate()
2285
  impersonate()
2286
  isImpersonating()
2287
GlideImportLog
2288
  error()
2288
  getImportRunHistory()
2289
GlideImportLog()
2289
  info()
2290
  setImportRunHistory()
2291
  warn()
2291
GlideImportSetRun
2292
  getImportSetRunSysID()
2292
  GlideImportSetRun()
2293
GlideImportSetTable
2293
  GlideImportSetTable()
2294
GlideImportSetTransformer
2296
  getImportSetRun()
2297
  GlideImportSetTransformer()
2297
  isError()
2298
  setImportSetID()
2299
  setImportSetRun()
2299
  setLogger()
2300
  setMapID()
2301
  setSyncImport()
2301
  transformAllMaps()
2302
GlideImportSetTransformMap
2303
GlideList2
2306
  addFilter()
2306
  get()
2306, 2307
  getChecked()
2307
getFixedQuery()  
2308
getGroupBy()  
2309
gListName()  
2309
getOrderBy()  
2310
getParentTable()  
2310
getQuery()  
2311
getRelated()  
2312
getTableName()  
2312
getTitle()  
2313
getView()  
2313
isUserList()  
2314
refresh()  
2314
refreshWithOrderBy()  
2315
setFilter()  
2316
setFilterAndViewRefresh()  
2317
setFirstRow()  
2317
setGroupBy()  
2318
setOrderBy()  
2318
setRowsPerPage()  
2319
showHideGroups()  
2320
showHideList()  
2320
sort()  
2321
sortDescending()  
2322
toggleList()  
2322
toggleListNoPref()  
2323
GlideListProperties  
getListID()  
2339
getListName()  
2339
getTitle()  
2340
GlideListProperties()  
2339
hasActions()  
2340
hasBottomNav()  
2341
hasBottomVCR()  
2341
hasFilter()  
2342
hasHeader()  
2343
hasHeaderContextMenu()  
2343
hasListMechanic()  
2344
hasPopup()  
2344
hasRowContextMenu()  
2345
hasSearch()  
2345
hasTitle()  
2346
hasTitleContextMenu()  
2347
hasTopVCR()  
2347
isOmitFilter()  
2348
isSaveFilterHidden()  
2348
isShowLinks()
GlideListV3
addFilter(String filter)

get(Object DomElement)

get(String listId)

getchecked()

getFixedQuery()

getFormTarget()

groupBy()

getListName()

getOrderBy()

getParentTable()

getQuery(Object options)

getReferringUrl()

getRelated()

getRelatedListType()

getRelationshipId()

getRowCount()

getRowsPerPage()

getTableName()

setSaveFilterHidden()

setShowLinks()

setTitle(String title)

setVCR()

GlideListV3
addFilter(String filter)

get(Object DomElement)

get(String listId)

getchecked()

getFixedQuery()

getFormTarget()

groupBy()

getListName()

getOrderBy()

getParentTable()

getQuery(Object options)

getReferringUrl()

getRelated()

getRelatedListType()

getRelationshipId()

getRowCount()

getRowsPerPage()

getTableName()
getTitle() 2330
ggetView() 2331
isUserList() 2331
refresh(Number firstRow, Object additionalParams) 2332
refreshWithOrderBy(Number firstRow, Object additionalParams) 2332
setFilter(String filter, Boolean saveOrderBy, Boolean saveGroupBy) 2333
setFilterAndRefresh(String filter) 2333
setFirstRow(Number firstRow) 2334
setFormTarget(String target) 2334
setGroupBy(String groupBy) 2334
setOrderBy(String orderBy) 2335
setReferringUrl(String url) 2335
setRowsPerPage(Number numRows) 2336
showHideGroups(Boolean showFlag) 2336
showHideList(Boolean showFlag) 2337
sort(String field) 2337
sortDescending(String field) 2337
toggleList() 2338
toggleListNoPref() 2338
GlideListV3 API 2323
GlideMenu
clearImage(GlideMenuItem item) 2364
clearSelected() 2364
getItem(String itemId) 2365
setDisabled(GlideMenuItem item) 2365
setEnabled(GlideMenuItem item) 2366
setHidden(GlideMenuItem item) 2366
setImage(GlideMenuItem item, String imgSrc) 2367
setLabel(GlideMenuItem item, String label) 2367
setVisible(GlideMenuItem item) 2368
GlideMenu API 2364
GlideModalFormV3
addParm(String name, String value) 2368
GlideModalForm(String title, String tableName, Function onCompletionCallback, Boolean readOnly) 2369
render() 2370
setCompletionCallback(Function callbackFunction)
setOnloadCallback(Function callbackFunction)
setSysID(String sys_id)

GlideModalFormV3 API

GlideModalV3
get(String id)
getPreference(String name)
GlideModal(String id, Boolean readOnly, Number width)
render()
renderWithContent(Object html)
renderWithContent(String html)
setPreference(String name, String value)
setPreferenceAndReload(Array properties)
setTitle(String title)
setWidth(Number width)
switchView(String newView)

GlideModalV3 API

GlideNavigationV3 API

open(String url, String target)
openRecord(String tableName, String sys_id)
refreshNavigator()
reloadWindow()

GlideNotificationV3 API

show(String type, String message)

GlideOAuthClient API

requestToken(String clientName, String jsonString)
requestTokenByRequest(String clientName, GlideOAuthClientRequest request)
revokeToken(String clientName, String accessToken, String refreshToken, GlideOAuthClientRequest request)

GlideOAuthClientRequest
gerantType()
ger HEADER(String name)
ger HEADERS()
ger PARAMETER(String name)
getPassword() 2388
getRefreshToken() 2388
getScope() 2388
getUserName() 2389
setGrantType(String grantType) 2389
setHead(String name, String value) 2390
setParameter(String name, String value) 2390
setPassword(String password) 2390
setRefreshToken(String refreshToken) 2391
setScope(String scope) 2392
setUserName(String userName) 2392
GlideOAuthClientResponse
getAccessToken() 2395
getAccessTokenSysID() 2396
getBody() 2393
getContentType() 2393
getErrorMessage() 2394
getExpiresIn() 2397
getRefreshToken() 2398
getRefreshTokenSysID() 2399
getResponseCode() 2394
getResponseParameters() 2394
getScope() 2395
getToken() 2399
GlideQueryCondition 2439
addCondition(String name, String oper, Object value) 2440
addOrCondition(String name, String oper, Object value) 2441
gliderecord 4460
GlideRecord 2457, 4409
_next() 2509
_query(String field, String value) 2515, 2574
addActiveQuery() 2458, 2532
addDomainQuery(Object glideRecord) 2458
addEncodedQuery(String query) 2460
addFunction(Object function) 2461
addInactiveQuery() 2462
addJoinQuery(String table, String primaryField, String joinTableField) 2465
addJoinQuery(String table, String primaryField)
addJoinQuery(String table)
addNotNullQuery(String fieldName)
addNullQuery(String fieldName)
addQuery(String name, Object operator, Object value)
applyTemplate(String template)
autoSysFields(Boolean e)
canCreate()
canDelete()
canRead()
canWrite()
changes()
dateNumericValue()
deleteMultiple()
deleteRecord()
find(String columnName, String value)
get(Object name, Object value)
getAttribute(String fieldName)
getClassDisplayValue()

getED()
getElement(String fieldName)
getEncodedQuery()
getEscapedDisplayValue()
getFields()
getLabel()
getLink(Boolean noStack)
getLocation()
getPlural()
getRelatedLists()
getRelatedTables()
getRowCount()
getRowNumber()
getTableName()
getValue(String fieldName)
GlideRecord(String tableName)

hasAttachments()
hasNext()
initialize()
insert()
insertWithReferences()
instanceOf(String className)  
isValid()  
isValidField(String fieldName)  
isValidRecord()  
newRecord()  
next()  
operation()  
orderBy(String fieldName)  
orderByDesc(String fieldName)  
query tables 4460, 4462, 4464  
query(String field, String value)  
restoreLocation()  
saveLocation()  
setAbortAction(Boolean b)  
setDisplayValue(String name, Object value)  
setEncodedQuery(String encodedQuery)  
setForceUpdate(Boolean force)  
setLimit(Number limit)  
setLocation(Number rowNumber)  
setNewGuid()  
setNewGuidValue(String guid)  
setQueryReferences(Boolean queryReferences)  
setUseEngines(Boolean e)  
setValue()  
setWorkflow(Boolean e)  
update(Object reason)  
updateMultiple()  
updateWithReferences(Object reason)  
GlideRecord variable set to null  
GlideRecordSecure 4409  
GlideRecordUtil  
getCI GR(String sys_id)  
getFields(GlideRecord now_GR)  
getGR(String baseTable, String sys_id)  
getTables(String tableName)  
mergeToGR(Object hashMap, GlideRecord now_GR, Object ignore)
populateFromGR(Object hashMap, GlideRecord now_GR, Object ignore)
2584
GlideRecordUtil API
2580
GlideScopedEvaluator
2198, 2202
evaluateScript(GlideRecord grObj, String scriptField, Object variables)
2198, 2202
getVariable(String name)
2200
GlideScopedEvaluator()
2200, 2203
putVariable(String name, Object value)
2201, 2204
GlideSecurityUtils
2602
cleanURL(String url)
2602
ensureRelativeURL(String url)
2602
escapeScript(String script)
2603
isURLWhiteListed(String url)
2603
GlideServletRequest
2604
GlideSession
2610
clearClientData(String paramName)
2610
getClientData(String paramName)
2610
getLanguage()
2611
getRoles()
2612
getTimeZoneName()
2612
isInteractive()
2613
isLoggedIn()
2614
putClientData(String paramName, String paramValue)
2615
GlideSPScriptable
2639
getRecordVariablesArray(GlideRecord now_GR)
2639
GlideStringUtil
2656
dotToUnderBar(String sourceString)
2656
escapeAllQuotes(String sourceString)
2657
escapeForHomePage(String sourceString)
2657
escapeHTML(String htmlString)
2658
escapeNonPrintable(String sourceString)
2658
escapeQueryTermSeparator(String sourceString)
2659
escapeTicks(String sourceString)
2659
getHTMLValue(String sourceString)
2660
getNumeric(String sourceString)
2661
isBase64(String sourceString)
2661
isEligibleSysID(String sourceString)
2662
newLinesToBreaks(String sourceString)  2662
normalizeWhitespace(String sourceString)  2663
unescapeHTML(String htmlString)  2663
GlideSystem  2674, 4411
addErrorMessage(Object message)  2674
addInfoMessage(Object message)  2675
addMessage(String type, Object message)  2676
beginningOfLastMonth()  2677, 2751
beginningOfLastWeek()  2677, 2752
beginningOfNextMonth()  2678, 2752
beginningOfNextWeek()  2678, 2752
beginningOfNextYear()  2679, 2753
beginningOfThisMonth()  2680, 2753
beginningOfThisQuarter()  2680, 2754
beginningOfThisWeek()  2681, 2754
beginningOfThisYear()  2682, 2754
beginningOfToday()  2682
beginningOfTomorrow()  2683
beginningOfYesterday()  2683
calDateDiff(String startDate, String endDate, Boolean numericValue)  2684
dateDiff(String startDate, String endDate, Boolean numericValue)  2685
dateGenerate(String date, String range)  2686
daysAgo(Number days)  2687, 2755
daysAgoEnd(Number days)  2688, 2756
daysAgoLocal  2688
daysAgoStart(Number days)  2689, 2756
daysAgoStart(Number days)  2689, 2756
endOfLastMonth()  2689, 2758
endOfLastWeek()  2690, 2758
endOfLastYear()  2691, 2759
endOfNextMonth()  2692, 2759
endOfNextWeek()  2692, 2759
endOfNextYear()  2693
endOfNextYear()  2694, 2760
endOfThisMonth()  2694, 2760
endOfThisQuarter()  2695, 2760
endOfThisWeek()  2696, 2761
endOfThisYear()  2696, 2761
endOfToday()  2697
endOfTomorrow()  2698
endOfYesterday()  2699
eventQueue(String name, Object glideRecord, String parm1, String parm2, String queue)
2699
eventQueueScheduled(String name, Object glideRecord, String parm1, String parm2, Object expiration)
2700
flushMessages()
2701
getAvatar()
2701
getCurrentScopeName()
2702
getDisplayColumn(String tableName)
2703
getDisplayValueFor(String tableName, String recordID, String fieldName)
2703
gerErrorMessages()
2704, 2768
getEscapedProperty(String key, Object substituteObject)
2705
getImpersonatingUserDisplayName()
2705
getImpersonatingUserName()
2706
getErrorMessages()
2707
getInitials()
2707
getMessage(String messageID, Object args)
2708
getMessageS(String messageID, Object args)
2709
getNodeName(Object obj, Number index)
2711
getNodeValue(Object obj, Number index)
2710
getPreference(String key, Object default)
2712
getProperty(String key, Object alt)
2712
getScriptError(String script)
2713
getSession()
2714
getSessionID()
2715
getStyle(String tableName, String fieldName, String fieldValue)
2715
getUser()
2716
getUserDisplayName()
2717
getUserID()
2717
getUserName()
2718
getUserNameByUserID(String user_id)
2719
getXMLNodeList(String xml)
2719
getXMLText(String xml, String xpathQuery)
2720
hasRole(String roleName)
2721
hasRoleInGroup(Object roleName, Object group)
2721
hoursAgo(Number hours)
2722, 2777
hoursAgoEnd(Number hours)
hoursAgoStart(Number hours)

isFirstDayOfMonth(Object date)

isFirstDayOfWeek(Object date)

isFirstDayOfYear(Object date)

isInteractive()

isLastDayOfMonth(Object date)

isLastDayOfWeek(Object date)

isLastDayOfYear(Object date)

isLoggedIn()

isMobile()

lastWeek()

log(String message, String source)

logError(String message, String source)

logWarning(String message, String source)

minutesAgoStart(Number minutes)

monthsAgo(Number months)

monthsAgoEnd(Number months)

monthsAgoStart(Number months)

nil(Object obj)

now()

nowDateTime()

nowGlideDateTime

nowNoTZ()

quartersAgo(Number quarters)

quartersAgoEnd(Number quarters)

quartersAgoStart(Number quarters)

setProperty(String key, String value, String description)

setRedirect(Object URI)

setReturn(Object URI)

tableExists(String tableName)

userID()

workflowFlush(Object glideRecord)
getFileFromPath()
set(String name, String value)
toString(String path)
GlideURLV3
addParam(String name, String value)
getURL(Object additionalParams)
GlideURL(String contextPath)
GlideURLV3 API
GlideUser
firstName
getClientData(string key)
getCompanyID()
getDisplayName()
getDomainDisplayValue()
getDomainID()
getEmail()
getFirstName()
getFullName()
getID()
getLastName()
getMyGroups()

GlideUser (client)
setClientData()
GlideXMLUtil
removeInvalidChars(String xmlString)
validateXML(String xmlString, Boolean nsAware, Boolean forgiveUnclosed)

hasRole()
hasRole(String role, Boolean includeDefaults)
hasRoleExactly(String role, Boolean includeDefaults)
hasRoleFromList(String roles, Boolean includeDefaults)
hasRoles(Boolean includeDefaults)
isMemberOf()
lastName
userID
userName

GlideUser (client)
setClientData()
grant access
  update set picker
GSLog
Guided Tours API
  applyListFilter()
  endTour()
  events.off()
  events.on()
  getAllTours()
  loadPlayer()
  startTour()

hierarchical test suites
hierarchical tests
hierarchical data
  insert
HistoryWalker
  getUpdateNumber()
  getWalkedRecord()
  getWalkedRecordCopy()
  HistoryWalker(String tableName, String sysId, Boolean useAudit)
  HistoryWalker(String tableName, String sysId)
  isFieldLevelSecurity()
  isRecordLevelSecurity()
  isWithChanges()
  isWithJournalFields()
  isWithVariables()
  setFieldLevelSecurity(boolean fieldLevelSecurity)
  setRecordLevelSecurity(Boolean recordLevelSecurity)
  setWithChanges(Boolean withChanges)
  setWithJournalFields(Boolean withJournalFields)
  setWithVariables(Boolean withVariables)
  walkBackward()
  walkForward()
  walkTo(int updateCount)
HostnameJS
  format(String hostname, String source)
  getDomainName()
  getSysName()
HostnameJS API
  HR Core API
  hr_ActivitySet
  createLECaseByService()
hasActiveCaseByService()
hr_ActivitySet
hr_ActivityUtils
createCaseFromProducer()
hr_Utils
getPrimaryJob()
switchPrimaryJob()
hr_Utils API
http client connection
management
HTTP connection
management
properties
HTTP Response Codes
HttpRequestAuthedData
HttpRequestData
Human resources
format(String message, Object map)
getMessage(String msgKey, Function callback)
getMessages(Array msgKeys, Function callback)
IdentificationEngine
createOrUpdateCI(String source, String input)
createOrUpdateCIEnhanced(String source, String input, String options)
identifyCI(String jsonString)
runIdentificationAudit(GlideRecord now_GR)
IdentificationEngineScriptableApi
createOrUpdateCI(String source, String input)
createOrUpdateCIEnhanced(String source, String input, Object options)
i18NV3 API
formatICalComponent(Array arrEvents)
formatICalEvent(Array arrEvent, Boolean useAlarm)
formatRecurringRule(Object ruleObj)
getDateFromScheduleDateTime(GlideScheduleDateTime scheduleDateTime)
getSDT(String sdtStr, String timeZone)
getTimeFromScheduleDateTime(GlideScheduleDateTime scheduleDateTime)
formatICalUtilSNC
formatICalComponent(Array arrEvents)
formatICalEvent(Array arrEvent, Boolean useAlarm)
formatRecurringRule(Object ruleObj)
getDateFromScheduleDateTime(GlideScheduleDateTime scheduleDateTime)
getSDT(String sdtStr, String timeZone)
getTimeFromScheduleDateTime(GlideScheduleDateTime scheduleDateTime)
3113
identifyCl(String jsonString)
3159unIdentificationAudit(GlideRecord now_GR)
3200
IdentifyReconcile API
6494
IDResult
IDResult(Boolean explore,
String sysID)
3200
IDResult API
3200
IEvent
getQueueSize()
3201
IEventSender
3201, 3202
isFullQueue()
3202
impersonate
879
Import
2292
Import Set API
6683
Import Set Transformer object
2296
in-bound access
43
inbound REST
660
Inbound REST
4953
OAuth
4985
OAuth example
4986
response codes
4987
inbound SOAP
insert related records
7413
Inbound web services
examples
7435
incident
example
priority 1
7487
priority 1
example
7487
installation settings
4452
interaction
accept()
3212
Interaction
3209
create(Object options)
3209
getInteraction(GlideRecord interaction)
3210
transferToAgent(String sysID)
3210
transferToQueue(String sysID)
3211
Interaction Management API
6698
IPAddressFixup
dedupe(String tableName,
String ip)
3212
dedupeAll()
3213
fix()
3213
getParentIP()
3214
setParentIP(String ip)
3214
syncIP()
3214
IPAddressFixup API
3212
IPService
creates
3215
description
3215
getFromArrayList(Array list)
3215
IPService(Object source)
3216
name
3216
port
3216
protocol
3216
serviceName
3217
sysID
3217
IPService API
3215

J

J2js
3220
j2js(Object javaObject)
3221
Java Apache Axis2 Web Services Client Examples
7435, 7435, 7436, 7438
JavaScript debug window
4634
using
4634
JavaScript engine
4595
JavaScript log
4632
JavaScript Log
4633, 4633
JavaScript modes
4595
JavaScript regular expressions
4386, 4387
Java syntax
4386
JavascriptProbe
addParameter(String name,
String value)
3217
create()
3218
JavascriptProbe(String
midServer)
3218
setName(String name)
3219
setSource(String ip)
3220
JavascriptProbe API
3217
Jelly
4443
jelly replacement
add escaping
4531
JSON
decode(String str)
3222
encode(Object jsonObject)
3223
JSON()
3224
parse(String str)
3224
stringify(Object jsonObject)
3225, 3227
JSON API
3222
JSON Scoped API
3227
JSONParser
JSONParser()
3226
JSONParser API
3226
JSUtil
doesNotExist(Object item)
3272
escapeAttr(String text)
escapeText(String text)
getBooleanValue(GlideRecord, String field)
has(Object item)
instance_of(Object item, String class)
isJavaObject(Object value)
logObject(Object obj, String name)
nil(Object item)
notNil(Object item)
toBoolean(Object item)
type_of(Object value)
unescapeAttr(String text)
unescapeText(String text)

JSUtil API
K
Knowledge Management
REST API
6709
L
Legacy Notify
3541
limit the number of results
use filters
7460
Logger
log(String msg)
logError(String msg)
Logger API
3301
Logging
2288
LSOFParser
error(String msg)
initFileDescriptor()
initProcess()
on_c(String line)
on_endFileDescriptor()
on_endProcess()
on_f(String line)
on_n(String line)
on_p(String line)
on_T(String line)
parse(String lsOfOutput)
parse(String lsOfOutput)
Logger API
3303
LSOFParser API
3303
M
Managing Services
383, 385, 385, 385, 386, 387, 387, 388, 389
maximum number of rows returned
specify
7583
MetricBase Time Series API
6743
Microsoft
.NET
7441, 7442
web services client
examples
7441, 7442

MIDServer
getByName(String name)
3309
getDefault(DiscoverySchedule schedule)
3310
getForPing(DiscoverySchedule schedule, DiscoveryRange range)
3310
hostname
3310
hostOS
3311
ip
3311
name
3311
network
3311
routerIP
3311
status
3312
sysID
3312
url
3312
version
3312
windowsDomain
3313

MIDServer API
3309

MIDServerAjax
ajaxFunction_testProbe()
3313
MIDServerAjax()
3313
MIDServerAjax API
3313

MIDServerCluster
clusterExists()
3314
getClusterAgent()
3314
MIDServerCluster(GlideRecord agent)
3315
MIDServerCluster API
3314

MIDServerFileSync
notifyMIDServers(String table)
3316
MIDServerFileSync API
3315

MIDServerFinder
getMIDServers()
3317
getMIDServersBySysId()
3317
getStatusMessage()
3318
MIDServerFinder()
3318
setActive(Boolean flag)
3318
setCapabilities(Array capabilities)
3319
setDebug(Boolean onOrOff)
3319
setRanges(String ranges)
3320
setRangesByIPorHostname(String ipOrHostname)
3320
MIDServerFinder API
3316

MIDServerManage
grab_logs(String agentName, String logs)
3321
MIDServerManage()
3322
restart(String agentName)
3322
stop(String agentName)
3322
system_msg(String
agentName, String cmd,
String name)
3323
text_probe(String
agentName, String probeID,
String topic, String ename,
String source)
3323
upgrade(String
agentName)
3324
MIDServerManage API
3321
MIDServerSelector
findAgent(String target)
3325
getError()
3325
getWarning()
3325
MIDServerSelector()
3326
setCapabilities(Array
capabilities)
3326
MIDServerSelector API
3324
MLPredictor
    MLPredictor()
3335
MLSolution
3342
    getClusterAssignments()
3344
    getClusterForRecord()
3346
    getClusterInfo()
3346
MLSolutionFactory
3352
MLSolutionResult
    findActiveSolution()
3353
    getClusterAssignments()
3354
    getClusterInfo()
3355
    MLSolutionResult()
3356
MLSolutionUtil
3356
    MLSolutionUtil()
3356
Mobile Client GlideForm (g
form) scripting
4514
mobile deep link
3359
mobile deep linking
3359
Mobile g_form
2273, 2274, 2274, 2275
Mobile Glideform
2273, 2274, 2274, 2275
Mobile GlideForm
2273
MobileDeepLinkGenerator
3359
    getFormScreenLink(String
formScreenId, String
tableName, String
recordSysId)
3361
    getScreenLink(String
documentId, Object
uiParams)
3361
    MobileDeepLinkGenerator(String
clientType)
3360
Modify REST authorization
credentials
3837
MonitorMIDServer
    killOldRequests()
markDowners()
monitor()
MonitorMIDServer()
sendHeartbeatRequests()
MonitorMIDServer API
move test
multi-table processor
creating
Mutual authentication
Outbound web services
Protocol profile
namespace identifier

NameValuePairs
mapToString()
NameValuePairs(Object mapForm)
NameValuePairs(String stringForm)
quotify(Object value)
stringToMap()
NameValuePairs API
NetwareHandler
classifyAndIdentify()
NetwareHandler API
new service categories
Email notifications
forms
service catalog categories
tables
user roles
news
adding
scroller
7488
scroller
adding
7488
news panel
adding
scrolling
7488, 7488
scrolling
adding
7488, 7488
Notify API
call(String notifyPhoneNumber,
String toPhoneNumber,
GlideRecord conferenceCall,
GlideRecord conferenceCallRecord,
String userSysId, String groupSysId, GlideRecord sourceRecord)
call(GlideRecord callRecord)
dequeueCall(GlideRecord callRecord)
forwardCall(GlideRecord call, String destination, String dtmf)

getAvailableClients(String notifyNumber)

getParentCallID(GlideRecord callRecord)

getPhoneNumbers()

getTokens(GlideRecord record)

generateTokenTTL(String owner)

hasCapability(String notifyPhoneNumber, String capability)

kick(GlideRecord user)

modifyCall(GlideRecord callRecord, NotifyAction notifyAction)

mute(GlideRecord participantRecord)

queueCall(GlideRecord callRecord)

sendBulkSMS(NotifyPhoneNumber notifyPhoneNumber, String toPhoneNumbers, String messageBody, GlideRecord source)

sendSMS(NotifyPhoneNumber notifyPhoneNumber, String toPhoneNumber, String messageBody, GlideRecord source)

unmute(GlideRecord participantRecord)

Notify On Task

Notify participant

Notify participant management

NotifyAction

addConference.setEndOnExit(Boolean endOnExit)

addConference.setHangupOnStar(Boolean hangupOnStar)

addConference.setMuted(Boolean muted)

addConference.setName(String name)

addConference.setRecord(Boolean record)

addConference.setStartOnEnter(Boolean startOnEnter)

addConference()

addDial.setCallerID(String callerID)

addDial.setClientRecord(String tableName, String sysID)

addDial.setDTMF(String value)

addDial.setHangupOnStar(Boolean hangupOnStar)
addDial.setPhoneNumber(String phoneNumber)
3409
addDial.setRecord(Boolean record)
3410
addDial.setTimeout(Integer timeout)
3410
addDial()
3405
addGather.addPlay()
3412
addGather.addSay()
3413
addGather.setFinishKey(String finishKey)
3414
addGather.setNumberOfDigits(Integer numberOfDigits)
3415
addGather.setTimeout(Integer timeout)
3415
addGather()
3411
addHangUp()
3416
addPlay.setLoop(Integer loop)
3419
addPlay.setURL(String url)
3419
addPlay()
3418
addQueue.setDequeue(Boolean dequeue)
3417
addQueue.setName(String name)
3418
addQueue()
3416
addRecord.setFinishKey(String finishKey)
3421
addRecord.setMaxDuration(Integer seconds)
3421
addRecord.setTimeout(Integer timeout)
3422
addRecord()
3420
addReject()
3423
addSay.setLanguage(String language)
3425
addSay()
3424, 3426
addSMS.setMessage(String message)
3427
addSMS.setTo(String to)
3428
append(NotifyAction action)
3429
fromJson(String json)
3429
setCallRecord(GlideRecord callRecord)
3431
setText()
3426
NotifyAction
toJson()
3431
NotifyAction
toJson()
forwardCall(Object argument)
3439
getAvailableClients(Function callback)
3440
getParentId(String callId, Function callback)
3441
getStatus()
3442
hangupCall()
3442
init()
3443
mute(Boolean muted)
3443
Notify WebRTC Client
3433
Object notifyConfig, Boolean initializeVendorClientLazily
3437
pickupCall()
3444
SendDtmf(String digits)
3444
setCallerId(String value, Function autoSelectVendorCallback)
3446
setClientAvailable(Boolean available)
3445
NotifyConferenceAPI
addParticipants(Object data)
3508
endConference(Object data)
3520
NotifyConferenceUtil
3447
NotifyConferenceUtils
addToConferenceByPhoneNumber(String toNumber, GlideRecord confGR)
3447
addToConferenceByUserId(String userId, GlideRecord confGR)
3449
doConferenceAction(String action, Object data)
3450
getConferenceInputDataTemplate()
3458, 3521
getServiceProviderCapabilities()
3463
isActionSupported(Number action)
3466
kickByParticipantGR(GlideRecord notifyParticipantGR)
3469
muteByParticipantGR(GlideRecord notifyParticipantGR)
3471
NotifyConferenceUtils
3472
unmuteByParticipantGR(GlideRecord notifyParticipantGR)
3473
NotifyNow
3541
addConferenceCallParticipant(String conferenceCall, String participant)
3541
convertLocalPhoneNumberToE164(String userID, String phoneNumber)
3554
getConferenceCallParticipants(String conferenceCallId, Boolean isCallable)
3555
getFrequentlyCalledUsers(Number limit)
getPreferredE164SMSNumber(GlideRecord user)
getPreferredE164VoiceNumber(GlideRecord user)
getPreferredEmailAddress(GlideRecord user)
getReadyState()
getStatus()
initiateConferenceCall(String, String,GlideRecord, Boolean)
isCallable(String participant)
isSMSCapable()
isSMSCapable(String userID)
isVoiceCapable()
isVoiceCapable(String userID)
kick(GlideRecord participant)
mute(GlideRecord participant)
sendEmailQuestion(String emailAddress, String question, GlideRecord sourceRecord, String emailSubject)
sendSMS(String phoneNumber, String smsBody)
sendSMSQuestion(String phoneNumber, String question, GlideRecord sourceRecord)
umute(GlideRecord participant)
NotifyOnTaskClient
doConferenceAction(String action, Object data)
start(Object data)
NotifyPhoneNumber
getDialCode()
getID()
getNumber()
getOwner()
getTerritory()
isShortCode
supportsConferenceCall()
supportsIncomingPhoneCall()
supportsIncomingSMS()
supportsOutgoingPhoneCall()
supportsOutgoingSMS()
supportsRecording()
supportsWebRTC()

NotifyScoped

call(String notifyPhoneNumber, String toPhoneNumber, GlideRecord conferenceCall, GlideRecord conferenceCallRecord, String userSysId, String groupSysId, GlideRecord sourceRecord)

dequeueCall(GlideRecord callRecord)

forwardCall(GlideRecord call, String destination, String dtmf)

getAvailableClients(String notifyNumber)

getPhoneNumbers()

getShortCodes()

getTokens(GlideRecord, record)

getTokenTTL(String owner)

hasCapability(String notifyPhoneNumber, String capability)

kick(GlideRecord participant)

modifyCall(GlideRecord callRecord, NotifyAction notifyAction)

mute(GlideRecord participantRecord)

queueCall(GlideRecord callRecord)

sendBulkSMS(NotifyPhoneNumber notifyPhoneNumber, String toPhoneNumbers, String messageBody, GlideRecord source)

sendSMS(NotifyPhoneNumber notifyPhoneNumber, String toPhoneNumber, String messageBody, GlideRecord source)

unmute(GlideRecord participantRecord)

OAuth

OAuthUtil

profile

provider

scope

third-party provider scenario

OAuthUtil

interceptRequestParameters(requestParamMap)

parseTokenResponse(accessTokenResponse)

OCRiCalSNC
invalRotaRespCache
(GlideRecord
rotaRespCacheGR)
3604
OCRosterSpanApprovalUtil
3585
approvePTOSpan
(GlideRecord rosterSpanGr)
3585
getContextualCalUrlPerSpanProposal
(GlideRecord rosterSpanProposalGr)
3585
getFromDateDisplayValuePerSpanProposal
(GlideRecord rosterSpanProposalGr)
3585
getPTOApproversList
(GlideRecord rosterSpanProposalGr)
3586
getToDateDisplayValuePerSpanProposal
(rosterSpanProposalGr GlideRecord)
3586
getUserNamePerSpanProposal
(GlideRecord rosterSpanProposalGr)
3587
isPTOApprovalRequired ()
3587
rejectPTOSpan
(rosterSpanGr GlideRecord)
3588
OCRotalCalendar
3607
OCRotalCalendarSNC
3598
createCalendarEvents
(String groupId, String rotaId, String userId, Object dateRangeObj)
3598
createCustomEvent (String groupName, Object scheduleItemSpan, String calendarLink)
3599
createPlaceholderCalendar
(GlideRecord rotaGR, Object dateRangeObj, String calendarLink)
3600
getCalendarEvents (String groupId, String rotaId, String userId, Object dateRangeObj, Boolean useCache)
3600
getEventsFromTable (String groupId, String rotaId, String userId, Object dateRangeObj)
3601
getIntersectRotaSpanItem
(ScheduleTimeSpan timeSpan, Array rotaSpanItems)
3601
getMemberCalendarURL
(String groupId, Object rotaId, String userId)
3602
getOnCallCalendarURL
(GlideRecord rotaGR)
3602
handleOverrideMember
(AJAXScheduleItem scheduleItem)
3603
handleRotaMember
(AjaxScheduleItem scheduleItem, Object rotaSpanItems, Object definitionItems, Array repeatRotaSpanIdArr)
3603
bg
matchRotaSpanRule(ScheduleTimeSpan timeSpan, Array rotaSpanItems, Array repeatRotaSpanIdArr, Object seriesStartTimes)

populateCalendarSubscriptionSettings()

processSeriesEvent(List scheduleItems, Array repeatRotaSpanIdArr, String userId)

saveCalendarEvents(String groupld, String rotaId, String userId, Object dateRangeObj, String result)

sendCalendarURL(GlideRecord rotaGR)

updateExceptionList(AJAXScheduleItem scheduleItem, ScheduleTimeSpan timeSpan, Array rotaSpanItems)

OOCRotaMember

deactivateUser(String userid, GlideDateTime deactivateDate)

hasChanged(GlideRecord current, GlideRecord previous)

hasOrderChanged(GlideRecord current, GlideRecord previous)

recalculate(GlideRecord current, GlideRecord previous)

validateDates(String from, String to)

OOCRotation

buildrotas

getEndDate()

getGrouplds()

getRosterIds()

getRotaGr(String rotaIds, String groupIds, String rosterIds, String userIds)

getRotaIds()

getStartDate()

getTimezone()

getUserIds()

setEndDate(String EndDate)

setGrouplds(String GroupIds)

setRosterIds(String RosterIds)

setRotaIds(String RotaIds)

setStartDate(String startDate)

setUserIds(String UserIds)
OCRotationV2
3596
groups()
3596
getRostersByRotas(String rotaSysIds)
3596
getRotasByGroup(String groupSysIds)
3597
getSpans()
3597
OCSeriesEventGenerator
3650
getMemberCalendar
(String groupId, String rotaId, String userId, Object dateRangeObj, Object memberSchedules, String calendarLink)
3650
getRosterICalEvents
(GlideRecord rotaScheduleSpanGR, GlideRecord memberScheduleSpanGR, Array seriesStartTimes, Object excludeItems, String repeatUntil, String timeZone, String calendarLink)
3650
getRosterMemberEvents
(GlideRecord rotaMemberGR, Array seriesStartTimes, Object excludeItems, String repeatUntil, String calendarLink)
3651
OCTimer
3652
log (String name, String entry)
3652
millisToTime (Integer millis)
3652
result ()
3653
start (String name)
3653
stop (String name)
3653
ODBC
7560
behavior
7579
client applications
7582
data sources
7582
debug log
7565
debug logs
generate
7577
logging level
7565
SQL queries
field length
7581
field size
7581
ODBC driver
7545, 7584, 7585
configuring
7560, 7562, 7570
create a user
7546
Crystal Reports 2008
7593
define an ACL rule for the ODBC role
7547
download and install the ODBC driver
7552
Excel 2010
7590
installation requirements
7548
installing
7556
SQL support
7584
test
7574
troubleshoot
7578
ODBC driver quick start
getting started
7546
On-call rotation
3588, 3596
OnCallContactPreferenceUtil
getCatchAllContacts()
3614
getContactsAtByEscalatee()
3610
getUserPreferences()
3613
OnCallContactPreferenceUtil()
3610
OnCallContactPreferenceUtil
API
3610
OnCallRotation
addCurrentToNotified()
3648
cancelEscalations(GlideRecord escalationGr)
3649
continueEscalations(GlideRecord escalationGr)
3649
getCatchAll(String rotaID)
3617
getchallType(String rotaID)
3617
getEscalateeAt(String groupId, GlideDateTime dateTime, Number position)
3624
getEscalatees()
3618
getEscalateesAt()
3635
getEscalationPlan()
3625
getEscalationType(String rotaID)
3633
getPrimaryUser()
3637
getPrimaryUserByRota(String rotaId)
3638
getPrimaryUserName()
3638
getPrimaryUserNameByRota(String rotaId)
3639
getPrimaryUsers()
3640
OnCallRotation()
3634
spansOverlap(ArrayList parent, ArrayList child, String timeZone)
3634
startEscalations(GlideRecord now_GR, String eventName, String escalationScriptName)
3647
who()
3645
whoAt(String groupSysId, GlideDateTime gdt, String rotaSysIds)
3647
whosNext()
3648
whosOnCall(String groupId, String rotaSysIds, String rosterSysIds, GlideDateTime gdt)
3640
OnCallRotation API
3616
onLoad
4633, 4633
openFrameAPI
3569
  hide()
  3570
  init(Object config, function
  successCallback, function
  failureCallback)
  3570
isVisible(function callback)
  3571
openCustomURL(Object
details)
  3572
openServiceNowForm(Object
details)
  3572
openServiceNowList(Object
details)
  3574
setFrameMode(mode)
  3574
setHeight(height)
  3575
setIcons(Array icons)
  3575
setPresenceIndicator(presence)
  3576
setSize(Number width,
Number height)
  3577
setSubtitle(String subTitle)
  3577
setTitle(String title)
  3578
setTitleIcon(Object icon)
  3578
setWidth(width)
  3579
show()
  3579
subscribe(openFrameAPIEVENT
event, function
eventCallback)
  3580
version()
  3583
operational metrics
  6861
order guide
  running automatically
  4465, 4466, 4467, 4469
OrderGuide
  getId()
  3665
  init(Map request)
  3665
isTwoStep()
  3667
navigateFromMap(Map
itemDetails)
  3668
OrderGuide(String sys_id)
  3664
OrderGuideJS
  isIncludeItems()
  3667
  isUseCustomCart()
  3668
Outbound REST
  7495
  authentication
  7500
Create a REST message
  7496
Define a header
  7498
Define an HTTP method
  7497
Generate a script preview
  7514
generate variables
  7513
MID Server
  7499
mutual authentication
  7512
OAuth 2.0
  Google Contacts tutorial
REST message elements
Scripting
Testing HTTP methods
Variable substitution
Outbound SOAP
  External XSD file
Outbound SOAP web service
  Video tutorial
Outbound SOAP Web Service
  basic authentication
configure soap with a proxy
connectivity details
mid server
outbound soap security
soap message
soap message functions
soap messages
test the soap message
use soap message in a script
variable substitution
web service security
Outbound web services
  Mutual authentication
output variable
override execution plan
  script
packages call
  find
packages call removal tool
  activate
Packages Call Removal Tool
error types
Packages calls
PADomainUtils
  copy(String runAs)
  copyDashboard(String dashboardId, String runAs)
  copyJob(String paJob, String runAs)
  isWriteable(String table, String id)
  move(String runAs)
  PADomainUtils()
  PADomainUtils(String domainFrom)
  setFoundation(boolean foundation)
  setOverrides(Boolean overrides)
parent instance
parse currency
pass widget data
4021
pause script
4611
Perl
client
7443
examples
7443
web services
7443
plugin
Enhanced Web Service Provider - Common
7417, 7417
plugin activate
6146
plugin rollback
6146
Predictive Intelligence API
6788
preserve applications
117
preview problems
resolving
475
priority 1
example
incident
7487
incident
example
7487
ProbeHandlerCim
getCimQueries()
3783
getNamespace()
3783
getParameters()
3783
run()
3784
setProbeParameters(Object params)
3784
ProbeHandlerCim API
3783
ProbeHandlerMulti
addProbes(String parent, String sysID, GlideRecord probe)
3785
addProbes(String parent, String sysID)
3785
run()
3786
runProbeHandler(String params, String sysID, GlideRecord probe)
3786
setParameter(String parent, String name, String value)
addBoolean(String dbName, String paramName, String paramMap)
addDebug(String paramMap)
addJavascriptHandlers(String paramMap)
addOIDAutoResolve(String paramMap)
addOIDSpec(String paramMap)
addSensorHandler(String paramMap)
addStage(String paramMap)
addTimingSpecs(String paramMap)
getParameters()
run()
setProbeParameters(String params)
setProbeParameters(Object params)

processor
multi-table
processors
API components
creating
secure
simple
URL endpoints
pull data
local data store
Push Installation API
pushes and pulls
Team Development
put the class name here
Python
client
example
web services

Q
query

bn
query tables
4460

queue
get(sys_id)
3794

Queue
3794
join(String)
3795

Queue, Advanced Work Assignment (AWA)
3793

QuickRanges
createItem(String table, String id, String type)
3795
onMakeRanges()
3796

QuickRanges API
3795

RecordToHTML
RecordToHTML(String table, String sys_id, String pattern, Boolean link)
3796
setValue(String fieldName, String value)
3797
toString()
3798

RecordToHTML API
3796

reference
qualifiers
4472

Reference
qualifiers
4540

reference glide list
from a script
4430, 4431

RenderProperites
3829

RenderProperties

getEncodedQuery()
3829

getListControl()
3830

getParameterValue(String parameterName)
3830

getReferringURL()
3830

getViewID()
3831

getViewName()
3831

getWindowProperties()
3831

isInDevStudio()
3832

isInteractive()
3832

isManyToMany()
3833

isRelatedList()
3833

RequestAuthAPI
3837

resolving collisions
412, 413

ResponseTemplate
3842

query(String tableName, String recordId, String searchTerm, Number limit, Number offset, Boolean includeEvaluatedBody, String errorFormat, Object opts)
3843

render(String templateId, String tableName, String recordId, String errorFormat, Object opts)
3847

ResponseTemplate()
REST
6683, 7500
 REST - Inbound
660
REST & SOAP API analytics
7489
API naming
7493
collection and cleanup
7492
dashboards
7490
REST API
4979, 6330
Aggregate
5160
CORS domain requirements
4983
define a CORS rule
4984
explore a table
4972
Returning session debug
logs
4982
User Role Inheritance
7269
REST endpoint modification
2925
REST message
Basic auth profile
7501
Define a header
7498
elements
7495
REST message with basic
auth
7501
REST message with OAuth
7502
Testing HTTP methods
7498
Variable substitution
7513

REST Message
Define an HTTP method
7497
Generate a script preview
7514
Scripting
7514
REST OAuth
Google provider tutorial
7509, 7510, 7511
REST request modification
2925
REST sign results data
2911
RESTAPIRequestBody
3849
body
3849
getHeader(String header)
3849
getSupportedResponseContentTypes()
3850
headers
3850
pathParams
3850
queryParams
3851
queryString
3851
uri
3852
url
3852
RESTAPIRequest
3849
body
3849
getHeader(String header)
3849
getSupportedResponseContentTypes()
3850
headers
3850
pathParams
3850
queryParams
3851
queryString
3851
uri
3852
url
3852
RESTAPIRequestBody
3852
data
3854
dataStream
3854
dataString
3855
hasNext()
3855
nextEntry()
3856
RESTAPIResponse
3857, 3862
    getStreamWriter()
    3857
    setBody(Object body)
    3858
    setContentType(String contentType)
    3858
    setError(Object error)
    3859
    setHeader(String header, String value)
    3860
    setHeaders(Object headers)
    3860
    setLocation(String location)
    3861
    setStatus(Number status)
    3861
RESTAPIResponseStream
    writeStream(Object stream)
    3862
    writeString(String data)
    3863
RESTMessagev2
    execute()
    3865, 3865
    RESTMessageV2()
    3868
    RESTMessageV2(String name, String methodName)
    3869
    setHttpMethod(String method)
    3875
    setRequestorProfile(String requestorContext, String requestorId)
    3881
RESTMessageV2
    disableForcedVariableSubstitution()
setQueryParameter(String name, String value)
setRequestBody(String body)
setRequestBodyFromAttachment
setRequestHeader(String name, String value)
setStringParameter(String name, String value)
setStringParameterNoEscape(String name, String value)

RESTResponseV2
getAllHeaders()
getBody()
getCookies()
getErrorCode()
getHeader(String name)
getHeaders()
getQueryString()
getResponseAttachmentSysid()
getStatusCode()
haveError()
waitForResponse(Number timeoutSecs)

retrieve displayed values
choice list
retrieve value
role
rollup
Rosters
Rotas
RoundingInterval
getRoundedValue(Number value)
RoundingInterval(String type)

RoundingInterval API
routing
RSS Feed Generator
RSS Feed Reader
rules
run automated test
runtime access

S
Schedule Pages
Schedules
Scoped Application
Scoped CartJS
Scoped administration
Scopes

Restricted caller access
43, 45, 46
addToCart(Map request)
CartJS(String cartName)
checkoutCart()
empty()
getCartID()
getCartItems()
getDeliveryAddress()
getRequestedFor()
getRequestedForDisplayName()
getSpecialInstructions()
orderNow(Map request)
setDeliveryAddress(String address)
setRequestedFor(String user)
setSpecialInstructions(String specialInstructions)
submitOrder(Map request)
updateItem(Map request, String cart_item_id)
Scoped CatalogClientScript
addScript(String script)
appliesToCatalogItem(Boolean flag)
appliesToRequestedItem(Boolean flag)
appliesToTargetRecord(Boolean flag)
Scoped CatalogJS
create(Boolean standardUpdate)
deleteRecord(String sysID, Boolean standardUpdate)
setAttributes(Map attributes)
setCatalogItem(String sysId)
setOnChangeVariable(String sysId)
setVariableSet(String sysId)
Scoped CatalogSearch
CatalogSearch()
search (String catalogID, String categoryID, String term, Boolean mobile, Boolean depthSearch)
Scoped CatCategory
availableForUserCriteria(String action, Array[] criteriaIDs)
notAvailableForUserCriteria(String action, Array[] criteriaIDs)
Scoped CatItem
availableForUserCriteria(String action, Array criterialDs)
1293
notAvailableForUserCriteria(String action, Array criterialDs)
1300
Scoped Client
Client()
1412
deleteSeries(GlideRecord now GR, String metric)
1412
put(DataBuilder metricData)
1413
Scoped CMDBGroupAPI
1470
Scoped Conversation
1546, 1546
addSubscriber(String sysID)
1547
get(String sysID)
1546
removeSubscriber(String sysID)
1548
Scoped Data
getEnd()
1625
getPeriod()
1626
getStart()
1626
Scoped DATA
getLabel()
1625
getMetricName()
1625
getSubject()
1627
getTableName()
1627
getValues()
1627
size()
1628
Scoped DataBuilder
1628
add(GlideDateTime start
Number value)
1629
add(GlideDateTime start,
Array value)
1628
DataBuilder(String table,
String subject, String metric)
1630
Scoped Flow
1854
startAsync(String scopeName.flowName,
Map flowInputs)
1854
Scoped Glide Element
getDecryptedValue()
2167
Scoped GlideAggregate
1901
addAggregate(String agg,
String name)
1902
addEncodedQuery(String query)
1903
addNotNullQuery(String fieldName)
1905
addNullQuery(String fieldName)
1906
addQuery(String name,
String operator, String value)
1904
addTrend(String fieldName,
String timeInterval)
1906
getAggregate(String agg,
String name)
1907
getAggregateEncodedQuery()
2009
gDayOfMonthNoTZ()
2010
gDisplayValue()
2010
gDisplayValueInternal()
2011
gMonthNoTZ()
2012
gValue()
2012
gYearNoTZ()
2013
GlideDate()
2013
setDisplayValue(String asDisplayed)
2014
setValue(String o)
2014
subtract(GlideDate start, GlideDate end)
2015
Scoped GlideDateTime
2068
add(GlideTime gd)
2069
add(Number milliseconds)
2069
addDaysLocalTime(Number days)
2070
addDaysUTC(Number days)
2071
addMonthsLocaltime(Number months)
2071
addMonthsUTC(Number months)
2072
addSeconds(Number seconds)
2072
addWeeksLocalTime(Number weeks)
2073
addWeeksUTC(Number)
2073
addYearsLocalTime(Number years)
2074
addYearsUTC(Number years)
2075
after(GlideDateTime gdt)
2075
before(GlideDateTime gdt)
2076
compareTo(Object o)
2077
equals(object dateTime)
2077
gDate()
2078
gDayOfMonthLocalTime()
2078
gDayOfMonthUTC()
2079
gDayOfWeekLocalTime()
2079
gDayOfWeekUTC()
2080
gDaysInMonthLocalTime()
2080
gDaysInMonthUTC()
2081
gDisplayValue()
2081
gDisplayValueInternal()
2082
gDSTOffset()
2082
gInternalFormattedLocalTime()
2083
gLocalDate()
2084
gLocalTime()
2084
gMonthLocalTime()
2085
getMonthUTC()
2085
getNumericValue()
2086
getTime()
2086
getTZOffset()
2087
getUserFormattedLocalTime()
2087
getValue()
2087
getWeekOfYearLocalTime()
2088
getWeekOfYearUTC()
2088
getFullYearLocalTime()
2089
getFullYearUTC()
2089
GlideDateTime()
2090
GlideDateTime(GlideDateTime g)
2090
GlideDateTime(String value)
2091
hasDate()
2091
isDST()
2091
isValid()
2092
onOrAfter(GlideDateTime gdt)
2092
onOrBefore(GlideDateTime gdt)
2093
setDayOfMonthLocalTime(Number day)
2094
setDayOfMonthUTC(Number day)
2094
setDisplayValue(String asDisplayed)
2095
setDisplayValue(String value, String format)
2095
setDisplayValueInternal(String value)
2096
setGlideDateTime(GlideDateTime g)
2097
setMonthLocalTime(Number month)
2097
setMonthUTC(Number month)
2098
setValue(String o)
2098
setValueUTC(String dt, String format)
2099
setYearLocalTime(Number year)
2100
setYearUTC(Number year)
2100
subtract(GlideDateTime start, GlideDateTime end)
2102
subtract(GlideTime time)
2101
subtract(Number milliseconds)
2102
toString()
2103
Scoped GlideDBFunctionBuilder
2103
add()
2104
build()
2104
concat()  
constant(String constant)  
datediff()  
dayofweek()  
divide()  
field(String field)  
GlideDBFunctionBuilder()  
length()  
multiply()  
subtract()  
 Scoped GlideDigest  
getMD5Base64(String source)  
getMD5Base64FromInputStream(GlideScriptableInputStream inputStream)  
getMD5Hex(String source)  
getMD5HexFromInputStream(GlideScriptableInputStream inputStream)  
getSHA1Base64(String source)  
getSHA1Base64FromInputStream(GlideScriptableInputStream inputStream)  
getSHA1Hex(String source)  
getSHA1HexFromInputStream(GlideScriptableInputStream inputStream)  
getSHA256Base64(String source)  
getSHA256Base64FromInputStream(GlideScriptableInputStream inputStream)  
getSHA256Hex(String source)  
getSHA256HexFromInputStream(GlideScriptableInputStream inputStream)  
add(GlideDuration duration)  
getByFormat(String format)  
getDayPart()  
getDisplayValue()  
getDurationValue()  
getRoundedDayPart()  
getValue()  
 Scoped GlideDuration  
GlideDuration()  
GlideDuration(GlideDuration another)  
GlideDuration(Number milliseconds)  
GlideDuration(String displayValue)  
setDisplayValue(String asDisplayed)
setValue(Object o)
2130
subtract(GlideDuration
duration)
2131
Scoped GlideElement
1112, 1152, 1709, 1828, 1853,
1923, 2160, 2196, 2338, 2602,
2656, 2807, 2831, 3261, 3307,
3893, 4090, 4163, 4267, 4271,
4279, 4298, 4304, 4313, 4315,
4315, 4336, 4360
changes()
2161
changesFrom(Object o)
2162
changesTo(Object o)
2163
dateNumericValue()
2163
getChoices(String
dependent)
2165
getChoiceValue()
2166
getDisplayValue(Number
maxCharacters)
2167
getED()
2168
getGlobalDisplayValue()
2168
getReferenceTable()
2171
getRefRecord()
2172
nil()
2173
setDisplayValue(Object
value)
2175
setError(String errorMessage)
2175
setPhoneNumber(Object
phoneNumber, Boolean
strict)
2176
setValue(Object value)
2176
toString()
2177
Scoped
GlideElementCurrency
getElementDisplayValue()
2179
Scoped
GlideElementCurrency2
2177
ggetAsEnteredDisplayValue()
2178
ggetReferenceDisplayValue()
2179
ggetSessionDisplayValue()
2181
setDisplayValue(String
displayValue)
2181
Scoped
GlideElementDescriptor
2182
ggetAttachmentEncryptionType()
2182
ggetEncryptionType()
2183
ggetInternalType()
2184
ggetLabel()
2186
ggetLength()
2186
ggetName()
2187
ggetPlural()
2188
hasAttachmentsEncrypted()
2188
isAutoOrSysID()
isChoiceTable() 2190
isEdgeEncrypted() 2190
isMandatory() 2191
isVirtual() 2192
Scoped GlideEmailOutbound 2192
  addAddress( String type, String address) 2193
  addAddress(String type, String address, String displayName) 2193
getSubject() 2194
getWatermark() 2194
GlideEmailOutbound() 2194
setBody(String bodyText) 2195
setFrom(String address) 2195
setReplyTo(String address) 2195
setSubject(String subject) 2196
Scoped GlideEvaluator 2202
Scoped GlideExcelParser 2204
Scoped GlideFlow 2225
Scoped GlideFormScratchpad 2276
Scoped GlideImpersonate 2285
Scoped GlideLocale 2362
  get() 2362
getDecimalSeparator() 2363
groupingSeparator() 2363
Scoped GlidePluginManager 2399
  isActive(String plugin_id) 2399
Scoped GlideQueryCondition 2443
  addCondition(String name, String oper, Object value) 2444
  addOrCondition(String name, String oper, Object value) 2444
Scoped GlideRecord 2531
  _next() 2569
  addEncodedQuery(String query) 2533
  addFunction(Object function) 2534
  addJoinQuery(String joinTable, String primaryField, String joinTableField) 2535
  addNotNullQuery(String fieldName) 2536
  addNullQuery(String fieldname) 2537
  addQuery(String name, Object value) 2539
  addQuery(String name, String operator, Object value) 2539
addQuery(String query)
canCreate()
canDelete()
canRead()
canWrite()
chooseWindow(Number firstRow, Number lastRow, Boolean forceCount)
deleteMultiple()
deleteRecord()
get(Object name, Object value)
getDisplayValue()
getElement(String fieldName)
getEncodedQuery()
getLastErrorMessage()
getLink(Boolean noStack)
getRecordClassName()
getRowCount()
getTableName()
getUniqueValue()
getValue(String name)
GlideRecord(String tableName)
hasNext()
initialize()
insert()
isActionAborted()
isEncodedQueryValid(String query)
isNewRecord()
isValid()
isValidField(String columnName)
isValidRecord()
newRecord()
next()
opration()
orderBy(String name)
orderByDesc(String name)
query(String field, String value)
setAbortAction(Boolean b)
setNewGuidValue(String guid)
setValue(String name, Object value)
setWorkflow(Boolean enable)
update(String reason)
2578
updateMultiple()
2579
Scoped GlideSchedule
2591
add(GlideDateTime
startDate, GlideDuration
offSet)
2591
duration(GlideDateTime
startDate, GlideDateTime
endDate)
2592
getName()
2592
GlideSchedule()
2593
GlideSchedule(String sysID,
String timeZone)
2593
isInSchedule(GlideDateTime
time)
2593
isValid()
2594
load(String sysID,
String timeZone, String
excludeSpanID)
2595
setTimeZone(String
timeZone)
2595
whenNext(GlideDateTime
time, String timeZone)
2596
Scoped GlideScopedEvaluator
getVariable(String name)
2203
Scoped
Scoped GlideScriptableInputStream
2597
Scoped
Scoped GlideScriptedProcessor
redirect(String url)
2598
writeJSON(Object o)
2598
writeOutput(String
contentType, String s)
2599
writeOutput(String s)
2598
Scoped
GlideSecureRandomUtil
2599
getSecureRandomInt()
2600
getSecureRandomIntBound(Number
bound)
2600
getSecureRandomLong()
2601
generateSecureRandomString(Number
length)
2601
Scoped GlideServletRequest
getContentType()
2604
getHeader(String name)
2605
getHeaderNames()
2605
getHeaders(String name)
2606
getParameter(String name)
2606
getParameterNames()
2607
getQueryString()
2607
Scoped GlideServletResponse
sendRedirect(String
location)
2608
setContentType(String type)
2608
setHeader(String key, String
value)
2609
setStatus(Number status)
2609
Scoped GlideSession
2615
cgetClientData(String paramName)
2615
cgetClientIP()
2616
cgetCurrentApplicationId()
2617
cgetLanguage()
2618
cgetSessionToken()
2619
cgetTimeZoneName()
2619
cgetUrlOnStack()
2620
isImpersonating()
2621
isInteractive()
2621
isLoggedIn()
2622
putClientData(String paramName, String paramValue)
2622
Scoped GlideSPScriptable
2623
canReadRecord(GlideRecord now_GR)
2623
canReadRecord(String table, String sysId)
2624
cgetCatalogItem(String sysId, Boolean isOrdering)
2627
cgetCatalogItem(String sysId)
2625
cgetDisplayValue(String fieldName)
2628
cgetField(GlideRecord now_GR, String fieldName)
2628
cgetFields(GlideRecord now_GR, String fieldNames)
2629
cgetFieldsObject(GlideRecord now_GR, String fieldNames)
2629
cgetForm(String tableName, String sysId)
2630
cgetKBCategoryArticles()
2630
cgetKBCategoryArticleSummaries(String sys_id, Number limit, Number maxChars)
2631
cgetKBCount(String sys_id)
2632
cgetListColumns(String tableName, String view)
2632
cgetMenuHREF(GlideRecord page)
2633
cgetMenuItems(String sysId)
2633
cgetParameter(String name)
2634
cgetPortalRecord()
2634
cgetRecord()
2635
cgetRecordDisplayValues(Object data, GlideRecord from, String names)
2635
cgetRecordElements(Object data, GlideRecord from, String names)
2636
getRecordValues(Object data, GlideRecord from, String names)
2636
getStream(String table, String sysId)
2641
getUserInitials()
2642
getValue(String name)
2642
gVALUES(Object data, String names)
2643
getWidget(String sysID, Object options)
2645
isUserCriteriaEnabled()
2647
mapUrlToSPUrl(String url)
2646
userCanSeeSearchSourceByCriteria(String searchSourceID)
2647
Scoped GlideSysAttachment
2664
copy(String sourceTable, String sourceID, String targetTable, String targetID)
2665
deleteAttachment(String attachmentID)
2665
gETAttachments()
2666
gETContent(GlideRecord sysAttachment)
2667
gETContentBase64(GlideRecord sysAttachment)
2667
gETContentStream(String sysID)
2668
GlideSysAttachment()
2669
write(GlideRecord record, String fileName, String contentType, String content)
2669
writeBase64(GlideRecord now_GR, String fileName, String contentType, String content_base64Encoded)
2670
writeContentStream(GlideRecord gr, String fileName, String contentType, GlideScriptableInputStream inputStream)
2671
Scoped GlideSysListControl
2672
gETControlID()
2672
GlideSysListControl(String tableName)
2672
isOmitEditButton()
2673
isOmitNewButton()
2673
Scoped GlideSystem
2749
addErrorMessage(String message)
2749
addInfoMessage(String message)
2750
base64Decode(String source)
2751
base64Encode(String source)
2750
dateGenerate(String date, String range)
2755
debug(String message, Object parm1, Object parm2, Object parm3, Object parm4, Object parm5)

2757

endOfNextWeek()

2759

error(String message, Object parm1, Object parm2, Object parm3, Object parm4, Object parm5)

2762

eventQueue(String name, Object instance, String parm1, String parm2, String queue)

2763

eventQueueScheduled(String name, Object instance, String parm1, String parm2, Object expiration)

2763

executeNow(GlideRecord job)

2765

generateGUID()

2765

getCallerScopeName()

2766

getCssCacheVersionString()

2767

getCurrentApplicationId()

2767

getCurrentScopeName()

2768

getEscapedMessage(String id, Object args)

2769

getMessage(String id, Object args)

2770

getProperty(String key, Object alt)

2772

getSession()

2772

getSessionID()

2773

getSessionToken()

2773

getTimeZoneName()

2774

getUrlOnStack()

2774

getUser()

2775

getUserDisplayName()

2775

getUserID()

2776

getUserName()

2776

hasRole(Object role)

2777

include(String name)

2779

info(String message, Object parm1, Object parm2, Object parm3, Object parm4, Object parm5)

2780

isDebugging()

2780

isInteractive()

2781

isLoggedIn()

2782

isMobile()

2782

nil(Object o)

2784

setProperty(String key, String value, String description)

2786

setRedirect(Object o)

2786

tableExists(String name)

2787
urlDecode(String url)
2788
urlEncode(String url)
2787
warn(String message,
Object parm1, Object
parm2, Object parm3,
Object parm4, Object
parm5)
2788
xmlToJSON(String xmlString)
2789
yearsAgo(Number years)
2789
Scoped GlideTableHierarchy
2792
getAllExtensions()    
2792
getBase()            
2793
getHierarchy()       
2793
getName()           
2794
getRoot()            
2794
getTableExtensions() 
2795
getTables()          
2795
GlideTableHierarchy(String
tableName)           
2796
hasExtensions()      
2797
isBaseClass()        
2796
isSoloClass()        
2797
Scoped GlideTextReader
2798
getEncoding()        
2798
GlideTextReader( GlideScriptableInputStream
inputStream)  
2798
Scoped GlideTime
2799
getByFormat(String format)
2799
defDisplayValue()     
2800
defDisplayValueInterna()     
2801
defHourLocalTime()      
2801
defHourOfDayLocalTime() 
2802
defHourOfDayUTC()       
2802
defHourUTC()           
2802
defMinutesLocalTime()   
2803
defMinutesUTC()        
2803
defSeconds()          
2804
defValue()           
2804
GlideTime()          
2804
GlideTime(Number
milliseconds)        
2805
setDisplayValue(String
asDisplayed)         
2805
setValue(String o)   
2806
subtract(GlideTime
startTime, GlideTime
endTime)             
2806
Scoped GlideUICompatibility
2829
getCompatibility()   
2829
GlideUICompatibility(String scopeName)
  isBlocked()
Scoped GlideURI
  GlideURI()
Scoped GlideUser
  getCompanyID()
  getDisplayName()
  getEmail()
  getFirstName()
  getID()
  getLastName()
  getName()
  getPreference(String name)
  getRoles()
  getUserRoles()
  hasRole(String role)
  isMemberOf(String group)
  savePreference(String name, String value)
Scoped GlideXMLUtil
Scoped IdentificationEngine
Scoped Interaction
Scoped JSON
  parse(String str)
Scoped Messaging
  send(GlideRecord messagingApplication, String recipient, GlideRecord messagingContent, GlideRecord target)
Scoped NotifyPhoneNumber
  getDialCode()
  getID()
  getNumber()
  getOwner()
  getTerritory()
  supportsConferenceCall()
  supportsIncomingPhoneCall()
  supportsIncomingSMS()
  supportsOutgoingPhoneCall()
  supportsOutgoingSMS()
  supportsRecording()
  supportsWebRTC()
Scoped OrderGuide
Scoped Queue
Scoped RenderProperties
  getEncodedQuery()
getListControl() 3834
getParameterValue(String parameterName) 3834
getReferringURL() 3834
getViewName() 3835
getWindowProperties() 3835
isInDevStudio() 3836
isInteractive() 3836
isManyToMany() 3836
isRelatedList() 3837
Scoped SentimentAnalyser 3929
Scoped SPScriptedFacet
  addFacetItem(String label, Object valueObj) 4031
Scoped SPScriptedFacetService
  createFacet(String label, String id) 4032
  createMultiChoiceFacet(String label, String id) 4033
Scoped Subflow
  startAsync() 4090
Scoped TemplatePrinter
  space(Number spaces) 4162
Scoped Transformer
  execute(GlideDateTime start, GlideDateTime end) 4185
  groupBy(String field) 4185
  metric(String metricName) 4186
  Transformer(GlideRecord sourceRecords) 4187
Scoped TransformerRuleList 4191
Scoped UserCriteria 4244
  create(Object columnValues, Boolean standardUpdate) 4244
  deleteRecord() 4245
  read(String[] columns) 4245
  setActive(Boolean active) 4246
  setAdvanced(boolean advanced) 4246
  setCompanies(String companies) 4247
  setDepartments(String departments) 4247
  setGroups(String groups) 4248
  setLocations(String locations) 4248
  setMatchAll(Boolean match_all) 4249
  setName(String name) 4249
  setRoles(String roles) 4250
  setScript(String script) 4250
  setUsers(String users)
update(Object columnValues, String reason)

UserCriteria()

UserCriteria(String sys_id)

Scoped Workflow

debug(String message, Object args)

error(String message, Object args)

getVariable(String name)

info(String message, Object args)

inputs

name()

removeVariable(String name)

result

scratchpad()

setResult(String result)

setVariable(String name, Object value)

warn(String message, Object args)

Scoped XMLDocument2

createElement(String name)

createElementWithTextValue(String name, String value)

getDocumentElement()

getFirstNode(String xPath)

getNextNode(Object current)

getNode(String xPath)

getNodeText(String xPath)

parseXML(String xmlDoc)

setCurrentElement(XMLNode element)

setNamespaceAware(Boolean aware)

toString()

XMLDocument2(GlideScriptableInputStream inputStream)

XMLDocument2()

Scoped XMLNode

getAttribute(String attribute)

getAttributes()

getChildNodeIterator()

getFirstChild()

getLastChild()

getNodeName()

getNodeValue()
Scoped XMLNodeIterator
4333
hasNext()
4334
next()
4334
Scoped, Global HistoryWalker
2889
ScopedGlideScriptedProcessor
2597
ScopedGlideServletResponse
2608
script access
28, 29
script debugger
4597
User interface
4602
Script Debugger
4606, 4611, 4613, 4616
Script Debugger status
4613
script field highlighting
4636
script include
646, 7401, 7402
AJAXClientTiming
4523
script includes
protection policy
64
Script Includes
change privacy
4435, 4436
client-callable
4435, 4436
privacy settings
4435
script syntax error
check
4397
Scriptable Service Catalog Variables
4473
Scripted REST API
streaming vs large objects
example
7339
scripted REST APIs
7326
Define query parameters
7325
enable versioning
7328
Require resource ACL
7327
service headers
7324
Scripted REST APIs
7318
add a version
7328
content type
7329
create
7323
create a resource
7323
Examples
7334
Maximum request size
7330
RESTAPIRequest
3849
RESTAPIRequestBody
3852
RESTAPIResponse
3857
RESTAPIResponseStream
3862
script samples
7334
streaming file attachments
example
Scripted Transformer
4213
scripted web service
7400, 7400
scripting
  API suggestions
  4495
business rules
  4432
messages
  4385
Scripting
  business rules
  4422, 4425, 4426, 4428,
  4429, 4429, 4429, 4431
debug
  4625
ScriptLoader
  getScripts(Array scripts,
    Function callback)
    3928
  getScripts(String filePath,
    Function callback)
    3927
ScriptLoader API
  3927
scripts
  4452
Scripts - Background
  4619
Selectable areas
  7485
scroller
  adding
    news
    7488
news
  adding
    7488
scrolling
  adding
    elements
    7486
    news panel
Server API
4411
server scripting
4452
  DurationCalculator
  4453
Server Scripting
  DurationCalculator
  4456, 4458
script include form
4433
script includes
  4433, 4434, 4435, 4436

service catalog
648
  client scripts
  4512
Service Catalog
  Client Scripts
  4511, 4512, 4513, 4514
Service Catalog API
6928
  service catalog checkout
    adding
      field
      4541
Service Catalog in Service
  Portal
  783, 784, 786, 787, 788, 789,
  790, 792, 794, 798, 800, 803,
  880, 893
Service Creator
  373, 373, 374, 383
    installed with
    375
service design interface
  publish a service
  390
ServiceNow studio
  automatic recovery
  336
  draft records
  336
ServiceNow Studio
  317, 324, 325
  automatic recovery
  334
  draft records
  334

ServiceRegistryQuery
  findForPortProbe(PortProbe
  portProbe)
  3935
  get(Object source)
  3936
  getShazzamQuerierClassname()
  3936
ServiceRegistryQuery(Object
  source)
  3937
  toPortProbeMap()
  3936
  toPortProbeXml()
  3937
ServiceRegistryQuery API
  3935
Set authorization credentials
  1216
set breakpoints
  4619
share applications
  applications
  123
  publish
  123
  store
  123
ShazzamLaunch
  addPortProbe(String
  midPPs, String portProbes,
  IPIncludeExcludeCollection
  ipiec)
  3937
  addRange(IPCollection ipc,
    String portProbes, Boolean
    samePhase)
  3938
  fireShazzamProbe(String
    portProbe, String ip, String
    sensor)
launch(String ip, Boolean samePhase, String sensor)
makeProbeSpec(String portProbe)
ShazzamLaunch(String status, String schedule)

ShazzamLaunch API

ShazzamResult
active
alive
domainName
hostName
ip
scanners

ShazzamResult API

ShazzamScanner
contents
domainName
hostName
name
port
portProbe
protocol
result
service

ShazzamScanner API

Sign endpoint

SkillDeterminationUtils
assignSkillsToWorkItem()
determineWorkItemSkills()

SkillDeterminationUtils()

SLA Repair API

SLARepair
repairByFilter(String filter, String sourceTable)
repairByGlideRecord(GlideRecord now_GR)
repairBySysId(String sysId, String sourceTable)

setAuditEnabled(Boolean onOrOff)
setRunWorkflow(Boolean onOrOff)
setValidateOnly(Boolean, onOrOff)

SLPServiceRegistryQuery
getShazzamQuerierClassName()
toPortProbeMap()

SLPServiceRegistryQuery API

SMS configuration

SMSPreferenceHandler
getConfig()

SMSPreferenceHandler(String provider)

SNEventSender

SNI support

SNMPGatherDocParts

gather(String doc, Object probe)

SNMPGatherDocParts API

SNMPNetworkInterfaces

process(String cmdb_ci, String doc, Boolean isComputer, String agentName, String ecc_sys_id)

SNMPNetworkInterfaces API

SNMPResponse

buildIndex(Object cur, String name, String extName)

buildReferences()

getAmbiguousOIDs(String refBase, String refLeaf)

getOIDInt(String name)

getOIDObj(String name)

getOIDTable(String parentName, String entryName)

getOIDText(String name)

normalize(Object cur)

SNMPResponse API

SOAP

Outbound web service

Retrieve a large number of records

security

web service

SOAP default security policy

SOAP message

hierarchical

SOAP requests, signed

SOAP session report

view

SOAP strategies

SOAP web service

FAQ

PERL script, example

security

SOAP message, example

SOAP Web Services

scripted web services

SOAPMessagev2

execute()

executeAsync()
SOAPMessageV2
4003
  Asynchronous message example
    7533
Direct message example
  7532
getEndpoint()
  4004
getRequestBody()
  4005
getAddressHeader(String headerName)
  4005
getAddressHeaders()
  4006
MID Server example
  7534
Recordless message example
  7535
setBasicAuth(String userName, String userPass)
  4006
setEccCorrelator(String correlator)
  4007
setEccParameter(String name, String value)
  4008
setEndpoint(String endpoint)
  4008
setHttpTimeout(Number timeoutMs)
  4009
setLogLevel(String level)
  4009
setMIDServer(String midServerName)
  4010
setMutualAuth(String profileName)
  4010
setRequestBody(String requestBody)
  4011
setRequestMethodHeader(String headerName, String headerValue)
  4011
setSOAPAction(String soapAction)
  4012
setStringParameter(String name, String value)
  4013
setStringParameterNoEscape(String name, String value)
  4013
setWSSecurity(String keystoreId, String keystoreAlias, String keystorePassword, String certificateId)
  4014
SOAPMessageV2()
  4014
SOAPMessageV2(String soapMessage, String soapFunction)
  4015
SOAPResponseV2
  4015
getAllHeaders()
  4016
getBody()
  4016
getCookies()
  4017
getErrorCode()
  4017
getErrorMessage()
  4018
getHeader(String name)
  4018
getHeaders()
  4019
getStatusCode()
  4019
haveError()
waitForResponse(Number timeoutSecs)

source control
363, 366, 367
apply remote changes
360
available operations
346
branch
365
commit changes
361
import application
348
link application
351
switch branch
364
view commit history
368
spAriaUtil
4020
sendLiveMessage(String message)
4020
spContextManager
4021
addContext(String key, Object context)
4022
getContext()
4022
getContextForKey(String key, Boolean returnPromise)
4023
updateContextForKey(String key, Object context)
4024
spModal
4025
alert(String message).then(fn)
4025
confirm(String message).then(fn)
4026
open(Object options).then(fn)
4027
prompt(String message, String default).then(fn)
4030
SPScriptedFacet
4031
SPScriptedFacetService
4032
spUtil
4034
addErrorMessage(String message)
4034
addInfoMessage(String message)
4035
addTrivialMessage(String message)
4035
createUid
4036
format(String, Object)
4038
get(String widgetId Object data)
4036
getHeaders()
4037
getHost()
4037
getPreference(String preference, Function callback)
4037
getURL()
4038
recordWatch(Object $scope, String table, String filter, Function callback)
4040
refresh(Object $scope)
4041
setPreference(String preference, String value)
4042
setSearchPage(String page)
4043
update(Object $scope)
4043
SQL Server
2008
7584, 7585
2012
7584, 7585
Linked Server
7584
StandardCredentialsProvider
getCredentialBylID()
4045
credentials()
4044
StandardCredentialsProvider()
4044
StartDiscovery
addAdditionalIP(String status, String ip, String sensor)
4046
discoverFromSchedule(String ip, Boolean samePhase, String sensor)
4046
cancelScript()
4047
scheduleCancelJob()
4047
startFromIP(DiscoverySchedule schedule, String ip)
4048
startFromSchedule(GlideRecord schedule, GlideRecord job)
4048
StartDiscovery API
4046
363, 366
state change
form color
4547
StateManagementScriptableApi
addbulkCIAction(String requestorId, String sysIdList, String ciActionName, String ciActionListOld, String leaseTime)
4049
extendCIActionLease(String requestorId, String ciSysId, String ciActionName, String leaseTime)
4050
cIActions(String ciSysId)
4050
cIActions(String ciSysId)
4051
isCompatibleCIAction(String actionName, String otherActionName)
4051
isLeaseExpired(String requestorId, String ciSysId, String ciActionName)
4052
isNotAllowedAction(String ciType, String opsLabel, String actionName)
4052
isNotAllowedOpsTransition(String ciType, String opsLabel, String transitionOpsLabel)
4053
isValidRequestor(String requestorId)
4053
registerOperator()
4054
removeBulkCIAction(String requestorId, String sysIdList, String ciActionName)
setBulkCIOperationalState(String requestorId, String sysIdList, String opsLabel, String opsStateListOld)
 unregisterOperator(String requestorId)

StateManagementScriptableApi API

Static WSDL
7400, 7401, 7401
Static WSDL
7400, 7400, 7402
step description script
939
step in
4606
step out
4606
step over
4606
step-through script
4606
StopWatch
getTime()
4057
restart()
4057
StopWatch()
4057
StopWatch(Date initialDate)
4058
toString()
4058
StopWatch API
4056
string translation
1720, 1753
Studio
automatic recovery
335
draft records
335
subscriptions
enforcing
72

SummaryTableWriter
generate()
4092
setQuery(String query)
4092
setStackOn(String column)
4093
setTitle(String title)
4093
setTrend(String field, String interval)
4094
SummaryTableWriter(String tableName, String column)
4094
tableName)
syntax editor
4388
editing JavaScript scripts
4391
keyboard shortcuts
4398
navigate to a line number
4391
script macros
4389
syntax editor plugin
4389
sys_id
7447
system clone
117
system update sets
442

T
Table API
7181
explain a table
4972
table cleanup
621, 623
Tables
14, 14, 15, 15, 4460
access settings
32
TableUtils
\begin{itemize}
\item drop(String tableName)
\item dropAndClean(String tableName)
\item dropTableAndExtensions(String tableName)
\item getAbsoluteBase()
\item getAllExtensions()
\item getHierarchy()
\item getTableExtensions()
\item getTables()
\item hasExtensions()
\end{itemize}

Task Active State
Management business rule
4426

TaskStateUtil
ATTR_DEFAULT_CLOSE
4179
ATTR_DEFAULT_WORK
4178
ATTR_INACTIVE_STATES
4179
getDefaultCloseState()
4179
getDefaultWorkState()
4180
getInactiveStates()
4180
isStateInactive(String state)
4181
runMarkClosed
4181
runTaskCloser
4182
runTaskReopener
4182
setDefaultWorkState(String defaultWorkState)
4182
SYSTEM_DEFAULT_CLOSE
4183
SYSTEM_DEFAULT_WORK
4183
SYSTEM_INACTIVE_STATES
4183
TaskStateUtil(GlideRecord task)
4184

TaskStateUtil API
4177

Team Development
390
access rights
418
approve or reject push
397
back out
397, 409
best practice
391
change the parent instance
399
getInnerSegmentClass()
4166
getInnerSegmentEndTimeMs()
4167
getInnerSegmentStartTimeMs()
4167
getIsChanged()
4167
getPointIconClass()
4168
getPredecessors()
4168
getSpanColor()
4169
getSpanText()
4169
getStartTimeMs()
4170
getSysId()
4170
getTable()
4170
getTooltip()
4171
setAllowXDragLeft()
4171
setAllowXDragRight()
4172
setAllowXMove()
4172
setAllowYMove()
4173
setAllowYMovePredecessor()
4173
setInnerSegmentClass()
4174
setInnerSegmentTimeSpan()
4174
setPointIconClass()
4175
setSpanColor()
4175
setSpanText()
4176
setTimeSpan()
4176
setTooltip()
4177
tracked customizations
442
transaction details
4616
transactions
4616
Transform log
2827
Transform tabular format
4213
Transformer
4216
getRow()
4215
transform()
4216
Transformer structured
documents
4213
TransformerDefinition
4187
recordPath()
4187
transformerDefinition(Object
transformerRuleList, String
recordPath)
4188
validate()
4190
TransformerRuleList
addRule(String rule, String
path)
4192
setName(String name)
4193
theApplyMap(Object map)
4193
thenAdaptCurrency(String
currencyCode, Boolean
outputNumericCurrencyValue)
4194
thenAdaptDuration(String inputDuration, String outputDuration)  
thenAdd(Number operand)  
thenApplyPattern(String matchPattern, String outputPattern)  
thenCeiling(Number operand)  
thenConcat(String value)  
thenConcatSymbol(String symbol)  
thenDivideBy(Number operand)  
thenDivideInto(Number operand)  
thenFloor(Number operand)  
thenFormat(String matchPattern, String outputPattern)  
thenMax(Number operand)  
thenMin(Number operand)  
thenMultiply(Number operand)  
thenReplace(String matchString, String replaceString)  
thenRoundUp(Number operand)  
thenRoundDown(Number operand)  
thenSplit(String splitPattern, String replaceString)  
thenSubtract(Number operand)  

TransformerRuleList API  
TransformPart  
add(Number constant)  
avg()  
bottom(Number count)  
ceil(Number ceiling)  
collect()  
count()  
div(Number constant)  
filter(Object aggregator, Object duration)  
floor(Number floor)  
fractiles(Array fractions)  
getResult()  
groupBy(String field)  
interpolate(Object count)  
iqr()  
label(String label)  
limit(Object count)
log(Number base)
max()
median()
metric(String metric)
min()
mul(Number constant)
partition(String aggregator, GlideDateTime duration, GlideDateTime base)
resample(Number count)
resample(Number min, Number max)
resample(String aggregator, GlideDuration duration)
resample(String aggregator, Number numValues)
round(Number precision)
stddev()
sub(Object constant)
sum()
top(Number count)
TransformResult
byGroup()
getByLabel(String label)
getData()
deleting 482
getting started 446
introduction 441
load from XML 468
merging 461
navigating between records 459
overwriting during upgrade 452
previewing 472
procedures 454
reports 460
retrieving 470
review previews 473
save as XML 467
select in System Settings 456
selecting 454
transferring 469
view and compare customizations 458
update sets
default 445
UPSHandler
classifyAndIdentify() 4243
UPSHandler API 4243
URL query
execute script 4451
usage monitoring 72
use case 633, 633
use interactive SQL ODBC 7582
useful scripts
Modify a GlideDateTime Field Value 4563
useful approval assignment scripts 4567
Useful Scripts
abort a database action in a business rule 4580
calculating durations 4586
default before-query business rule 4581
scheduling scripts 4578
useful field scripts 4572
workflow use case 4588, 4588, 4590, 4592, 4592, 4593
user interface 4443
User interface
script debugger 4602
User interface elements 14
user object
get 4540
User Role Inheritance API 7269
user roles
user session
UserSkillAnalyzer
UserSkillRanking

versions

Versions

view progress automated test

VMUtils

VMUtils API

Walkworkflow

Web Service

web service API
osCleanupName(String name)

WindowsOSNameHelper API

wizard script
examples
record generators

wizard scripts

Wizard UI policy

Workflow

broadcastEvent()
cancel()
cancelContext()
deleteWorkflow()
fireEvent()
fireEventById()
getContexts()
getEstimatedDeliveryTime()
getEstimatedDeliveryTimeFromWFVersion()
getReturnValue()
getRunningFlows()
getVersion()
getVersionFromName(String workflowName)
getWorkflowFromName()

hasWorkflow()
restartWorkflow()
runFlows()
startFlow()
startFlowFromContextInsert()
startFlowRetroactive()

workflow scratchpad
access from business rules

WorkflowDuration

addSeconds()
calculate()
calculateTimeLeft()
getEndDateTime()
getSeconds()
getTotalSeconds()
setEndDateTime()
setStartDateTime()
setUsedSecs()
setWorkflow()

WorkflowModelManager

getActivityHistoryRecordById()
getAllTransitionedIntoActivity()
getAllTransitionedIntoActivityId()

getActivityHistoryRecordByld()
getExecutedHistory()
4306
getFinalExecutedActivityIdList()
4307
getFinalExecutedActivityList()
4307
getNextByExecutedOrder()
4308
getNextByExecutedOrderId()
4309
getNextByTransitionId()
4309
getPreviousByExecutedOrder()
4310
getPreviousByExecutedOrderId()
4311
getPreviousByTransition()
4312
getPreviousByTransitionId()
4312
WorkflowModelManagerAjax
4313
getExecutedHistory()
4313
getFinalExecutedActivityIdList()
4314
getRolledBackActivityIdList()
4314
WorkflowScheduler
4315
run()
4315
WorkflowTimeline
4315
WS-security
error messages
7431
WS-Security
security policy
7419
WSDL Schema
hierarchical data
7414
XML, JSON, CSV, RSS, schema,
PDF
7463
XMLDocument Script Object
4490, 4494
XMLStreamingAPI
4336
XMLStreamingBuilder
4360
XMLUtilJS
escapeForXMLText(String
text)
4369
stringToValue(String str)
4369
unescapeForXMLText(String
text)
4369
valueToString(String
XMLValue)
4370
XMLUtilJS API
4368

X